Design Specification

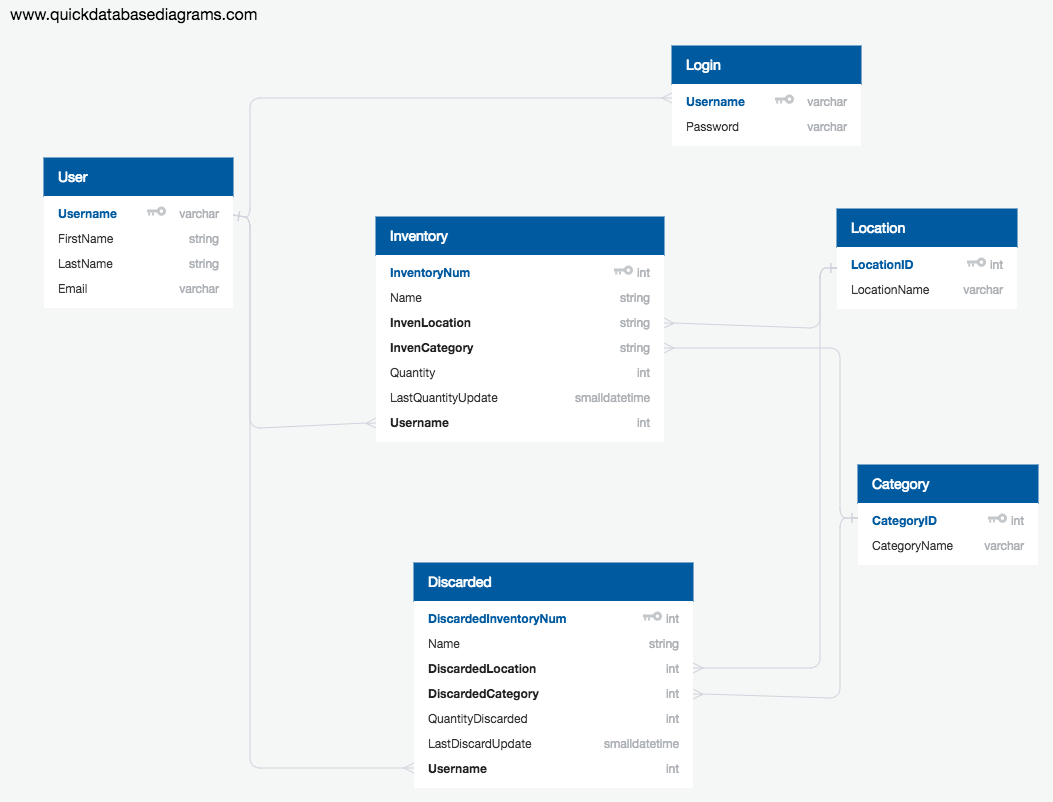
**Project Overview:** The context and purpose of this project is that, as of right now, the entire inventory system for the Sports Nutrition department on campus is carried out exclusively on paper. With multiple fueling stations on campus and several inventory items to keep track of at any given moment, the situation is fairly inefficient as it stands. This is why a web application with a database that has a remote CRUD interface and shows up collectively in a central location sounds like the perfect option.

**MVP:** The MVP details are as follows: a user-friendly inventory system with the ability to not only keep track of current items in the system, but also to be able to add, delete and update any items a user could desire. The system should also be able to keep track of any amount of inventory that thrown out, as to easily be able to discover when and where a certain item was damaged or disposed. Another key component of the system should be it’s ease of combing through the data. The sponsor detailed specifically that any user should be able to find or sort particular items based on product name, location, category, quantity, or the last quantity update. In terms of types of system users, there appears to be only a need for one. After speaking to the sponsor about this, she stated that no type admin or user hierarchy system is needed, and that all users should have equal access and privileges. This should obviously make things easier on my side of things, and should imply that all users will be utilizing the system in the same way.

**Development Environment:** The development environment I plan to use is fairly familiar, as I have used the same environment in a few projects in the past, and it has had exceptional results. For the back end functionality, I chose to go with PHP and MySQL. SQL is the standard for making several calls to and from a database, and partnered with my knowledge of this and PHP I procured from taking the Database Systems class, this seemed like the most straightforward approach. I plan to use Bootstrap for the front end, to keep a clean and uniform look throughout the entire application. Next, for the data sets, I plan to use my own at first, at least in the testing phase. Once the server and database are communicating correctly, I can slowly begin to implement the Sports Nutrition department’s actual product, as to ease the transition from the on-paper inventory to the fully-fledged final application. Finally, for the version control system, I chose GitHub, as this is the standard and most widely used platform for software development.

**Deployment Environment:** The deployment environment also seems like it will be relatively simple to handle. The only thing needed for the deployment side of things would be a server to actually host the website. In the past, I have used the university’s own Turing server to host my various projects, so using this again was the first thing that came to mind. Since when the final project is handed over to the sponsor, it will probably still be hosted on Turing or something similarly also provided by the university, this seemed like the best choice for everyone.

**Architecture:** I have already designed the architecture of my database, which in total ended up with six tables. A User table, which keeps track of the username, first and last name, and the email of all registered users. There is also a Login table which takes the foreign key of the username, and stores it separately with the hashed password for each individual user. There is an Inventory table, which keeps track of the product’s name, location, category, quantity, the date and time of the previous quantity made, and the username of the person who made the update, connecting it the User table. The Location and Category have their own separate tables and the IDs are pulled from these tables as foreign keys to the Inventory table, as to not cause a data clash by user input. The final table is the Discarded table, which is basically a mirror image of the Inventory table, but for products that have been disposed of, instead of product that is currently available.



There are several ways to go about implementing a web application such as this. In terms of the database, there are plenty of choices, including Oracle, MS Access or SQL. I went with the latter because of the familiarity with how to properly access and manipulate the data I want/need. For the back end, any programming language will do, such as Ruby, C, Java, JavaScript, etc. I chose PHP because as I was learning SQL, I also learned PHP as well, and the thought of using both together seems like the simplest route to go about completing this project successfully. Lastly, for the front end, I plan to use Bootstrap, JavaScript, and CSS. I chose these because I also have previously used all of these in some capacity or another. I did not spend too much time asking around for strategies used by others or even consider using anything else. I have worked on a few projects before that are strikingly similar, so I am pretty confident in how I want to approach it.

**Testing:** I plan to test the individual components of the system by first making sure that each aspect of the CRUD interface is working as it is implemented. That is, when I add the page that will create an inventory item, I will test it thoroughly and make sure the connections between the server and database are working as planned. Also, I plan to try to actually break the webpages myself, and using this tactic to reinforce creating a finished project that a first time user could easily and confidently figure out how to operate without crashing the entire thing accidentally. After all of this is completed, I would like to send the partially finished project to the sponsor, to have them test it themselves and give notes on how anything could be added or improved on.

**Project Timeline:** By the end of Spring Break (March 16), I plan to have a functioning login and register page, and a basic landing page that should show all inventory items currently in the database. In the month following (April 16), I plan to add the ability to create, update, and delete items to the database. I also would like to have a functioning page showing the disposed of items and the several ways of filtering and sorting the data completed for both the pages. In the final stretch before finals and dead week, I would like to finish up any outstanding issues and also completely finish the front end aspects of the application, the task I usually save for last. With this projected timeline, I feel as if I can clearly finish the required minimum viable product by the end of the semester.

**Bibliography:**

Books:

* *Learning PHP, MySQL, JavaScript, CSS & HTML5: A Step-by-Step Guide to Creating Dynamic Website* by Robin Nixon

Websites:

* <https://www.webdesignerdepot.com/>