**ECE 356 – Project – Second Deliverable**

**1. Outline (briefly) any changes to your first deliverable document - if any.**

Everything.

**2. Briefly describe the GUI for your system using a user-interface flow chart.**

This application was going to be very difficult to design a GUI for since there were so many different user types that would be accessing the system such as finance, general staff, legal, doctors, patients, and admin. Each user type needs to have access to a different set of pages. However, a number of the pages overlap between users, but have slightly different interfaces.

For example, create patient information page is a page where you can enter a new patient into the system. However, we can repurpose this page to function differently by allowing it to create, view, and update patient information. Each of these different functionalities will be referred to as “modes”. As a result, there are a number of pages can be used in different modes. This design will save us development work since we don’t have to repeat code between similar pages with similar functionality.

We used this model as the core design of our project. We designed an application where when a user logs in it displays a set of tabs where each tab corresponds to a different page. However, the system needs to know which pages each user should see and what mode they should be in. The application determines what type of user is logged in and displays the appropriate pages in the proper modes based on the user type.

The application flow will be broken down based on the user type and what pages they will see.

**Login Page**

User enters login information

**Main Page**

User

**3. Briefly describe your system's software qualities with respect to portability and concurrency.**

The software system that we have designed is made to be very portable and handle concurrency well.

The software system is very portable and could be easily be deployed in a hospital environment without very much difficultly. The application itself is a designed using Java which works is compatible all modern day operating systems that have Java installed. The database would be slightly more complicated to setup if it were designed to be used by multiple people. The database would have to be deployed on a server that can be accessed by all the computers in the hospital. Each installation of the software on the system would have a configuration file in which you can set the database connection string to target the server where the database is deployed. This would likely be configured during the installation of the application.

Concurrency is a complicated problem since there potentially can be so many people accessing the system at a time. For example, there may be multiple staff booking appointments for doctors at a time and the appointment could overlap causing a problem in the scheduling system. Alternatively, a staff member could delete an appointment record that a doctor was trying to input a visitation record for and result in the visitation record not being entered into the system since there isn’t a corresponding appointment anymore which is a database requirement.

Our method of achieving concurrency is achieved by adding locks around database queries that could affect concurrency. Putting a lock on a table while you are making modifications to the data prevents other users from accessing the table until the query is complete. This would normally be a problem since you are locking people out from accessing the database, but since the queries are so fast, typically around 3ms, it shouldn’t cause any problems for other users. Furthermore, we can add triggers to tables that test inserts into tables to verify that the data is legal and doesn’t violate any of the table rules (such as booking a doctor with two appointments at the same time).

**4. How did you test your system - any tools used, test drivers, etc.? Short one paragraph.**

In order to test that our database had been created correctly with the proper primary and foreign key references, proper data types, and field lengths in each table we needed a set of dummy data. This dummy data could be used to verify the database is created properly. However, it would have to be created within the constraints of the database taken into account and have matching foreign keys to maintain the database’s integrity.

This dummy data can also be used for further development and testing. When the application is being designed, it is much easier to test it when there is a set of data already in the system to ensure proper display and formatting of the data.

When it comes to the application, the testing will need to be done by hand to ensure everything is functioning properly. Therefore, we need to test each individual page for all the different error scenarios and make sure that the application handles it gracefully. These errors include problems such as entering invalid date formats, improperly formatted data, required fields that are empty, and double booking appointments for doctors. Each page must test for all of these error scenarios and have a way of handling each situation with an informative error message for the user. Testing typically takes nearly as much time as the development does because there are often so may edge case scenarios to test.

**5. Highlight any part of your overall project that you deem to be significant and perhaps novel.**

The fact that we lied and used C# instead. A novel idea.