**Hospital**

**Management Project**

Spring Semester 2017

CSC 4350, 5:30-7:15

26 March 2017

**Big Hero 5**

Gautam Ravichandran

Victoria Green

Ashwin Nair

Karankumar Parikh

Young Jun Son

Table of Contents:

Required Tractability Matrix…………………………………………………………………..3 - 6

Work Structure Document……………………………………………………………..……........ 7

Test Cases…………………………………………………………………………………….8 - 11

Test Case Rational…………………………………………………………………………..11 - 12

Constructive Cost Model………………………………………………………………………...13

Gantt Chart.………………………………………………………………………………………14

Dictionary………………………………………………………………………………….…… 15

**Hospital Management System (HMS)**

**HMS- 1.0 INTRODUCTION**

A Hospital Management System (HMS) is designed to automate and organize various day-to-day activities taking place in a hospital. It also stores patient and staff database for quick future access.

**HMS- 2.0 APPOINTMENTS**

**2.0.1** The System shall help in setting up patient appointments by considering each individual case and the schedule of assigned doctor.

**2.0.2** The System shall also give doctors an option to set up or cancel an appointment.

**HMS- 2.1 APPOINTMENT DATA REQUIRED**

**2.1.1** For each patient making an appointment, the system shall request the following information:

* Patient First and Last name
* Patient ID
* Patient Age
* Doctor
* Date
* Time
* Contact Number
* Reason for Visit

**HMS- 3.0 PATIENT RECORDS**

**3.0.1** The System shall maintain patient records efficiently. This information will be manually recorded and inputted into the system after the patient’s visit to the doctor.

**3.0.2** Such records shall be accessed by assigned doctors in the future.

**HMS- 4.0 STAFF RECORDS**

**4.0.1** The System shall maintain a record of doctor appointments, prescriptions given by the doctor and details of the doctor.

**4.0.2** The System shall also maintain relevant information about the nurses and other hospital staff.

**HMS- 4.1 STAFF STATIC DATA REQUIRED**

**4.1.1** For each hospital employee, the system shall request the following information:

* First and Last Name
* Staff Type
* Date of Birth
* Age
* Contact Number
* Gender

**4.1.2** This static data shall be obtained and documented when the staff member is hired.

**HMS- 4.2 STAFF ACTIVE DATA REQUIRED**

**4.2.1** For each hospital employee, the system shall request the following information:

* List of appointments

**4.2.2** This active data shall be consistently updated when changes are made.

**HMS- 5.0 PHARMACY**

**5.0.1** The System shall efficiently maintain medicine inventory and notify when a new order needs to be placed. It keeps track of Medicine ID, medicine name, quantity, and supplier.

**HMS- 5.1 REORDER SUPPLY**

For each item in 6.0, the system immediately reacts to the following situation:

* **5.1.1** When the count of an item reaches <=300, the system shall send a notification to the user with the name of the item and the number on hand.

**HMS- 6.0 INVOICE MANAGEMENT**

**6.0.1** The System shall manage patient invoices and transaction details. It keeps track of patient ID, patient name, contact number, doctor assigned, and total cost.

**HMS- 6.1 INVOICE CREATION**

For each invoice created, the system immediately reacts to the following situation:

* **6.1.1** When the doctor finishes recording procedures and medicine administered to the patient into the system, an invoice shall be generated.
* **6.1.2** The generated invoice shall add up the cost of everything administered at the appointment.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entry #** | **Paragraph #** | **System Specification Text** | **Type** | **Build** |
| 2.0.1 | 2.0 | The System shall help in setting up patient appointments by considering each individual case and the schedule of assigned doctor. | SW | B1 |
| 2.0.2 | 2.0 | The System shall give doctors an option to set up or cancel an appointment. | SW | B1 |
| 2.1.1 | 2.1 | For each patient making an appointment, the system shall request the following information: Patient First and Last name, age, patient ID, doctor, date, time, contact number, reason for visit. | SW | B1 |
| 3.0.1 | 3.0 | The System shall maintain patient records efficiently. | SW | B2 |
| 3.0.2 | 3.0 | Such records shall be accessed by assigned doctors in the future. | SW | B2 |
| 4.0.1 | 4.0 | The System shall maintain a record of doctor appointments, prescriptions given by the doctor and details of the doctor. | SW | B2 |
| 4.0.2 | 4.0 | The System shall maintain relevant information about the nurses and other hospital staff. | SW | B2 |
| 4.1.1 | 4.1 | For each hospital employee, the system shall request the following information: first and last name, gender, date of birth, age, contact number, staff type. | SW | B2 |
| 4.1.2 | 4.1 | This static data shall be obtained and documented when the staff member is hired. | SW | B2 |
| 4.2.1 | 4.2 | For each hospital employee, the system shall request the following information: list of appointments. | SW | B2 |
| 4.2.2 | 4.2 | This active data shall be consistently updated when changes are made. | SW | B2 |
| 5.0.1 | 5.0 | The System shall efficiently maintain medicine inventory and notify when a new order needs to be placed. It keeps track of Medicine ID, medicine name, quantity, and supplier. | SW | B3 |
| 5.1.1 | 5.1 | When the count of an item reaches <=300, the system shall send a notification to the user with the name of the item and the number on hand. | SW | B3 |
| 6.0.1 | 6.0 | The System shall manage patient invoices and transaction details. It keeps track of patient ID, patient name, contact number, doctor assigned, and total cost. | SW | B4 |
| 6.1.1 | 6.1 | When the doctor finishes recording procedures and medicine administered to the patient into the system, an invoice shall be generated. | SW | B4 |
| 6.1.2 | 6.1 | The generated invoice shall add up the cost of everything administered at the appointment. | SW | B4 |

Work Structure Document:

|  |  |
| --- | --- |
| Gautam Ravichandran | **Team Coordinator**  Final Document Handler  Java Coder  Front-end Developer |
| Victoria Green | GUI Coder/tester  Front end tester  RTM Update Manager |
| Ashwin Nair | Java Coder  Front-end Developer  GUI Coder  Final Product Double Checker |
| Karankumar Parikh | Front-end Developer  Database Manager  Setting up Database Backend |
| Young Jun Son | Java Coder  User Guide  Application Tester |

**Test Cases**

**TEST 1**

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Attribute Name | User Login |
| Tester | Ashwin Nair |
| Input | User enters the login credentials – username and login. |
| Action | If the login credentials are accurate, and if the username and password match, then the user will be deemed logged in and will have access to the user interface of the Hospital Management System. |
| Test Type | Acceptance Test, System Test |

**TEST 2**

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Attribute Name | Scheduling an Appointment |
| Tester | Ashwin Nair |
| Input | The user must provide:   * Patient Name (First and Last) * Patient ID * Doctor * Date * Time * Contact (Phone) * Visit Reason |
| Action | The system will process the information and schedule the appointment time according to staff availability. |
| Test Type | Acceptance Test, System Test |

**TEST 3**

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Attribute Name | Medicine Information |
| Tester | Ashwin Nair |
| Input | User selects medicine ID from the dropdown selection. |
| Action | The system returns the medicine’s Name, Supplier, and Quantity |
| Test Type | Acceptance Test, System Test |

**TEST 4**

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Attribute Name | Medicine Low Quantity Alert |
| Tester | Ashwin Nair |
| Input | User selects medicine, |
| Action | Medicine ID selection is rendered, and if the medicine quantity drops below the minimum of 300, the alert box notifies the user that the item is low in quantity |
| Test Type | Acceptance Test, System Test |

**TEST 5**

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Attribute Name | Medicine Reorder |
| Tester | Ashwin Nair |
| Input | User enters the quantity of the medicine needed to be reordered. |
| Action | Medicine Totals are updated in the database upon entering the quantity. |
| Test Type | Acceptance Test, System Test |

**TEST 6**

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Attribute Name | Creating New Staff Information |
| Tester | Ashwin Nair |
| Input | User fills out information for the following criteria:   * First and Last Name * Staff Type * Age * Contact (Phone) * Gender |
| Action | This staff member’s information is added in the database and rendered on log as a new staff member. |
| Test Type | Acceptance Test, System Test |

**TEST 7**

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Attribute Name | Updating Staff Information |
| Tester | Ashwin Nair |
| Input | User updates necessary information from the following criteria:   * First and Last Name * Staff Type * Age * Contact (Phone) * Gender |
| Action | This staff member’s information is updated in the database and rendered on log. |
| Test Type | Acceptance Test, System Test |

**TEST 8**

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Attribute Name | Creating New Patient Information |
| Tester | Ashwin Nair |
| Input | User fills out information for the following criteria regarding the patient:   * First and Last Name * Age * Contact (Phone) * Address * Gender |
| Action | This patient’s information is added in the database and rendered on log as a new patient in the server. |
| Test Type | Acceptance Test, System Test |

**TEST 9**

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| Attribute Name | Updating Patient Information |
| Tester | Ashwin Nair |
| Input | User fills out information for the following criteria regarding the patient:   * First and Last Name * Age * Contact (Phone) * Address * Gender |
| Action | This patient’s information is updated in the database and rendered on log. |
| Test Type | Acceptance Test, System Test |

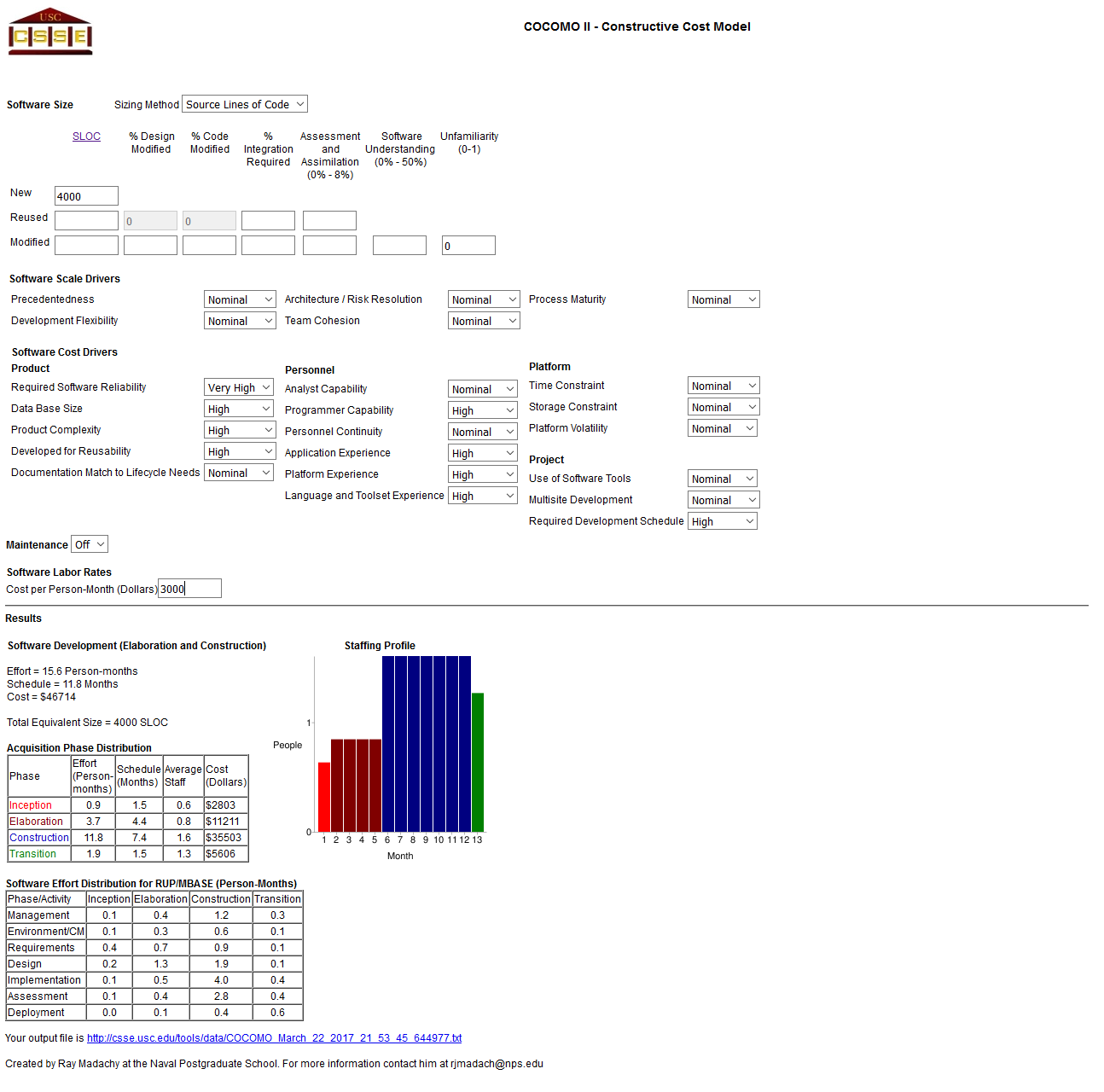
**Rationale**

The tests that were conducted were the following:

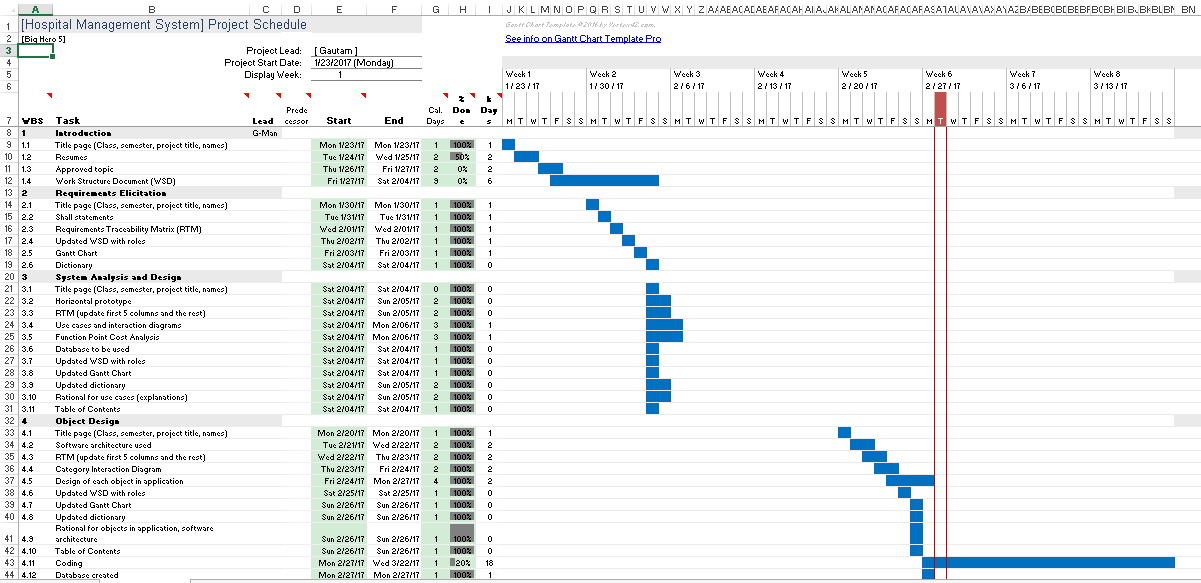
1. User Login - Tests the security of the application and user activity
2. Scheduling Appointments – tests the ability to schedule appointments between patients and hospital staff
3. Medicine Information – Tests to see if the medicine information is rendered from the database
4. Medicine Quantity Alerts – tests to see if the low quantity alerts are rendered if the medicine count drops below 300
5. Reorders/Restocking – tests the ability to reorder the medicine if the medicine quantity is low
6. Creating New Staff – tests the ability to create staff and their functionalities on the system.
7. Updating Staff – tests the ability to edit staff info if in case the staff information needs to be updated or is initially inputted incorrectly.
8. Creating New Patient – Tests the ability to add new patient info into the database to store patient specific information.
9. Updating Patient Info- tests the ability to edit patient info if in case the patient information needs to be updated or is inputted incorrectly.

The intent of the tests being conducted on these components was to test the functionality of subsections and the components they house. In this client-server model, it is vital that the components of the software work aptly. The tests that were conducted on the software were a combination of acceptance tests and System Tests. These tests were conducted to test the individual modals and their functions on an individualized and concise basis (acceptance tests), as well as tests that were conducted to test the Hospital Management System in its entirety (Integration/System Testing). The tests were conducted for these components, as the Hospital Management System incorporates multiple sub-systems in the form of Pharmaceutical Inventory, Staff and Patient Info Housing, and Scheduling Systems. Thus, by the testing frameworks incorporated, the system is deduced to having rendered the general actions necessary for the software functionality as stated in the test cases.

Constructive Cost Model:



Gantt Chart:



Dictionary:

Static:​ This term defines fixed values. These are values that are set to equal a certain value throughout its iteration. This includes the relevant hospital employee information, such as: name, gender, etc. are static as they are fixed values depending on the individual employee information. This also pertains to the same relevant descriptive details of the patients.

Database: ​This is a data storing system which will allow the software to access content of the system later by those including: doctors, nurses, and front desk employees.

GUI:​ (Graphic User Interface) This is the interface in which users will be able to access the content of the Hospital Management System. The GUI contains the UX/UI aspects of the program such as the window, buttons, search bars, tabs, etc.

Java:​ The programming language which is used for the software’s implementation and design. This programming language will address the contents of the Hospital

Management System and the layout of the system methodologies.

Active Data:​ The data that is dynamic or interchangeable within the system. This is the system information pertaining to the employees and the general hospital supply/accommodation and the changes in their data (scheduling/inventory/room availability)

SQLite:​ Database structuring/management tool which stores content related to the user. This user information includes but is not limited to: employee information, patient information, inventory/supply count, general hospital facilities.

UX/UI:​ (User Experience/User Interface) This relates to the content on the front-end side of the software and its accessibility/use or general functionality of the content when used by people for testing purposes or public use.

Use Case Diagram:​ Shows an overview of the system and functions.

Methods:​ These are functions which can be called throughout the program. Sometimes the code for how methods work is hidden from view in order to make code easier to read. These functions complete specific tasks, such as sorting, calculating, and adding/deleting.

Class:​ A class is a blueprint for objects created in a program. These classes will be made to contain organized code to control things like patient data, inventory, and patient accounts.

Object:​ An object is what a class controls. There can be a patient object, which will have states and behaviors such as name, date of birth, and wellness levels.

Horizontal Prototype:​ Shows the broad relationships between a system and maps out its range of abilities.