**Project Summary**:

Professional Inventory Manager, or PIM, is a software application that serves to assist commercial entities with organizing the storage of both their digital and physical products. PIM allows business owners and inventory associates to keep track of their merchandise in an efficient manner. Users are able to locate items and item information, modify this information, and dynamically change the quantity of items from the inventory spaces. Each item stores information such as: a commercial name, a unique identification number, quantity, pricing, and shipping date. Additionally, users can break down their entire inventory into a series of inventory spaces to allow for further modularity and better management of warehouses and backrooms. While this project is tailored more towards larger retail companies, other storage facilities, warehouses, and locally owned businesses will find our software to be useful.

// possible paragraph about functionality

// possible paragraph about user interface design

Along with creating a program that was easy for the user to maneuver through, our team wanted to also ensure that the proper safety precautions were taken to maximize security. For example, our application implements varying levels of authorization that a user can hold. If the user is considered to be an ‘employee’, they can modify item information such as an employee at a warehouse. If the user is considered to be an ‘administrator’, they can modify item information similar to the employee, as well as modify account information of other users. This allows only specified individuals to be able to control other employees, such as one would with a managerial position. Additionally, we implemented hashing with our credential storage system. When the user enters their login information, this information is hashed using the well-known %HASHNAME% algorithm before storing and/or comparing with other values. This prevents plaintext passwords from being stored at any point of the sign in process, which can be a large security threat if somehow an individual were to access the local database itself.

After completing this project, we found that there were many aspects we did well on. For example, our high and low-level design documents proved to be very beneficial for what would become the development stage of the software engineering process. Because we took the time to thoroughly plan out our project with the initial requirements, the coding process itself was greatly simplified. Additionally, our ability to adapt the requirements to the programming language that we used was also done very efficiently. There were a couple of times when someone had a question on how to implement a particular feature. First, we would discuss amongst ourselves to find a possible solution. If no one in the group had experience with that particular topic, we would branch out and perform research online in order to find an adequate solution. This proved to be very effective in allowing us to implement most of, if not all of our original requirements.

Reflecting on what we’ve done with this project, there are some things that we could improve upon in the future. One thing that should have been done differently was the distribution of work. At the beginning of the project, many of our documents were set up using Google Docs, Google Sheets, etc. which allow live collaboration. With these documents our team members worked almost independently, each contributing what they could when they felt they had time (within the constraints of the due date). However, this led to the contributions being unequal. This issue could be solved in the future by setting out portions of each document for each team member to complete, to attempt to keep work distributed as equal as possible. Another thing we could do better in the future could be the creation of a current, to-date checklist that could be used to maintain what requirements we need, what has been completed, and what has been retconned. Though we did have good communication and knowledge of what to do, it felt like we could have had an easier time if we used a single document to check on rather than going through all the older documents we had compiled over the creation of PIM.

Besides things to improve upon, there were also obstacles and issues we had to work through during the creation of PIM. One problem we ran into while working on PIM was trying to find out how to display data in the dataGridView sections of the Windows Form. The dataGridView has an option for data source selection, yet this did not allow us to select our SQL tables we had created. We tried making a .NET version, but this did not solve the problem either. Eventually, we discovered that the dataGridView sections can be coded to show the SQL data information which solved our problem. This discovery also led to us being able to code the rest of the Main Form, as it uses quite a few of the same code blocks to display data in other areas.