# CMPUT 291 – Mini Project 1 Design Document

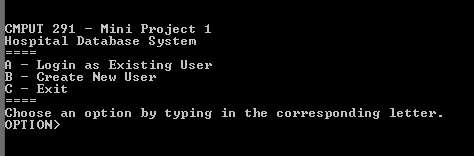
Shouyang (Dan) Zhou (1410390), Christopher Saunders, Jordan Vogel

## Overview

This project is a miniature database driven hospital IT system. It allows hospital and administrative staff to store and retrieve, and modify information related to their duties by their role.

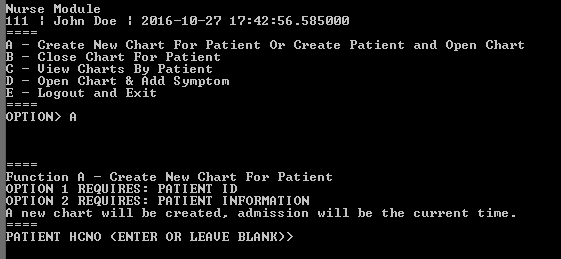
**User Guide**

This application allows the user to access database functions within the scope of their role. The application is two parts, a login menu and a role dependent menu. The login is as follows:



The general structure of the menus is the same. There is a title, a choices section, and a desired input (generally in capital letters followed by a “>” icon). A choice between items will generally follow a letter such as in the above.

Some input sections will not display your input; this is meant to hide sensitive information from others.



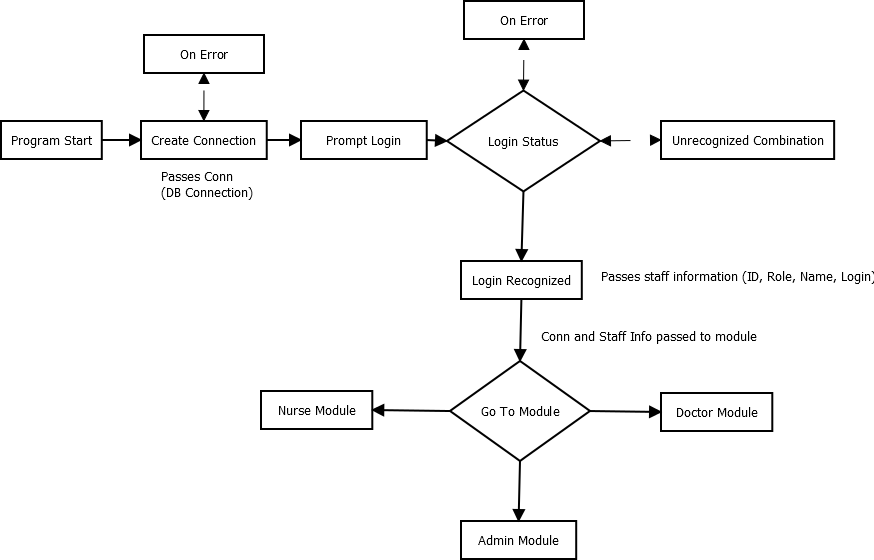
The above illustrates a role dependent menu and one of the functions present. Note that a history is preserved such that information and inputs displayed. There are generally five menu options present. The input flow of each depends upon its nature, just follow the prompts given.

The database will update immediately once an application function is completed barring any errors or invalid inputs.

Some things will require that the database administrator interacts directly with the database, for example the deletion of patients.

## Design Strategy

The following illustrates the program flow:



**Each module** will contain a main function that offers the following:

1. Print the menu options.
   1. On module initiation, unrecognized command and the completion of a task (on success, or exit), menu options should be presented. They are displayed in the following format:

|  |
| --- |
| [Module Name]  [Staff ID] | [Staff Name] | [Current Date]  ====  A – [1 Line Description]  B – [1 Line Description]  C – [1 Line Description]  D – [1 Line Description]  E – [Logout]  ==== |

1. Do an assigned task (Tasks A-E)
   1. For standardization, each function is assigned a letter.
   2. Navigation will use these letters are triggers.
   3. Each function will take the following format:
      1. (As examples): NUR\_A(conn), DOC\_B(conn), ADM\_C(conn)
      2. Conn is the connection made during the main module.
      3. Additional parameters may be passed after conn
   4. Note that logging out will close the connection object, conn and exit the application

**Each function** should be designed in the following format:

## Testing Strategy

Each person tests another person’s function / module. Some functions are either trivial or the same across modules, these are simply tested by whoever.

## Workload Allocation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Task** | | **Programmer** | **Tester** | **Notes** |
| **Main Program** | |  |  |  |
|  | Connection | Dan | All |  |
|  | Log In | Dan | All |  |
|  | Encryption | Dan | All |  |
| **Doctors Module** | |  |  |  |
|  | Landing | Dan | Jordan |  |
|  | A | Jordan | Dan |  |
|  | B | Jordan | Dan |  |
|  | C | Jordan | Dan |  |
|  | D | Jordan | Dan |  |
|  | E (Same Across) | Dan | Dan |  |
| **Nurse Module** | |  |  |  |
|  | Landing | Dan | Chris | See Module for generic functions that can be reused. |
|  | A | Dan | Chris |  |
|  | B | Dan | Chris |  |
|  | C (A in Doctors) | Jordan | Chris |  |
|  | D (B in Doctors) | Jordan | Chris |  |
|  | E | Dan | Chris |  |
| **Admin Module** | |  |  |  |
|  | Landing | Dan | Jordan |  |
|  | A | Chris | Jordan |  |
|  | B | Chris | Jordan |  |
|  | C | Chris | Jordan |  |
|  | D | Chris | Jordan |  |
|  | E | Dan | Jordan |  |