Online Health Monitoring System

Deliverables I

COP 4331, Fall, 2014

Team Name: Team 14

Team Members:

- Jon Carelli - email - page
- Chris McCue - email - page
- Chris Chaffin - email - page
- Ethan Pitts - email - page
- David Gundler - email - page
- James Luke - email - page

Contents of this Document

Deliverables I

Online Health Monitoring System

Concept of Operations

COP 4331, Fall, 2014

Modification history:

Version	Date	Who	Comment
v0.0	08/15/00	G. H. Walton	Template
v1.0	09/08/14	Ethan Pitts	Filled out information for the concept of operations.
v2.0	09/15/14	C. N. McCue	Major revisions- expanded many sections

Team Name: Team 14

Team Members:

- Jon Carelli - email - page
- Chris McCue - email - page
- Chris Chaffin - email - page
- Ethan Pitts - email - page
- David Gundler - email - page
- James Luke - email - page

Contents of this Document

The Current System

Currently, there are many barriers to patient-doctor communication. The American health care system has a bureaucratic structure which hinders doctor access. Because of legal issues, only a doctor can give medical advice or answer questions about pill schedules. Taking pills is confusing. There are so many unimportant medical warnings that are confusing to the patient. It is confusing whether “take daily twice” means to take two pills each day or take one pill for two days. What is a confused patient to do? Realistically, these minor services should not require a doctor’s visit.

Often, there are waiting times of weeks for a patient to see his or her doctor. Also, doctor to patient communication is kept minimal at the doctor's office. Patients wait in multiple waiting rooms to finally spend 15 minutes with a doctor. Doctors then hurry out to the next patient. Besides an expensive, elusive, time-consuming visit to the doctor's office, it is nearly impossible for doctors to communicate directly with their patients. Patients are usually seeking medical care because of discomfort. This inefficient health care system exasperates and confuses already uncomfortable and vulnerable patients.

The Proposed System: Needs

There was a time when the doctor-patient relationship was a much more intimate relationship. The ingredients of this relationship were ready access, good communication and common sense. Government attempts to reform health care, such as ObamaCare, have been miserable failures. It is nearly impossible to reform the health care bureaucracy from the top down. A more realistic way to reform the health care system is with minor reforms from the bottom up. Wal-mart has done this with $4 prescriptions. Others, such as Publix and Target, followed suit. Currently, it seems that the most effective health care reforms are bottom up reforms coming from the private, business sector.

It is unrealistic to think that the traveling family doctor of old would make a comeback. However, technology has greatly improved communications. An America can use Skype to communicate with a friend in mainland China. Texts can be sent while one is endangering others on the road. Drivers wait at green lights talking on their cell phones. Global communication technology has made the world a much smaller place. While technology can be annoying, most of the time it is very beneficial.

It is proposed that a system be developed with the objective of improving health care communication and decreasing bureaucratic confusion. An Online Health Monitoring System (OHMS) is proposed. The system will improve doctor-patient communication by taking advantage of high-tech communications that are readily available. SMS text messaging and video chat technologies will be integrated to facilitate better doctor-patient communication. The system will also make minor tasks, such as pill taking and appointment scheduling, more user-friendly. Instead of reading confusing pill labels and traversing the bureaucratic maze of appointment making, these minor tasks will be executed via an online interface.

The Proposed System: Users and Modes of Operation

There will be two users: doctors and patients. One can log in to the system as either a doctor or patient.

Doctor:

The doctor user should have multiple patients assigned to him or herself. The doctor should be able to request appointments with patients, send messages to patients and video call patients. The doctor should also be able to schedule when a patient takes his or her medication.

Patient:

The patient user should be assigned to a doctor. The patient should be able to request appointments with the doctor, send messages to the doctor, video call the doctor and see his or her medication reminders. The patient should also receive SMS messages concerning taking medication and reminding the patient of upcoming appointments.

The Proposed System: Operational Scenarios

Typical Patient Scenario:

Jim has back pains. His doctor has prescribed a medicine regiment for him. Jim and his wife will be out of the house all day. He logs into the Online Health Monitoring System and writes down his pill schedule for that day from the calendar. Jims

puts all the needed pills for the day into a container and brings his cell phone with him. He will receive texts throughout the day concerning when to take each pill.

Atypical Patient Scenario:

Sheila is taking medicine for her heart condition. She has the symptoms of another heart attack. As with all medical products, there will be warnings and disclaimers not to use this system for emergency conditions. Sheila should call 911. She should not use the video chat or text messaging. The system is only for minor, routine tasks and issues.

Typical Doctor Scenario:

This system will require time on the doctor's part. Doctors must perceive some benefit to using this system, such as time saved elsewhere. Everyday, from 1 p.m. to 2 p.m., Doctor Wayne update his patient pill schedules and appointments. If any extra time is left over, he sends texts and video chats with patients.

Atypical Doctor Scenario:

Doctor Watley is on vacation and he forgot to notify his patients. His patients have begun relying exclusively on his pill schedule and appointments from the system. There must be a disclaimer about relying exclusively on the system. It is only meant to be a small convenience.

The Proposed System: Operational Features

Must Have:

- Doctor ability to create pill schedules
- Patients should be reminded of when to take pills via text messaging
- Doctor ability to create appointments
- Patients should be reminded of hospital appointments via text messaging
- Ability for patient-doctor and doctor-patient text messaging
- Video chat

Would Like to Have:

- Patients should be able to view pill schedules
- Patients should be able to view appointments
- Patient ability to view doctor's calendar

The Proposed System: Expected Impacts

Although this system will not reform the health care system, it will provide small reforms in doctor-patient relations. Patients will find that the user-friendly system reduces confusion concerning pill schedules and reduces their anxiety in managing appointments. Patients will appreciate better communication with their doctor. This, in turn, will build more loyal patients. In the end, patients may begin to prefer doctors that use this system.

Once doctors are convinced of the benefits of this system, increasing numbers of doctors will also begin using the system. Doctors could fill out pill schedules and manage appointments as they are conversing with patients about these. Doctors will find that they will be able to provide better care for their patients. Doctors should find it more common for people to show up on time for appointments and for patients to take their medication at the recommended times. Overall, the most major impact of the system will be closer, improved doctor-patient relationships. This will make the doctor's job more pleasant.

This will put the patient at ease. “An apple a day keeps the doctor away”. However, if you need health care, “A login a day to the Online Health Monitoring System keeps the doctor near.”

The Proposed System: Analysis

Expected Improvements: better doctor-patient communication, less pill schedules confusion, less forgotten appointments

Disadvantages: doctors spend more time communicating with patients (doctors must see benefits to using the system), patients have to learn a new system

Limitations: limited features

Risks: patients using system communications for high risk situations, technical issues leading to missing important appointments or medications, busy doctors neglecting to use the system

Alternatives and Tradeoffs: more doctor communication time means less time to visit patients, the overheard of using this system must outweigh the disadvantages for doctors and patients

---

Template created by G. Walton (GWalton@mail.ucf.edu) on August 30, 1999 and last modified on August 15, 2000.

This page last modified by Chris McCue (christopher.mccue@knights.ucf.edu) on 9/15/2014

Online Health Monitoring System

Software Requirements Specification

COP 4331, Fall, 2014

Modification history:

Version	Date	Who	Comment
v0.0	9/1/2014	Jon Carelli	Uploaded Template
v0.1	9/1/2014	Jon Carelli	Added title details, team contact info, acronym descriptions, and product assumptions.
v0.2	9/3/2014	Jon Carelli	Added stakeholders, event table
v0.3	9/3/2014	David Gundler	Started 3.1,3.3
v0.4	9/5/2014	Jon Carelli	Added the Use Case Diagram
v0.5	9/7/2014	Jon Carelli	Continued work on Event Table.
v0.6	9/10/2014	David Gundler	Continued work on requirements 3.1 – 3.7
v0.7	9/10/2014	Jon Carelli	Cleaned up formatting, changed UCD requirements per client request, and edited Event Table.
v0.8	9/11/2014	Jon Carelli	Created Use Case Diagram descriptions and Section 4.0 Supporting Material
v0.8	9/13/2014	Jon Carelli	Grammar check, finalized requirements 3.1-3.7
v0.9	9/16/2014	C. N. McCue	Completed “evaluations” sections
v1.0	9/17/2014	D. Gundler	Completed “dependency” sections

Team Name: Team 14

Team Members:

- Jon Carelli - email - web page
- Chris McCue - email - web page
- Chris Chaffin - email - web page
- Ethan Pitts - email - web page
- David Gundler - email - web page
- James Luke - email - web page

Contents of this Document

SECTION 1: Introduction

Software to be produced:

The online health monitoring tool is an environment in which a patient can keep track of their scheduled medicine doses coupled with an alert system to their phone or email. The system encourages collaboration via email or video chat to the patient's physician and can be helpful for the physician to keep track of what they have prescribed and to whom. The goal of the system is to be simple and easy to use in order to make the website accessible and frequently utilized by patients.

Reference Documents:

- Concept of Operations
- Project Plan

Applicable Standards:

- RFC 2616. Hypertext Transfer Protocol -- HTTP/1.1, June 1999. Published by the Internet Engineering Task Force (http://www.ietf.org/rfc/rfc2616.txt).

Definitions, Acronyms, and Abbreviations:

- **HTML** – Hypertext Markup Language, a language to create a layout for a webpage
- **HTTP** – Hypertext Transfer Protocol, the protocol used by web servers and web browsers to communicate
- **PHP** – A programming language used to create dynamic websites.
- **SMS** – Short Message Service, the protocol used in most cell-phones for the exchange of short messages. This will be used to notify patients when to take their medicine.
- **MySQL** – An open source database management program. This will store user accounts and passwords.
- **jQuery** – A library for JavaScript that cuts down on the amount of code needing to be written.

SECTION 2: Product Overview

Assumptions:

- The developers assume the users and physician will be using a modern web browser such as Google Chrome, IE 8+, Firefox or Opera. We assume the website will be running on a single dedicated server, and that there is a system admin who has access to that server.

Stakeholders:

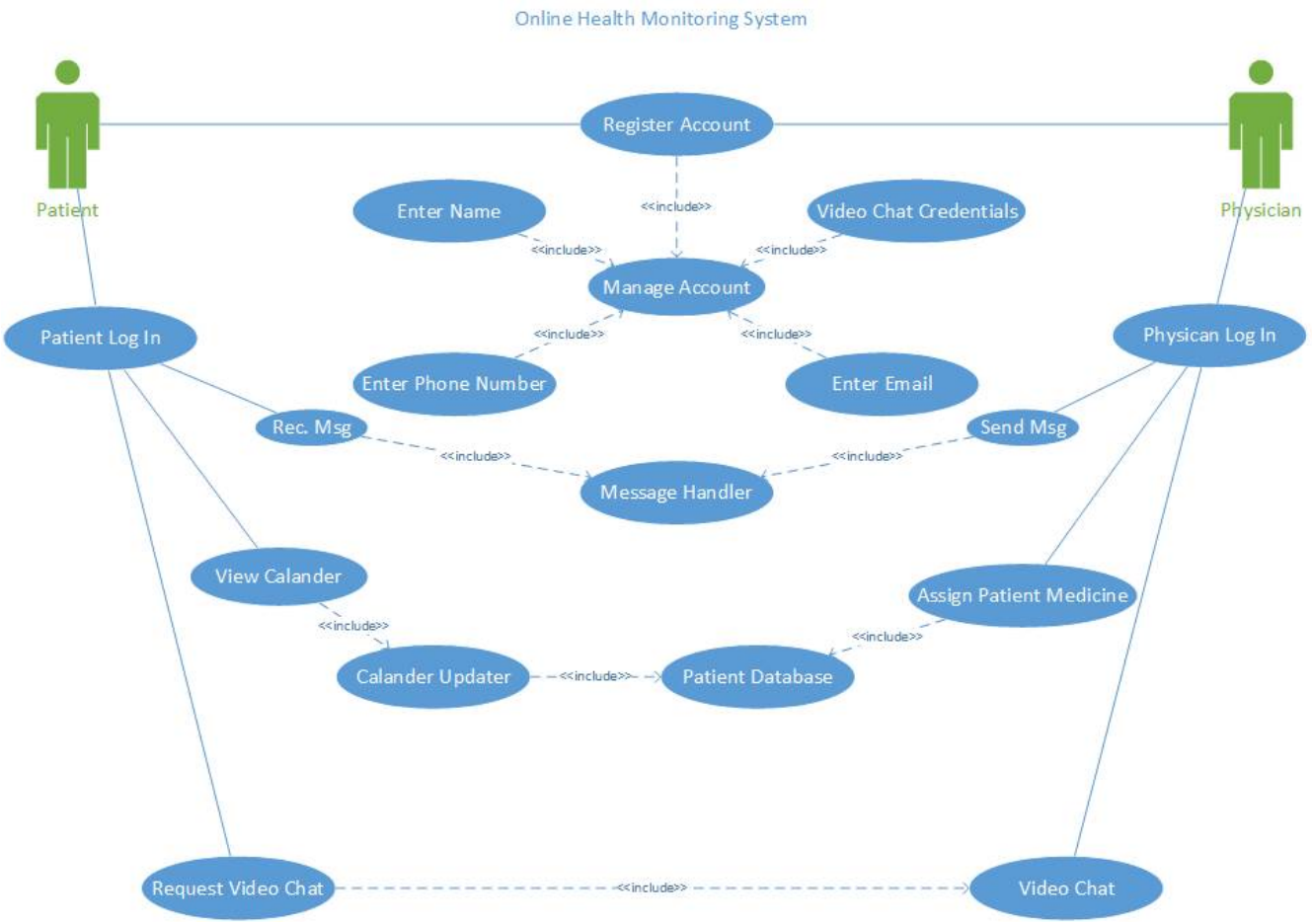
- **Client:** Dr. Turgut / Gurkan Solmaz gave the idea for the project and will be acting as client to decide what features will be implemented.
- **Patients**: Looking for an easy way to find out when to take medicine and be notified via SMS.
- **Physician:** Want an easy tool to keep track of patient's medicine intake.
- **Developers**: Want an efficient system as quickly as possible.

Event Table:

Event Name	External Stimuli	External Responses	Internal data and state
Patient Create Account	Clicking on "Patient Registration" button on the front page	Redirects to "New Patient" form	Creates a new Patient Node in respective database. User login and password are encrypted. Fill with provided information.
Log In Account	Entering account information in the appropriate area on the front page and pressing the "Log In" button.	Directs user to "User Page" page.	Access Patient Node in database. User supplied information is encrypted. Check credentials against those on file. Allows user access to their respective user page. Event manager updates user Calendar.
Physician Create Account	Clicking on "Physician Registration" button on the front page	Redirects to "New Physician" form	Creates a new Physician Node in respective database. Physician login and password are encrypted. Fill with provided information.
Physician Log In	Clicking the Physician Log In link, then entering information and pressing	Directs user to "Physician Page" page.	Access Physician Node in database. User supplied information is encrypted. Check

	the “Log In” button.		credentials against those on file. Allows user access to their respective user page.
Assign Patient Medicine	Finding patients name using drop down list, entering medicine name, entering date and time patient must take the medicine, clicking the ‘Assign Medicine’ button.	Directs physician to a completed request page, then auto re-loads the physician page.	Accesses the patient’s user node in database and updates medicine list and when it should be taken. Will also send information to SMS tool and queue based on time to send SMS message.
Send Message	Selecting a patient in a drop down list, entering a message in the text field and clicking the ‘Send Message’ button.	Showing a ‘Message Sent’ alert to the user.	Contacts the message handler with the Sender ID and Receiver ID, message handler begins process to send message to the Receiver ID.
Receive Message	No action needed	User will see a popup on their page displaying the message.	Message Handler finds user ID, runs script to display message on user’s page.
View Calendar	Log into patient page	Calendar will display on the patient page automatically with when the patient needs to take medicine.	Calendar updater will run when patient loads the patient page. Will check patient database for patient medicine schedule and update accordingly.
Video Chat	Log into patient page, click ‘Video Chat’ button, redirects to video chat page, can request a video chat with a physician.	The physician can choose to accept or not. If accepted, a video chat box will pop up and automatically connect the patient and physician.	None

Use Case Diagram:



Use Case Descriptions:

Register Account:
Prompts the user for a username, password and email. Creates a user account with the given information.
Exceptions:
See Functional Requirements 3.1.1

Patient Log In Account:
Prompts the user for a username, password. Log's into the Patient Login Page.
Exceptions:
See Functional Requirements 3.1.2

Physician Log In Account:
Prompts the user for a username, password. Log's into the Physician Login Page.
Exceptions:
See Functional Requirements 3.1.3

Manage Account:
Allows user to edit phone, name, email, and video chat credentials.
Exceptions:
Requesting user is not logged in or registered

Enter Name:
Allows the user to edit the name recorded in the database.
Exceptions:
Invalid name characters or length

Enter Phone Number:
Allows the user to edit the phone number recorded in the database.
Exceptions:
Invalid characters entered

Enter Email Address:
Allows the user to edit the email address recorded in the database.
Exceptions:
String is not an email address

Enter Video Chat Credentials:
Allows the user to edit or input their video chat credentials.
Exceptions:
Invalid credentials.

Send Message:
Allows the Physician to send a message to a patient
Exceptions:
Message too long

Receive Message:
Allows the Patient to receive a message.
Exceptions:
None

Message Handler:
Handles the sent message and passes it along to the patient
Exceptions:
None

Assign Patient Medicine:
Allows the Physician to assign medicine to an individual patient.
Exceptions:
Medicine name too long
Patient is already taking medicine at that time
Patient is already taking that medicine

Patient Database:
Stores patient information, including when and what medicine to take.
Exceptions:
None

Calendar Updater:
Checks patient database for newly added medicine, pushes update to patient Calandar.
Exceptions:
None

View Calendar:
Automatically loads calendar on patient log in page.
Exceptions:
Patient not logged in.

Request Video Chat
Patient clicks a link that opens video chat program, and automatically calls the Physician.
Exceptions:
Physician not online

Video Chat
A successful video chat begins
Exceptions:
None

SECTION 3: Specific Requirements

3.1 Functional Requirements:

No: 1
Statement: Creating new user Node in respective database and filling with the needed information.
Source: Creation of a new Account.
Dependency: No. 10
Conflicts: None
Supporting Materials: UML
Evaluation Method: When the program runs successfully, it creates the user node and fills out the newly created node with the given data.
Revision History: David Gundler , 9/07/2014 , Statement/Source/Supporting Materials

No: 2
Statement: Log patient into existing account by accepting information in appropriate area on the front page and comparing it to account information on the Users Node in the database.
Source: Patient login.
Dependency: No. 10
Conflicts: None
Supporting Materials: UML
Evaluation Method: When it runs, it can successfully compare the information and log in the patient.
Revision History: David Gundler , 9/07/2014 , Statement/Source/Supporting Materials

No: 3
Statement: Log patient into existing account by accepting information in appropriate area on the front page and comparing it to account information on the Physician Node in the database.
Source: Physician login.
Dependency: No. 10
Conflicts: None
Supporting Materials: UML
Evaluation Method: When it runs, it can successfully compare the information and log in the physician.
Revision History: David Gundler , 9/07/2014 , Statement/Source/Supporting Materials

No: 4
Statement: Physician accessing and modifying patient's calendar.
Source: Calendar Maintenance.
Dependency: No. 10; No. 9
Conflicts: None
Supporting Materials: UML
Evaluation Method: See if the Physician can access and modify the patient's calendar.
Revision History: David Gundler , 9/07/2014 , Statement/Source

No: 5
Statement: Patients access to calendar.
Source: Calendar Viewing.
Dependency: No. 10; No. 8
Conflicts: None
Supporting Materials: UML
Evaluation Method: When logged in as patient, see if a patient can access and view calendar.
Revision History: David Gundler , 9/07/2014 , Statement/Source

No: 6
Statement: Sending and receiving messages from Patients and Physician.
Source: Message Handler.
Dependency: None
Conflicts: None
Supporting Materials: UML
Evaluation Method: Test the systems capability to send and receive messages.
Revision History: David Gundler , 9/07/2014 , Statement/Source

No: 7
Statement: Handling live video chat between Patients and Physician.
Source: Video Chat
Dependency: None
Conflicts: None
Supporting Materials: UML
Evaluation Method: Test the systems capability to connect to the video chat.
Revision History: David Gundler , 9/07/2014 , Statement/Source

3.2 Interface Requirements:

No: 8
Specific interface for the Patients.
Source: Patient interface
Dependency: None
Conflicts: None
Supporting Materials: UML
Evaluation Method: Testing all the capability of the patient-users interface.
Revision History: David Gundler , 9/09/2014 , Statement/Source/Supporting Materials

No: 9
Specific interface for the Physician.
Source: Patient interface
Dependency: Node
Conflicts: None
Supporting Materials: UML
Evaluation Method: Testing all the capability of the Physician-users interface.
Revision History: David Gundler , 9/09/2014, Statement/Source/Supporting Materials

3.3Physical Environment Requirements:

No: 10
Statement: Server that is capable of holding all necessary data.
Source: Website
Dependency: No. 11
Conflicts: None
Supporting Materials: UML
Evaluation Method: Seeing if the program fits on the server and runs.
Revision History: David Gundler , 9/07/2014, Statement/Source/Supporting Materials

No: 11
Statement: Secure space for Server that has temperature control, safety from all liquids and air quality maintained (dust / humidity).
Source: Server Maintenance
Dependency: None
Conflicts: None
Supporting Materials: UML
Evaluation Method: Research into server location and review uptime and safety precautions.
Revision History: Jon Carelli , 9/13/2014, Statement

No: 12
Statement: Device that is capable of accessing website (computer/smartphone/tablet etc.).
Source: Website access.
Dependency: None
Conflicts: None
Supporting Materials: UML
Evaluation Method: Taking a device and see if it can successfully access the site and run.
Revision History: David Gundler , 9/07/2014, Statement/Source/Supporting Materials

No: 13
Statement: Location that has access to the internet.
Source: Website access.
Dependency: No. 12
Conflicts: None
Supporting Materials: UML
Evaluation Method: Checking if environment has hotspot or Ethernet connection.
Revision History: Jon Carelli, 9/13/2014

3.4 Users and Human Factors Requirements:

No: 14
Statement: Users-Patients must have basic understanding of web navigation and filling out form documentation for account creation. Must have steps to filling out form documentation.
Source: Users-Patients.
Dependency: None
Conflicts: None
Supporting Materials: UML

Evaluation Method: Take feedback from Patients
Revision History: David Gundler , 9/09/2014 , Statement/Source/Supporting Materials

No: 15
Statement: Users-Physician must have understanding of web navigation, documentation, and the ability to edit calendar information. Documentation of how to edit and modify calendar information.
Source: Users-Physician.
Dependency: None
Conflicts: None
Supporting Materials: UML
Evaluation Method: Take feedback from physicians
Revision History: David Gundler , 9/09/2014 , Statement/Source/Supporting Materials

No: 17
Statement: Software must maintain control of the access dependent on the type of user.
Source: Users-Access
Dependency: None
Conflicts: None
Supporting Materials: UML
Evaluation Method: Test different logins for each type of user.
Revision History: David Gundler , 9/09/2014 , Statement/Source/Supporting Materials

3.5 Documentation Requirements:

No: 18
Statement: On-line documentation for the patients for filling out the form and account editing.
Source: Users-Patients
Dependency: None
Conflicts: None
Supporting Materials: UML
Evaluation Method: "Help" button click opens up online help pdf.
Revision History: David Gundler , 9/09/2014 , Statement/Source/Supporting Materials

No: 19
Statement: On-line documentation for the physician in account editing and calendar editing.
Source: Users-Physician
Dependency: None
Conflicts: None
Supporting Materials: UML
Evaluation Method: "Help" button click opens up online help pdf.
Revision History: David Gundler , 9/09/2014 , Statement/Source/Supporting Materials

3.6 Data Requirements:

No: 20
Statement: Retaining the data of the user account information and calendar information. Must be retained for authentication for accessing one's account and maintaining all calendar information that will inform the patients of important events.

Source: Information/Servers
Dependency: No. 10
Conflicts: None
Supporting Materials: UML
Evaluation Method: Random quality reports will be compared with user reports.
Revision History: David Gundler , 9/09/2014 , Statement/Source/Supporting Materials

3.7 Resource Requirements:

No: 21
Statement: Admin to maintain the system.
Source: User-Admin
Dependency: None
Conflicts: None
Supporting Materials: UML
Evaluation Method: Admin to support monthly reports for accountability purposes.
Revision History: David Gundler , 9/09/2014 , Statement/Source/Supporting Materials

No: 22
Statement: Secure space for Server that has temperature control, safety from all liquids, air quality maintained (dust / humidity) and constant power reliability.
Source: Server
Dependency: None
Conflicts: None
Supporting Materials: UML
Evaluation Method: Trainer to check off this item on installation checklist.
Revision History: David Gundler , 9/09/2014 , Statement/Source/Supporting Materials

No: 23
Statement: • **Server and the following tools and software.** **HTML – Hypertext Markup Language, a language to create a layout for a webpage** **HTTP – Hypertext Transfer Protocol, the protocol used by web servers and web browsers to communicate** **PHP – A programming language used to create dynamic websites.** **SMS – Short Message Service, the protocol used in most cell-phones for the exchange of short messages. This will be used to notify patients when to take their medicine.** **MySQL – An open source database management program. This will store user accounts and passwords.** **jQuery – A library for JavaScript that cuts down on the amount of code needing to be written.**
Source: Server
Dependency: None
Conflicts: None
Supporting Materials: UML
Evaluation Method: Application to show green check for requirements met and red x for unmet.
Revision History: David Gundler , 9/09/2014 , Statement/Source/Supporting Materials

3.8 Security Requirements:

No: 23
Statement: Access must be controlled for patients so that patients may only view their own information.
Source: Security/User-patients
Dependency: None
Conflicts: None
Supporting Materials: UML
Evaluation Method: Attempt to hack into system.
Revision History: David Gundler , 9/09/2014 , Statement/Source/Supporting Materials

No: 24
Statement: Access to all patient files must be totally secure, including calendar, account information, and medicines.
Source: Security/User-patients
Dependency: None
Conflicts: None
Supporting Materials: UML
Evaluation Method: Attempt to hack into system.
Revision History: David Gundler , 9/09/2014 , Statement/Source/Supporting Materials

No: 25
Statement: Access to the server must be secure physical or otherwise.
Source: Security/server
Dependency: None
Conflicts: None
Supporting Materials: UML
Evaluation Method: Attempt to hack into system.
Revision History: David Gundler , 9/09/2014 , Statement/Source/Supporting Materials

No: 26
Statement: Patient and Physician usernames and passwords will be encrypted in the database
Source: Security/user-patients & user-physician
Dependency: None
Conflicts: None
Supporting Materials: UML
Evaluation Method: Attempt to hack into system.
Revision History: David Gundler , 9/09/2014 , Statement/Source/Supporting Materials

3.9 Quality Assurance Requirements:

No: 27
Statement: Server reliability and availability must be up 95% of the time and must always be secure.
Source: Server reliability.
Dependency: None
Conflicts: None
Supporting Materials: UML

Evaluation Method: Perform quality tests for a week gather statistics.
Revision History: David Gundler , 9/09/2014 , Statement/Source/Supporting Materials

.

No: 28
Statement: Calendar and messaging must have a reliability of 99.99%.
Source: Server reliability.
Dependency: None
Conflicts: None
Supporting Materials: UML
Evaluation Method: Test Plan to include 10,000 test cases. 9, 999 of these must pass.
Revision History: David Gundler , 9/09/2014 , Statement/Source/Supporting Materials

No: 29
Statement: Must have 99.99% reliability to reach the Physician.
Source: Reliability.
Dependency: None
Conflicts: None
Supporting Materials: UML
Evaluation Method: Test Plan to include 10,000 test cases. 9, 999 of these must pass.
Revision History: David Gundler , 9/09/2014 , Statement/Source/Supporting Materials

---

SECTION 4: Supporting Material

- Questions sent to client and clients response in second hand:

To: ☐jcarelli@knights.ucf.edu;

What are the most important requirements to be implemented?

Database shortage of data user accounts. Classes and what is encrypted and not encrypted. What type of database used.
Different interfaces for client and doctors.
One platform needs to be used like JavaScript.
SMS massage for the prototype needs to be done but can be done later.
No admin needed in prototype but in documentation it is needed if planning on adding later.

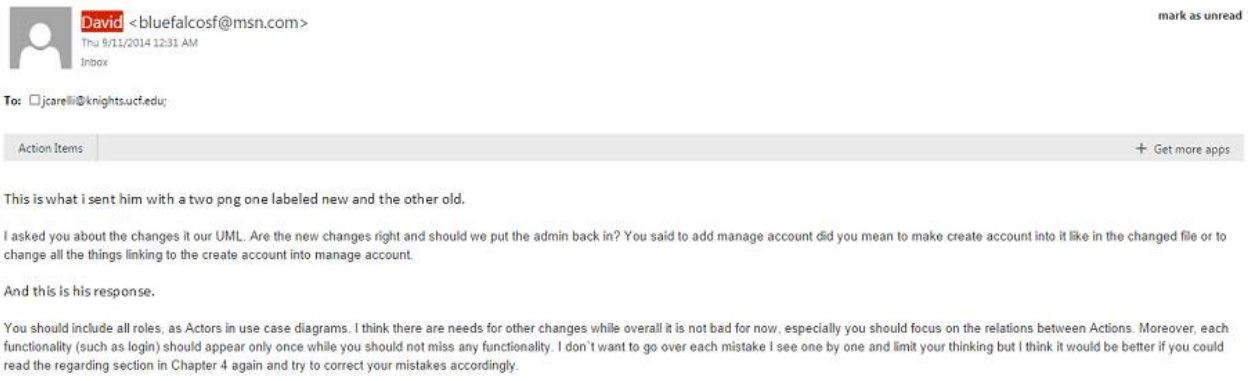
Clarify for the video chat, if clicking a link and utilizing Skype is an option?
Yes is okay.

Show him the Use Case Diagram and ask him if that works. (This might be a TA question, not a client question)?
Don't need the admin right now but okay.
Only one user. No extending from user to doctor actor and no extending from patient actor to user instead take out the user actor and have doctor and patient go to what user was pointing to.
Need to make manage account. Instead of having all the things that are coming off the create account change to manage account.

Other

How should doctors be able to track their patients?
Checking if medication is taken and some charts maybe needed.

Should we be able to respond through the SMS message?
Just to update the patient on events.



Template created by G. Walton (GWalton@mail.ucf.edu) on Aug 30, 1999 and last updated Aug 15, 2000

This page last modified by Jon Carelli (jcarelli@knights.ucf.edu) on 9/10/2014

Online Health Monitoring System

Project Management Plan

COP4331, Fall, 2014

Modification history:

Version	Date	Who	Comment
v0.0	08/15/00	G. H. Walton	Template
v1.0	8/30/14	C. N. McCue	First draft
V1.1	9/13/2014	C. N. McCue	Minor grammar and spelling corrections
v1.2	9/15/2014	C. N. McCue	Minor grammar and spelling corrections

Team Name: Team 14

Team Members:

- Jon Carelli - email - web page
- Chris McCue - email - web page
- Chris Chaffin - email - web page
- Ethan Pitts - email - web page
- David Gundler - email - web page
- James Luke - email - web page

Contents of this Document

Quality Assurance

Risk Management

Table of Work Packages, Time Estimates, and Assignments

PERT Chart

Technical Progress Metrics

Plan for tracking, control, and reporting of progress

Project Overview

The Online Health Monitoring System allows online management of patient health care. Doctors electronically manage patient care by setting health appointments and pill schedules. Patients are automatically sent text alerts concerning appointments and pills to take. Doctors and patients can communicate with each other via text messaging or video chat.

Reference Documents

- Concept of Operations
- Software Requirement Specification
- Test Plan

Applicable Standards

- Coding Standard:

  Code should be concise and readable. Methods will be no longer than one page. Indentation of four spaces will be required for each nested statement. Methods longer than ten lines will require documentation above the method. Variables and methods will be camel-cased. Class names will use upper case letters for each new word. Methods, classes and variables should clearly indicate what function they are performing.

- Document Standard:

  Document standards will be based off of the artifact templates. Font will be 10 point Times New Roman. Heading will be in bold-face Times New Roman 10 point font. Paragraphs will be single-spaced. There will be a double-spacing between paragraphs. All documents are expected to be proofread and checked using spelling and grammar checking. Modification history will include version, date, author and comment fields. The table of contents will list all the subdivisions of the document. Large figures and tables will be attached as an appendix.

- Artifact Size Metric Standard:

  Document size metrics will calculated based on the total of sections completed divided by total sections. For example, if seven out of fourteen sections are completed, the document will be considered 50% complete. For document revisions, if two subdivisions need revision and one is complete, 50% will be considered complete.

Presentations will be measured by slides complete divided by expected number of slides. If a presentation consists of twenty slides and five are completed, the presentation will be considered 25% complete.

Coding percentages will be based on the amount of classes and methods determined in the design phase. If there are ten methods expected and seven of those methods are complete, 70% will be considered completed.

Project Team Organization

The Project 14 Team members are Chris McCue, Jon Carelli, Chris Chaffin, Ethan Pitts, David Gundler and Luke James. Our goal is to divide the administrative and technical tasks equally among team members. Each member should have a main administrative artifact and module from the overall system. On the administrative side, Chris McCue is the project manager. Other team administrative tasks include Concept of Operations (Ethan), Software Requirements Specification (Jon and David) and the Test Plan (James Luke/Chris Chaffin). On the technical side, Jon Carelli has taken the initiative. Jon is also maintaining the team website.

We feel that it is crucial to maintain team communication throughout the life cycle of this project. Therefore, team members are required to weekly submit an individual log. As individual web pages are developed, the individual log will be posted in web page form. These will then be summarized in a team log. Based on the log, corrective actions will be taken to ensure deadlines are met. As needed, team members will communicate by email and cell phone. Face-to-face meetings will be held concerning each major deliverable.

Deliverables

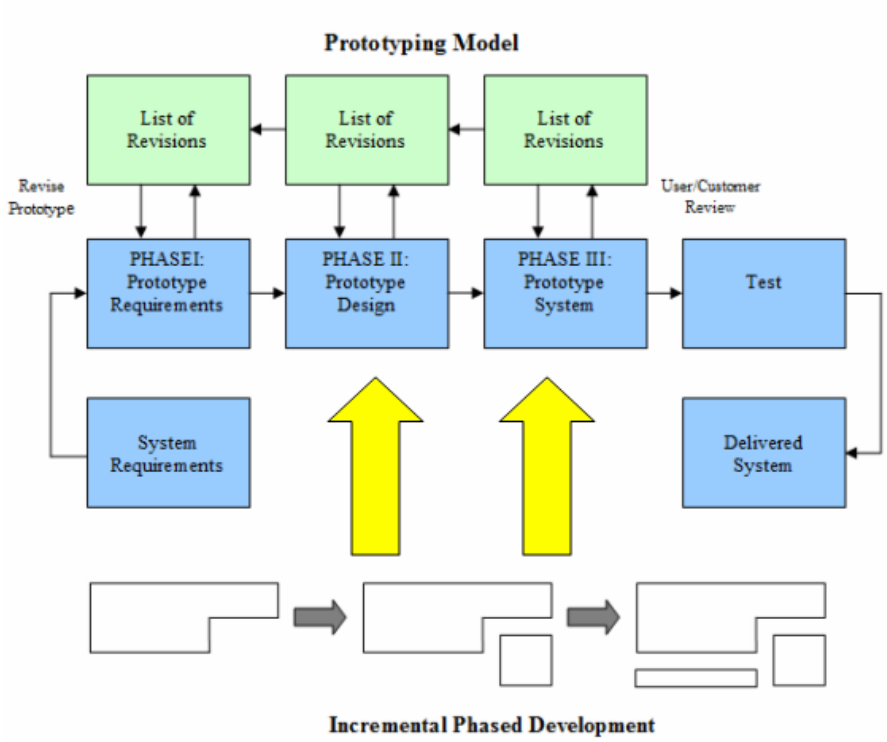
Artifact	**Due Dates**
Meeting Minutes	Same day as meeting
Individual Logs	September 12, 2014, September 19, 2014, ... (weekly)
Group Project Management Reports	September 11, 2014, September 25, 2014, October 9, 2014, ... (bi-weekly)
Concept of Operations	September 11, 2014
Project Plan	September 11, 2014
SRS	September 11, 2014
High-Level Design	October 23, 2014
Detailed Design	October 23, 2014
Test Plan	September 11, 2014
User's Manual	November 25, 2014
Final Test Results	November 25, 2014
Source, Executable, Build	November 25, 2014

Instructions	
Project Legacy	November 25, 2014

Software Life Cycle Process

Our team will follow a hybrid model. This will consist of phased development and the prototyping model. The Online Health Monitoring System lends itself to an incremental phased development approach. The system can roughly be broken into two parts: a scheduler and communications. Communications can further be decomposed into text messaging and video chat. Therefore, these can be developed independently of each other and then integrated into the main scheduler module gradually.

In addition to the incremental phased development approach, the prototyping model adds more structure without being overly rigid. Most team members are relatively inexperienced with administrative duties. Therefore, a flexible model, such as the prototyping model, is necessary. It is anticipated that requirements will undergo refinement and revision during the course of this project. The prototyping model allows for this repeated investigation of requirements and design. In addition, team members are relatively inexperienced in JavaScript. Given this uncertainty, it is important to use a model which reduces risk and uncertainty, hence, the prototyping model. The prototyping model provides a balance of structure and flexibility which suits this team.



Tools and Computing Environment

The computing environment will consist of Windows 7 and 8 operating systems. Tools shall include HTML, JavaScript, SMS for text messaging, MySQL and jQuery libraries. Video chat will consist of a readily available component, such as Skype.

Configuration Management

Configuration management is the task of all team members. Each team member is responsible for version and change control of the artifact he has completed. Artifacts will be placed on the team website and sent to any other team member requesting the artifact. Project management reports will list all artifacts completed.

Quality Assurance

Concerning administrative documents, the Project Manager will review these documents. If the Project Manager is unable to review a document, the responsibility for reviewing that document will be delegated to another available team member. The reviewer will then report the results to the Project Manager. If a document does not meet document standards, the Project Manager will ask the original author to revise the document to meet standards or the document will be assigned to another available team member. Since each major deliverable consists of multiple documents, these documents will be assigned to individuals. The target for each individual to submit his artifact is two weeks before the deliverable due date. This allows for a two-week window in which to check for quality assurance. In the end, all original and revised documents will be sent to the Project Manager and placed on the team website.

Once a code module is completed, the Project Manager will assign another team member to review it according to team coding standards. The results will be reported the Project Manager who will relay the results to the programmer. The programmer will have a maximum of two weeks to make the necessary revisions.

Risk Management

There are three identifiable risks at this point: JavaScript, video chat and text messaging. Many team members are inexperienced in JavaScript. However, Jon has JavaScript experience. He sent team members a tutorial on JavaScript. One of the current action items for every team member is to learn JavaScript. We believe that the video chat component will be the most complex module of code. Luke has researched options for video chat which will be discussed further during the design phase. Text messaging will also require more time than other modules. David has already researched text messaging. This will be an action item in the near future. Although there is low to moderate risk, we are implementing a plan to eliminate risk as early as possible.

Table of Work Packages, Time Estimates, and Assignments

See Appendix A.

PERT Chart

See Appendix B.

Technical Progress Metrics

The system is decomposed into five phases: three documentation phases, group formation phase and a coding phase. The group formation phase is 100% complete. Because the two weeks associated with group formation is considered a substantial portion of the allotted 16 week project cycle, it was considered as Phase 0- 5% of the project. Phase I consists of documentation. Phase I is further decomposed into drafts and revisions. Each template has a certain amount of subdivisions. If a template has ten subdivisions and four of the subdivisions have been completed, 40% of the document will be considered completed. For revisions, if two subdivisions need revision and one is complete, 50% will be considered complete. In addition to documentation for Phases II and III, presentations are a further subdivision. If a presentation is expected to consist of twenty slides and five are completed, the presentation will be considered 25% complete. For the Phase II design stage, the number of diagrams, classes and methods expected will be totaled. The number of these completed divided by the total will yield a percentage. Coding percentages will be based on the amount of classes and methods determined in the design phase. If there are ten methods expected and seven of those methods are complete, 70% will be considered completed.

In summary, percentages are considered the most useful metric. The project has been divided into phases, sub-phases and tasks. Each phase has an estimated percentage weight of time and effort associated with it. Each phase is further decomposed into sub-phases with associated percentage weights. Each sub-phase is further decomposed into tasks with associated percentage weights. In this way, if a task has a 2% weight and an individual's weekly log reports 50% of that task completed, 1% of the project will be considered completed. The bi-weekly Project Management Report will reflect a summary of these reported percentages.

Plan for tracking, control, and reporting of progress

Each team member will post the following information weekly: each action item assigned, time spent on that action item, percentage completed of the action item and cumulative percentage completed of that action item. In addition, each team member will estimate the aforementioned items for the next week. Team members will also report issues encountered and resolutions to these issues. Furthermore, a team member will summarize his overall status in one or two sentences. Initially, the first two individual logs will be emailed to the project manager. Afterward, individual logs will be posted on individual web pages. A reminder email will be sent Wednesday. Replies are expected by 9 a.m. Friday.

The project manager will perform a weekly analysis of individual logs. Weekly analysis consists of reading and analysis of individual logs. Time, percentage completed, total percentage completed will be analyzed. Quality and risks will be assessed and, if necessary, corrective actions will be taken. The weekly analysis deadline will be Monday at 9 a.m.

In addition, the project manager will issue a bi-weekly Project Management Report. The Project Management Report will include one or two sentences summarizing overall status, expenditures, technical progress metrics, an updated PERT chart and one or two sentences of comments. Comments will indicate any planned changes or corrective actions.

Template created by G. Walton (GWalton@mail.ucf.edu) on Aug 30, 1999 and last updated Aug 15, 2000

This page last modified by Chris McCue (christopher.mccue@knights.ucf.edu) on 9/15/14

Online Health Monitoring System

Test Plan

COP4331, Fall, 2014

Modification history:

Version	Date	Who	Comment
v0.0	8/15/00	G. H. Walton	Template
v1.0	9/9/14	C. Chaffin	First Draft
v2.0	9/9/2014	J. Luke	Added Section 4 and major revisions to other sections
v2.1	9/15/14	C. N. McCue	Minor Revisions

Team Name: Project 14

Team Members:

- Jon Carelli - email - web page
- Chris McCue - email - web page
- Chris Chaffin - email - web page
- Ethan Pitts - email - web page
- David Gundler - email - web page
- James Luke - email - web page

Contents of this Document

SECTION 1: Introduction

- Overall Objective for Software Test Activity:

  The objective of the test plan is to identify activities that will help produce an application with quality performance, usability, and functionality, by way of creating test cases and identifying bugs.

  The software test will ensure that our PHP functions are working as documented, our server connectivity is sufficient, our video chat service can support our target load, and that our web pages are displayed properly.

Reference Documents:

- Concept of Operations
- Software Requirement Specification
- Project Management Plan

SECTION 2: Description of Test Environment

We will run our tests on a Virtual Private Server, running a light LAMP stack. This is the most cost efficient and powerful combination for running our application on the web. The application requires at least 512MB of RAM, 1 GHz single core processor, and the ability to connect to the Internet. This test environment would be the same environment that the software would run in after deployment.

Developers will do initial testing, as development progresses users will be asked to test software and provide feedback. If a user finds a bug it will be recorded (create issue ticket with note that this was found by a user and not a team member) and sent to the development team for a solution. At this point, Chris Chaffin will be the tester. This, however, is subject to change.

SECTION 3: Stopping Criteria

The criteria for a stop will be rated according to the importance of the error.

Bugs will be classified on a scale of 1-4, one being the most critical and four representing a problem with a workaround.

- For a bug to be rated 1 there must be a critical error in the application. For example: on launching the application the program crashes before fully loading. Bugs with a rating of 1 will be the most important and of the highest priority.
- A rating of 2 is representative of a bug that needs to be addressed quickly and sometimes crashes the application.
- A rating of 3 is considered to be medium priority, but still needs to be addressed. This type of bug does not crash the system, but prevents some sort of work from being done.
- A rating of 4 is of the lowest priority. These types of bugs are changes that we would like to implement, but are not necessary for the end goal.

If errors are found during testing, the above criteria will be utilized to determine the seriousness of the bug. Bugs of 1 and 2 ratings will require immediate attention. If bugs of ratings 3 or 4 are encountered, we will continue the series of tests as far as possible, recording all bugs that appear. The results of the unit tests will indicate clearly which functions are problematic.

If no errors are found, it will be assumed that the software is working as intended, as long as the unit tests are thorough enough.

If the application is working as documented, meets specifications, and does not contain errors reachable by end users, it will be declared "good enough to deliver".

SECTION 4: Description of Individual Test Cases

- **Test Objective: Create a user from the public web form and activate it**
  - Test Description 1: We will test adding a user to the database by a web form, using the following data:
    - Email: any email reachable by the tester
    - Username: test_user
    - Password: UCF!Test

    Then, open the activation email and click on the activation URL

  - Test Conditions: We will run this test whenever the user signup form changes, and also if any database settings were changed.
  - Expected Results: You will be brought to a page that asks you to look for a confirmation email. An email with an activation URL will be sent to the tester's email address. The user will be added to the database.
- **Test Objective 2: Create and activate a doctor user from the public web form and activate it**
  - Test Description: We will test adding a doctor user to the database by a web form, using the following data:
    - Email: any email reachable by the tester
    - Username: test_doctor
    - Password: UCF!Test

    Then, open the activation email and click on the activation URL

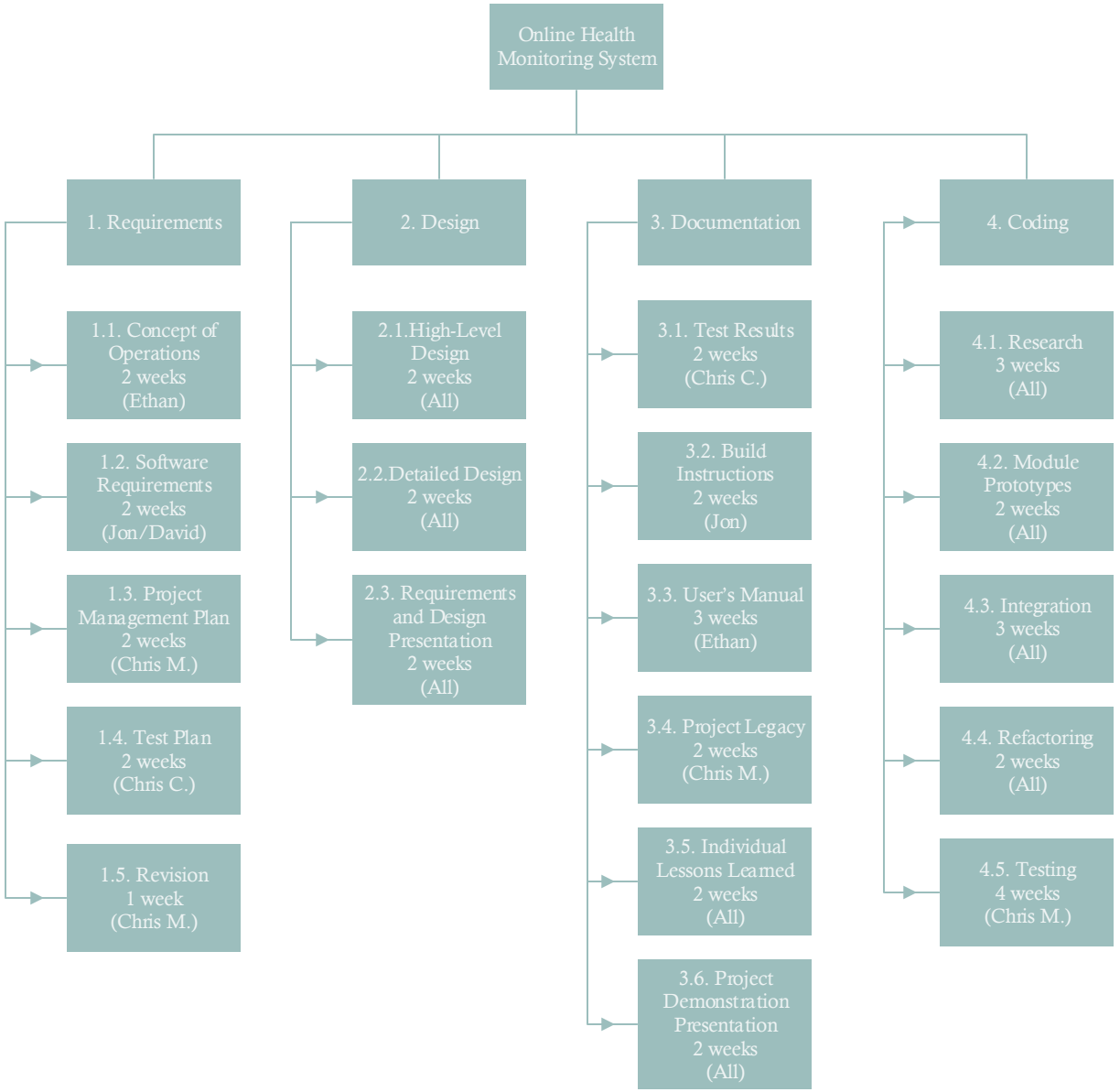
  - Test Conditions: We will run this test whenever the user signup form changes, and also if any database settings were changed.
  - Expected Results: You will be brought to a page that asks you to look for a confirmation email. An email with an activation URL will be sent to the tester's email address. The user will be added to the database.
- **Test Objective 3: User(patient) logs in**
  - Test Description: We will test logging in as a user, using the following credentials:
    - Username: test_user
    - Password: UCF!Test
  - Test Conditions: We will run this test whenever the user login form changes, and also if any database settings were changed.
  - Expected Results: You will be brought to the patient-specific dashboard page
- **Test Objective 4: User(patient) requests a video chat**
  - Test Description: First complete test #3. Navigate to the chat request form. Request a chat from *test_doctor* for the next available time slot.
  - Test Conditions: We will run this test whenever the video chat request form changes.
  - Expected Results: test_doctor will be notified that the time slot has been filled. The timeslot table in the database will be updated.
- **Test Objective 5: Doctor logs in**
  - Test Description: We will test logging in as a doctor, using the following credentials:
    - Username: test_doctor
    - Password: UCF!Test
  - Test Conditions: We will run this test whenever the user login form changes, and also if any database settings were changed.
  - Expected Results: You will be brought to the doctor-specific dashboard page
- **Test Objective 6: Test video chat session**
  - Test Description: Log into the test_user and test_doctor accounts on separate computers. At the time scheduled in test #4, navigate to the video chat page on both machines.
  - Test Conditions: We will run this test whenever the video chat page changes.
  - Expected Results: The two users will be connected in a video chat session at the time scheduled in Objective 4.
- **Test Objective 7: Assigning medicine**
  - Test Description: Log into the test_doctor account. Navigate to the schedule reminder page and select the test_user account. Schedule test_user to take TEST_MEDICINE once daily.
  - Test Conditions: We will run this test whenever the schedule page changes.

    - Expected Results: The reminder will be added to the database, and the test_user account will get a notification to take TEST_MEDICINE.
- **Test Objective 8: Viewing reminders**
    - Test Description: Log into the test_user. Navigate to the reminder calendar.
    - Test Conditions: We will run this test whenever the reminder calendar changes.
    - Expected Results: A calendar should display all the reminders for test_user.
- **Test Objective 9: Manage account**
    - Test Description: After logging into test user, navigate to the manage account page. Enter the following information to update the account.
        - Name: John Doe
        - Phone Number: 555-555-5555
        - Email: any email reachable by the tester
        - Video Chat credentials: a valid username and password to log in to the video chat service
    - Test Conditions: We will run this test whenever the user account system changes.
    - Expected Results: The results of the search should include the test_doctor account.
- **Test Objective 10: Sending a message to the patient**
    - Test Description: Log in as test_doctor. Navigate to the messaging page and select test_user from the drop down list. Enter "Test Message" into the text field and hit the "Send Message" button.
    - Test Conditions: We will run this test whenever the messaging system changes.
    - Expected Results: test_user will see a popup on their page, displaying the test message.
- **Test Objective 11: Logging out of a session**
    - Test Description: After logging into test user, log out of the account. Attempt to access the reminder calendar page. You should get a page that requests a log in.
    - Test Conditions: We will run this test whenever the user account system changes.
    - Expected Results: You should not be able to access any information about the test_user account after logging out.

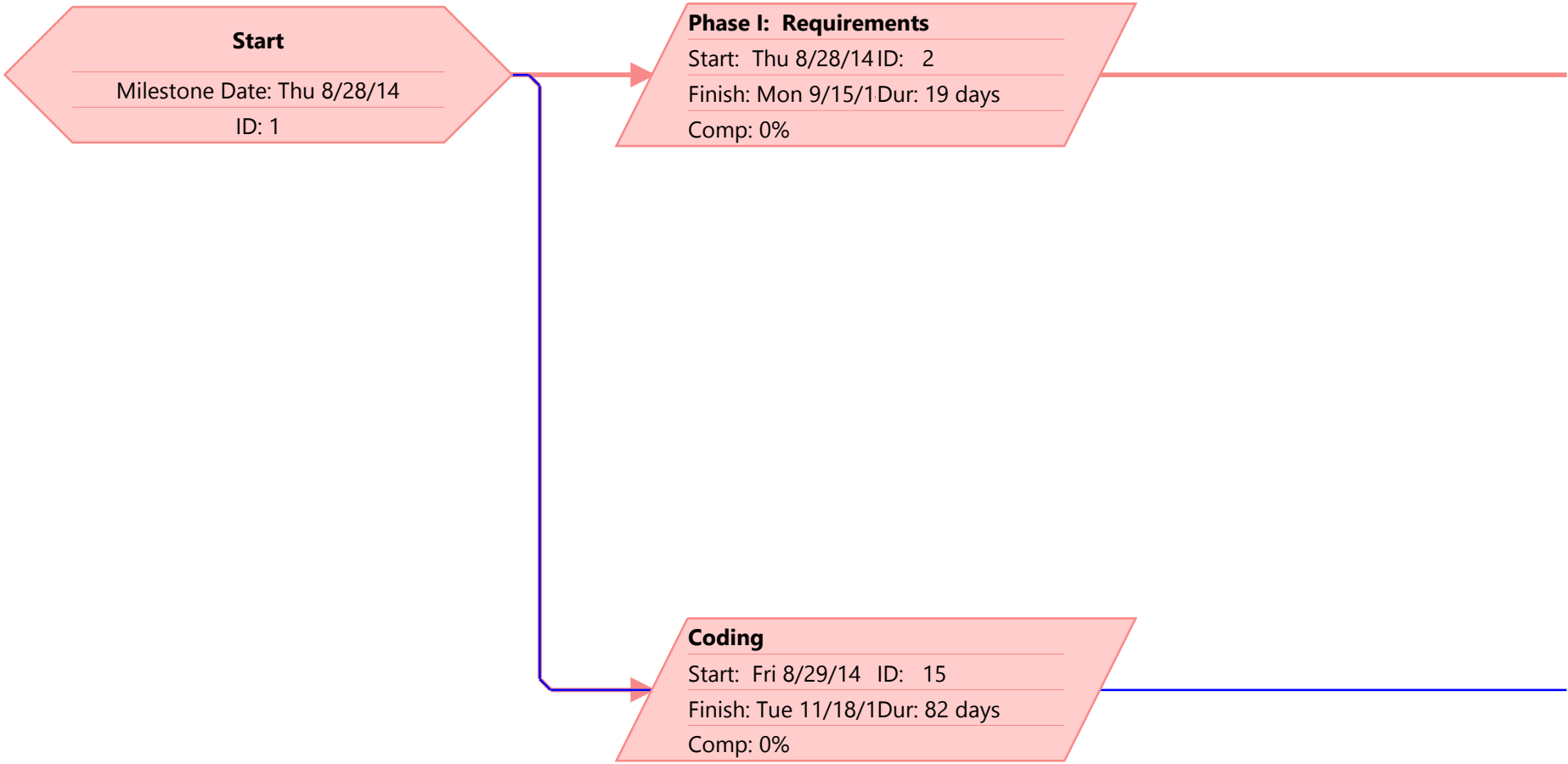
Template created by G. Walton (GWalton@mail.ucf.edu) on March 28, 1999 and last modified on August 15, 2000.

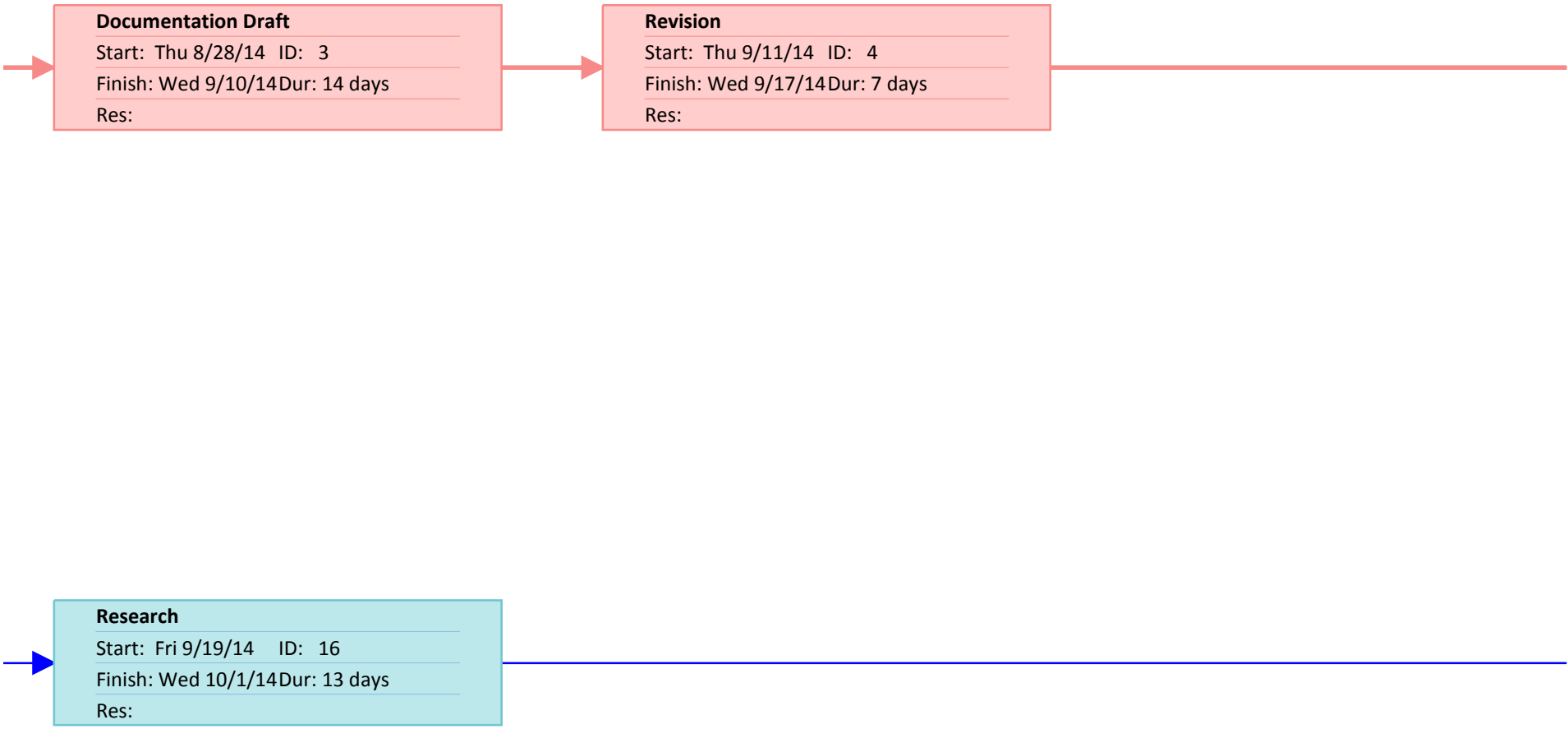
This page last modified by James Luke on 09/09/14

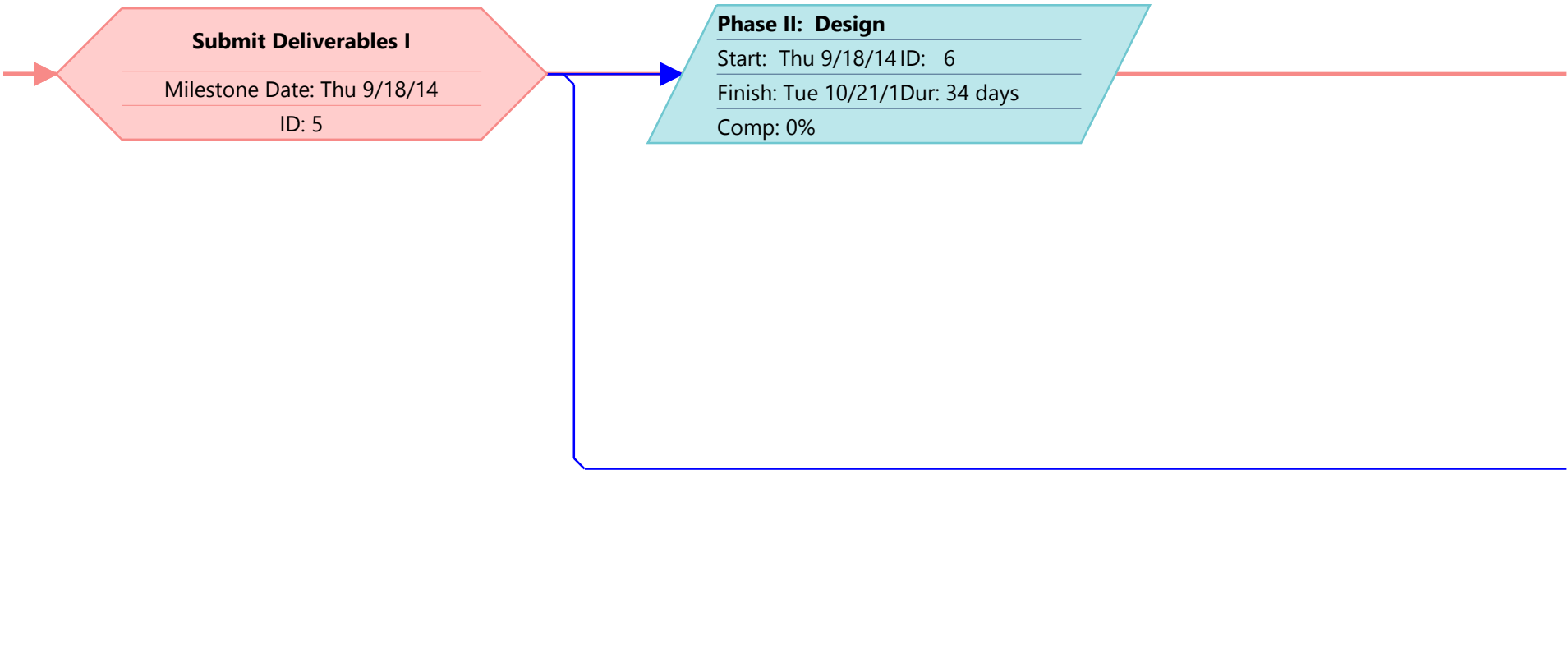
Appendix A

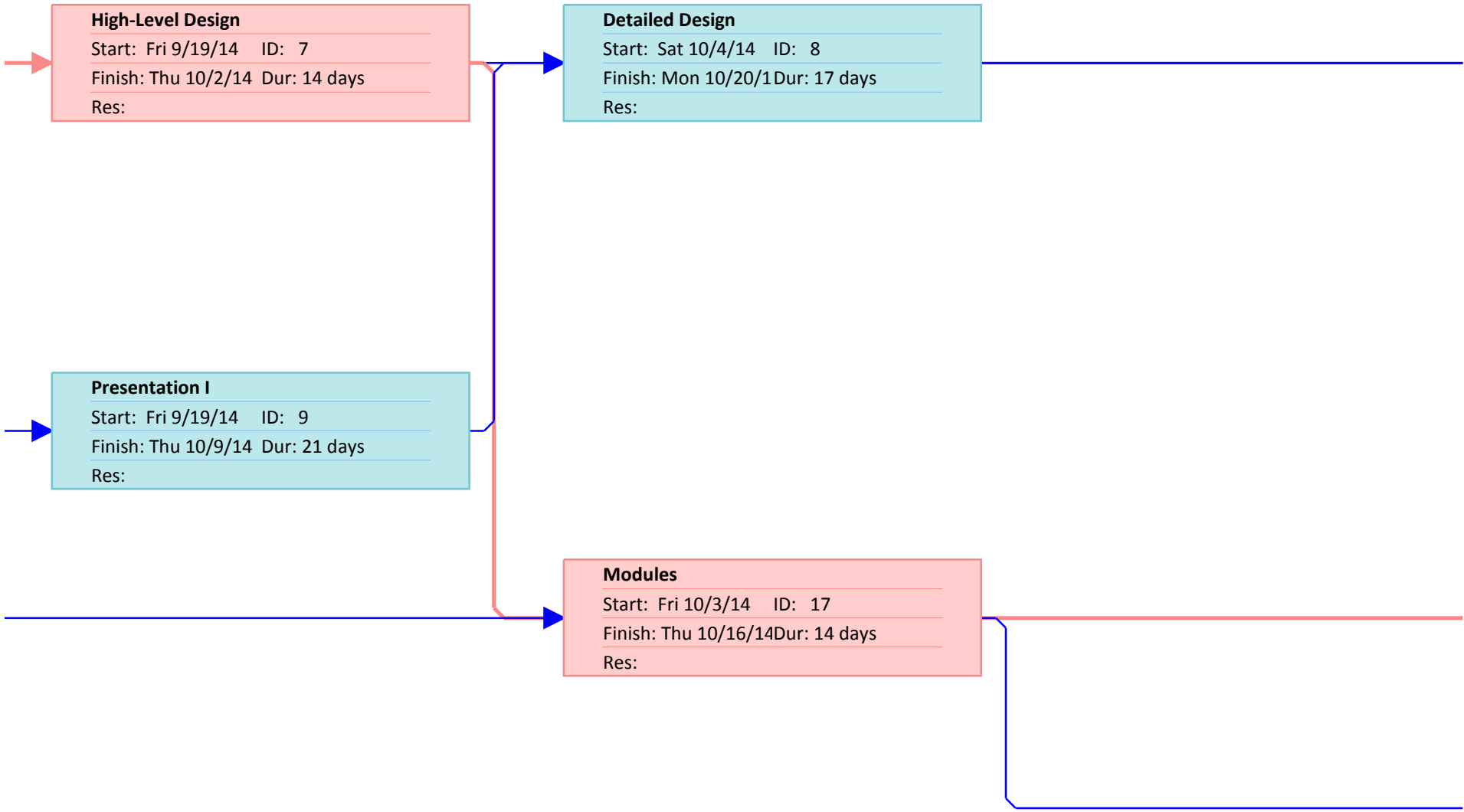

Online Health Monitoring System
1. Requirements
1.1. Concept of Operations 2 weeks (Ethan)
1.2. Software Requirements 2 weeks (Jon/David)
1.3. Project Management Plan 2 weeks (Chris M.)
1.4. Test Plan 2 weeks (Chris C.)
1.5. Revision 1 week (Chris M.)
2. Design
2.1.High-Level Design 2 weeks (All)
2.2.Detailed Design 2 weeks (All)
2.3. Requirements and Design Presentation 2 weeks (All)
3. Documentation
3.1. Test Results 2 weeks (Chris C.)
3.2. Build Instructions 2 weeks (Jon)
3.3. User's Manual 3 weeks (Ethan)
3.4. Project Legacy 2 weeks (Chris M.)
3.5. Individual Lessons Learned 2 weeks (All)
3.6. Project Demonstration Presentation 2 weeks (All)
4. Coding
4.1. Research 3 weeks (All)
4.2. Module Prototypes 2 weeks (All)
4.3. Integration 3 weeks (All)
4.4. Refactoring 2 weeks (All)
4.5. Testing 4 weeks (Chris M.)

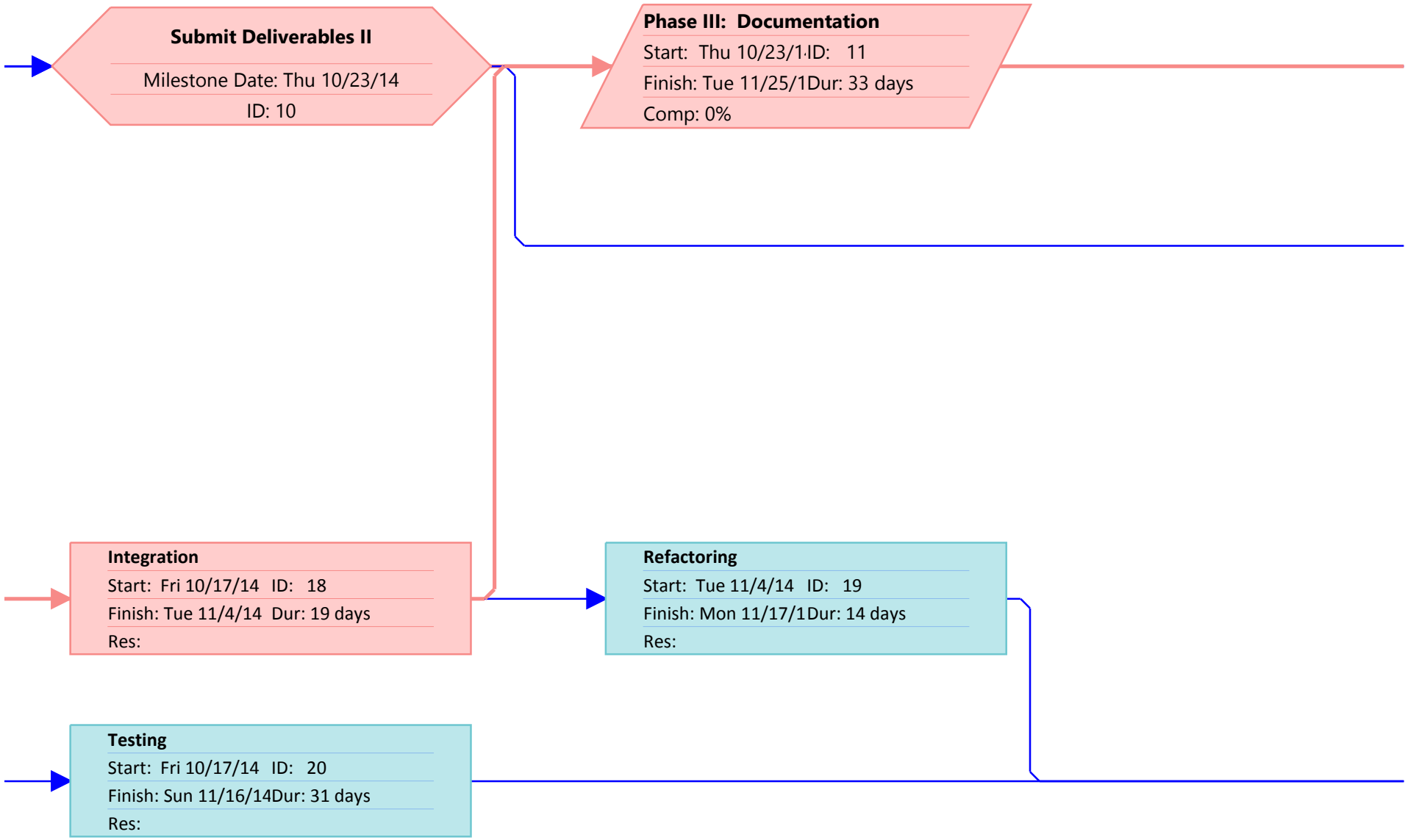
Appendix B

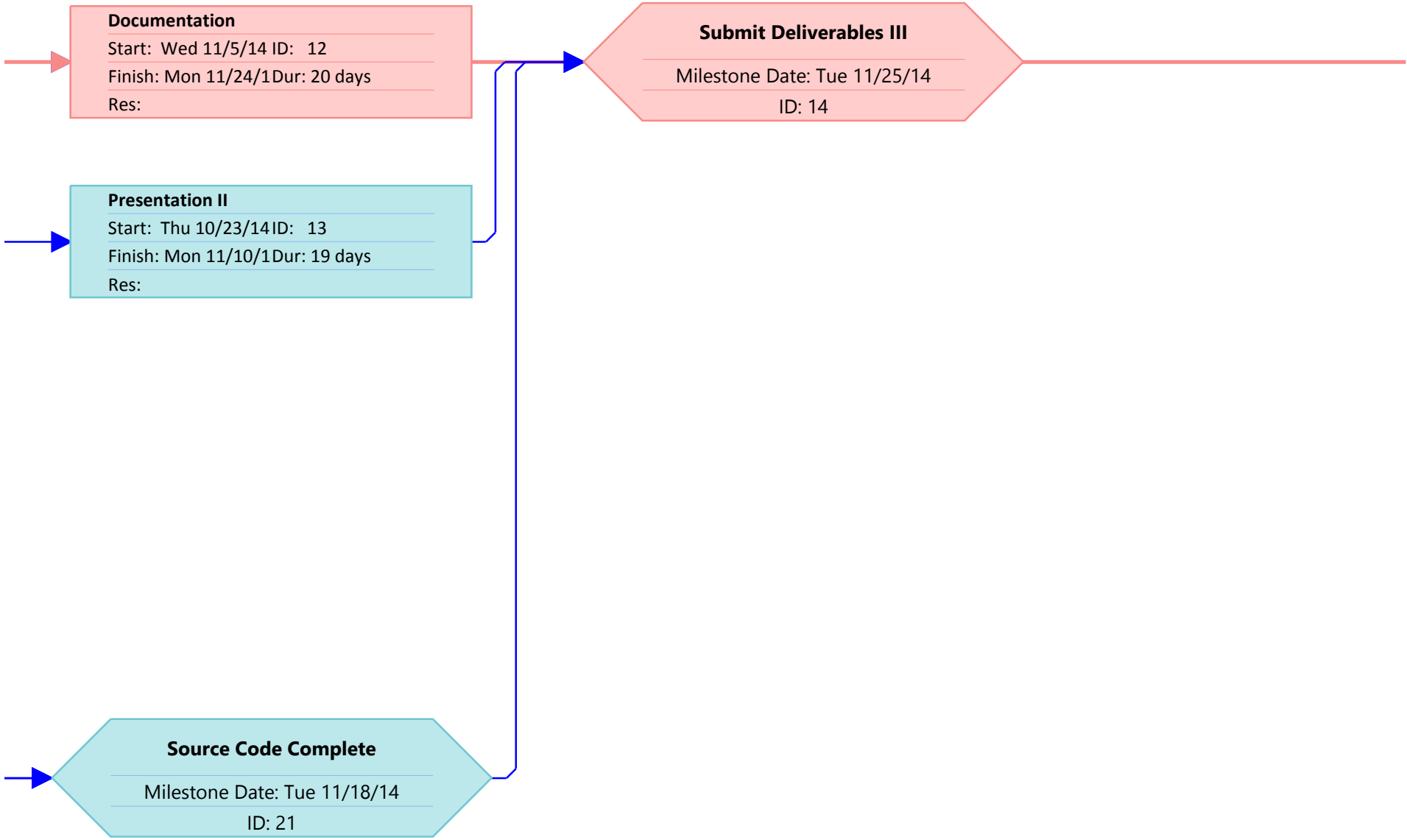

Start
Milestone Date: Thu 8/28/14
ID: 1
Phase I: Requirements
Start: Thu 8/28/14 ID: 2
Finish: Mon 9/15/1 Dur: 19 days
Comp: 0%
Coding
Start: Fri 8/29/14 ID: 15
Finish: Tue 11/18/1 Dur: 82 days
Comp: 0%

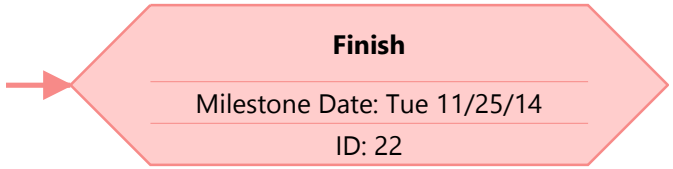

Documentation Draft
Start: Thu 8/28/14 ID: 3
Finish: Wed 9/10/14 Dur: 14 days
Res:
Revision
Start: Thu 9/11/14 ID: 4
Finish: Wed 9/17/14 Dur: 7 days
Res:
Research
Start: Fri 9/19/14 ID: 16
Finish: Wed 10/1/14 Dur: 13 days
Res:


Submit Deliverables I
Milestone Date: Thu 9/18/14
ID: 5
Phase II: Design
Start: Thu 9/18/14 ID: 6
Finish: Tue 10/21/1 Dur: 34 days
Comp: 0%


High-Level Design
Start: Fri 9/19/14 ID: 7
Finish: Thu 10/2/14 Dur: 14 days
Res:
Detailed Design
Start: Sat 10/4/14 ID: 8
Finish: Mon 10/20/1 Dur: 17 days
Res:
Presentation I
Start: Fri 9/19/14 ID: 9
Finish: Thu 10/9/14 Dur: 21 days
Res:
Modules
Start: Fri 10/3/14 ID: 17
Finish: Thu 10/16/14 Dur: 14 days
Res:


Submit Deliverables II
Milestone Date: Thu 10/23/14
ID: 10
Phase III: Documentation
Start: Thu 10/23/1 ID: 11
Finish: Tue 11/25/1 Dur: 33 days
Comp: 0%
Integration
Start: Fri 10/17/14 ID: 18
Finish: Tue 11/4/14 Dur: 19 days
Res:
Refactoring
Start: Tue 11/4/14 ID: 19
Finish: Mon 11/17/1 Dur: 14 days
Res:
Testing
Start: Fri 10/17/14 ID: 20
Finish: Sun 11/16/14 Dur: 31 days
Res:


Documentation
Start: Wed 11/5/14 ID: 12
Finish: Mon 11/24/1 Dur: 20 days
Res:
Submit Deliverables III
Milestone Date: Tue 11/25/14
ID: 14
Presentation II
Start: Thu 10/23/14 ID: 13
Finish: Mon 11/10/1 Dur: 19 days
Res:
Source Code Complete
Milestone Date: Tue 11/18/14
ID: 21


Finish
Milestone Date: Tue 11/25/14
ID: 22

Project: Milestone
Date: Tue 9/9/14