Requirements Document

Introduction: This project came from a conversation with several small business owners and startups about the lack of affordable inventory management systems. We are setting out to develop a simple system that startups can use to track what they do and don’t have in stock and store that in a database. The initial system will be created for Carolina Air, an air cleaner reseller. The information should be readily and easily available for the users.

User requirements Definition:  This will use a simple interface that takes minimal training, so that those without a lot of technical background can still use it efficiently. This system may be compatible with PoS systems. It will have a database to input inventory and remove inventory. It will also have an alert when products get under a certain amount to reorder. It may use UPC to help manage inventory stock.

System Architecture:  Our system will have three potential services. It may use a mobile device, a cloud service, or a desktop. The three hardware components that are intended are a mobile device camera, Pi camera, or a barcode scanner that the systems will implement. It may implement some form of an API to scan, read, and create the UPC. The main program will be programmed in C# using Visual Studio and we may need modules from other languages to wire the devices to the program. It will include an add function to add inventory to the database. This will be accomplished manually or may be accomplished with scanners. The program will include alerts if inventory drops below a certain point defined by the user and automatically reorders or offers the option to the user as defined by the user. A MYSQL database will be implemented. If possible, it should be configurable for different businesses. It may be stored and accessible from a cloud service. This will allow authentication from the cloud service or a mobile app. It may be able to connect to a Point of Sale system to sell out from the stock. This will remove inventory from the database through order creation. All this functionality may not be implemented immediately. See System Evolution. See System Architecture file for diagram.

System Evolution: In the future we would like to implement the use of scanners (may be implemented immediately) or image capturing apps like Raspberry Pi camera or phone and tablet cameras. Also, the use of cloud services to be able to login from anywhere using different devices to access the services. We would like to implement plug n play for UPCs (may be implemented immediately) in the future as well. There will be a mobile app that connects to the database through the cloud service, in the future.

Appendices: We will be using several different sets of software to build this. Visual Studio or Visual Studio Code will be our IDE. We will be using MYSQL ideally for the database implementation. We will use C# to code the application along with any other language or module to wire the devices. There is the potential for us to use a raspberry pi camera or mobile device now or in the future for UPCs. We would also like to implement cloud storage and back up using AWS or Azure.

Software Development Lifecycle

* Requirements analysis
  + Interview small businesses owners that are available
  + Come up with questions
* Design
  + Develop modules and classes
  + Develop naming convention
  + Develop flowchart or diagram
  + Develop target audience
* Coding/Testing
  + Code the software
  + Develop unit tests
  + Develop use case tests
  + Develop Integration Tests
  + Develop other tests we don’t know yet
* Deployment
  + Decide how it will be deployed(Cloud or desktop)
  + Create logs to monitor errors
* Maintenance
  + Refactoring
  + Scalability
  + User reviews