**Design Document for CMPUT 291 MINI PROJECT 1**

Group c291g21

Name: Rui Wu

CCID: rwu4

Name: Mark Kamprath

CCID: mkamprat

**1. General Overview**

On launch, users are able to login, register or exit the program. If login, user needs to enter his/her login and password. If he/she successfully logs in to the system, he/she will see a menu based on his/her role.

For a doctor, he/she able to search a patient based on name. The system will display a list of patients with a similar name, the doctor needs to select a name based on an index. Once the patient is selected, the doctor is able to see a list of charts related to the patient, each chart will indicate whether it is open/closed. If the chart is closed, the doctor is able to view the symptoms, diagnoses and medication related to that chart. If the chart is open, doctor will see options to add a symptom, diagnosis or medication.

For nurse, he/she will be able to add a patient to the database. He/she can also select a patient, and see a list of charts related to the patient, each chart will indicate whether it was open/closed. If all the charts are closed, he/she has the option to create a new chart. If there is an open chart, the nurse is able to add a new symptom, (observe date will be auto-generated) and/or the nurse is also able to close the chart.

**2.Detailed Design**

main.py

* connectToDB() - will open the database, and then pass the connection to below
* init() initial menu - This method will display the initial menu, where user can either login, register or exit
* login() login screen - This method will take in a login ID, and find a match. If there is match, prompt user for password, if the password also matched, proceed to menu based on his/her role. If either a login ID and password not match, return to initial menu
* register() register screen - Ask user for his/her name, role, staff\_id, login id and password and try to insert to the database. If there is a conflict, prompt the user to enter again

mainMenu.py

* doctorMenu() doctor menu - This method allows the doctor to select a patient or log out.
* nurseMenu() nurse menu - This menu allows the nurse to select a patient, add a patient, and/or log out
* adminMenu() admin menu – This menu allows the creation of reports based on supplied information and pre-determined queries.

patient.py

* patientSelect() select patient screen - Once the doctor wants to select a patient, the doctor enters a name and finds any patient name that is similar to the input. The doctor needs to specify the patient.
* patientChart() select patient chart - If the doctor/nurse selects a patient, the system will populate a list of charts that is related to the patient. This indicates whether the chart is open or closed. The doctor is able to select any chart to view. If all charts are closed, a nurse has the option to create a new chart.
* ViewChart() view patient chart - When the doctor selected a chart, the system will generate symptoms, diagnoses and medication related to the selected chart.
* doctorChartMenu() - if the chart is still open, then doctorChartMenu will appear, so the doctor can add an entry to symptoms, diagnoses and medications table.
* nurseChartMenu() - if the chart is still open , the nurseChartMenu will appear, where the nurse can add an entry to symptoms and/or close the chart.
* addSymptoms - add symptoms. For an open chart, staff can add a new observed symptom to the database. Observation date will be auto-generated.
* addDiagnosis() - add diagnosis. For an opened chart, staff can add a new observed diagnosis to the database, observation date will be auto-generated.
* addMedication() - add Medication. For an open chart, doctor can add a new medication to the database. The system will check whether the drug-name appears on the database, and it will check whether the dosage amount is larger than suggested amount, then it will check whether the patient is allergic to the particular drug or not
* closeChart() - Close the current chart.
* addPatient() - add a new patient to the system, requires(name, hcno, age group, address, phone, emgerency phone number).
* addChart() - add a new chart to the system. Auto generates chart id.

reports.py

- doctorReport() - Report outlining the amount and name of each drug a doctor has prescribed within a time period.

- rxTotal() - Report outlining the total amount prescribed for a drug in a given category within a time period.

- allRX ()- Report outlining all possible medications that have been prescribed over time after a given diagnosis.

- diagBeforeRX() - Report outlining all diagnoses made prior to prescribing a specific drug.

**3.Testing Strategy**

* Use the test file provide by one of the TA.
* 1. Ensure the functionality is working as expected with correct inputs.
* 2. Test incorrect inputs such as invalid hcno, invalid menu entry.
* 3. Try to insert a patient hcno already existed in the database.
* 4. Try to run multiple paths to see if any errors occur.

**4.Group work breakdown**

*First meeting*: Divide work amongst group members.

Rui is responsible for the UI and doctor tasks, Mark is responsible for nurse and administrative staff. Since nurse had two queries that are similar to two of the queries in the doctor tasks; Rui had the UI and 4 queries, Mark had 6 queries. The project is equally weighted for both team members.

Rui:

I am responsible to implement the command-line interface, such as login, registers, menus. He also work on the doctor part, where he implement the search query, search charts, select charts, view charts, add symptoms, diagnosis and medications. I spend about 5 to 5.5 hrs. Spend about 2.5 hrs to implement the use interface, 2 hrs to implement the functionalities for doctors. The last 1 hr to implement password encryption, trying to simplify the query (want to get as little data as possible) and some refactoring on the code.

Mark:

I was responsible for the tasks associated with nurses and administrative staff.

As I am not as quick, or talented as Rui in regards to programming – it took me much longer to code my portions. Roughly 10 hours was spent on the Nurse task, implementing the add patient and add chart portions. I, very gratefully, received advice from Rui a few times on areas where I performed sloppily, or he had to make slight modifications to the nurse menu so that it would work with the doctor tasks. This likely added a couple hours or more to Rui’s work load.

*Second meeting*: Explained how we executed our individual portions so that we could proceed without breaking each other’s code; we then combined the code we had up to that point.

Rui :

I spend 1 hr to double check on Mark’s code. Ran couple tests on it, fix few errors. Rui modify the Nurse menu, the closeChart(), addChart(), addPatient() and add try-catch inside the code to detect any un-nice inputs.

I spend another 0.5 hrs to run test between doctor and nurses. Go thru most of the paths to check for bugs.

Mark:

Another 12 hours or so was spent on the administrative task portion, roughly 3 hours was spent on each task.

Documentation/commenting code took another 20 minutes or so.

**5. Methods of communication**:

a) Text messaging.

b) Email.

c) In person-meetings – Either prior to class, or one of the engineer buildings when we were both at the university.