My contribution to the project included the design and creation of the database, design and creation of unit tests for the database functions, and creation of the back-end code to drive the application. I was able to improve my skills in database creation through working on this project. In the first semester, I designed a database schema that would allow us to store all the information we deemed necessary for the application. Working with our advisor, this design went through multiple iterations focusing on aspects like minimizing redundant data and the total number of tables. This resulted in a schema that was more lightweight and also required less work to integrate into the application. I also gained a better understanding of how an application can communicate with a database. I was able to write functions that would generate strings in the form of a command that can be used by the database. Then I can send these string to the database to perform any function I need, mainly the four CRUD operations. This experience also taught me a few things about software management that I didn't encounter in previous projects. For one, I learned how to create a gitignore file to avoid pushing autogenerated files with every commit. I also learned how to publish a piece of software to be used externally from the environment it was coded in.

The biggest success I experienced while working on this project was deciding to pull in all of the data from the database to store locally on startup of the MediStore Manager. Certain functions would require the data from multiple tables. For example, Creating a work order to deliver a bed to a patient requires us to get the ID of the patient, create the work order, and update the assigned person of the bed. Each of these would translate into a different call to the database to get the required information and another to update it. By pulling the data in at startup, we're able to easily access data from any table at the cost of some initial load time. This reduces the number of times we access the database and the risk of conflicts if multiple users are active at the same time. The biggest obstacle I faced while working on this project was unforeseen changes to the database. At multiple points during the project, it became clear that a change to the database structure was necessary to perform the actions we wanted from the application. Each time, this would require a teardown and reconstruction of the database, followed by modifying any of the functions written to interact with the changed tables to account for the differences. This added up to a decent amount of work that could have been avoided with a better understanding of all the features we wanted in the application during the database design phase.