**Thomas Martin**

**CS-360-16228-M01 Mobile Architect & Programming**

**3-3 Submit Project One**

**Southern New Hampshire University**

**September 20, 2024**

The goal of this project is to develop a mobile app that will allow my daughter, Courtney, to efficiently manage her company, "Donkey Fit Apparel," by monitoring and controlling the inventory of her apparel line. The app will track stock levels, monitor sales trends, and send notifications when items are out of stock. By streamlining the inventory management process, the app will enable Courtney to make data-driven decisions regarding reordering and optimizing sales. Additionally, it will meet all warehouse management requirements, ensuring the app is reliable and easy for users to navigate.

Key components of the app include a database, a login system, an inventory grid, inventory management features, and stock notifications. The database will consist of at least two tables: one for storing inventory items and another for managing user logins and passwords. Users will be able to log in securely or create an account if they are new to the app. Once logged in, they will be taken to an inventory grid that displays product names, images, categories, and stock levels. From this screen, users can add or remove items, adjust quantities, and receive alerts when stock levels hit zero.

The primary users of this app will be Courtney and authorized staff members who assist with inventory management. Two distinct user types can be identified: administrators, like Courtney, and warehouse or inventory staff. As the business owner, Courtney will be the primary user. Her main goals are to maintain an accurate record of all inventory items, track which products are selling, and know when items are out of stock or need reordering. She will need a straightforward interface that provides real-time updates on inventory levels and sends notifications when stock is low or depleted.

The second type of user will be warehouse or inventory staff. Their primary goals are to update stock levels, add new products, and ensure the physical inventory matches the system records. Their use of the app will focus on making quick updates, such as increasing or decreasing stock levels when items are received or sold.

The app’s design will prioritize a user-centered UI, ensuring Courtney and her staff can easily navigate and perform tasks with minimal effort. The user experience will emphasize simplicity, clarity, and efficiency, following Android’s Design and Quality Guidelines to ensure consistency, responsiveness, and intuitive use.

The necessary screens will include a login/signup screen, an inventory dashboard (grid screen), a product details screen, and a notifications feature. The product details screen will provide comprehensive control for managing specific items, including CRUD (Create, Read, Update, Delete) functionality.

Key components of the UI will include username and password fields, login and signup buttons, an inventory grid displaying products (with product name, image, and stock levels), and buttons to add or remove stock. There will also be filters for sorting products, fields for entering product details (such as name, category, size, and stock quantity), buttons to update or delete products, and input fields for adding new items. The app will also feature alerts or pop-up notifications to inform users when stock levels are low or out.

According to Android Design and Quality Guidelines, clear and actionable layouts, consistent navigation, and efficient user interactions are essential. By ensuring the app’s navigation is intuitive and the screens are uncluttered, users will be able to manage inventory efficiently and effectively.

The app’s functional requirements—user authentication, inventory management, stock notifications, and CRUD operations—will be represented in the code design using back-end API calls and UI logic. Data flow between the screens and the database will be handled through API calls, with the app acting as the client and the server managing database operations like POST, GET, PUT, and DELETE requests. Each screen will trigger these calls based on user interactions, ensuring the app remains responsive and the displayed data is always up to date. By maintaining a clear separation between the UI and back-end logic, the app will deliver a smooth user experience while securely handling and processing all data in the background.