SDLC for Grocer Website final project

Team Name: Fighting Senioritis

Team: Blake Conner, Chris Fischer, Kevin Wood

Team Goal: To create a useful and robust application that propels us forward into getting an A in this class.

Final Project: Grocery Application that users can go to for purchasing gifts found at their local supermarket.

Planning/Requirement Analysis:

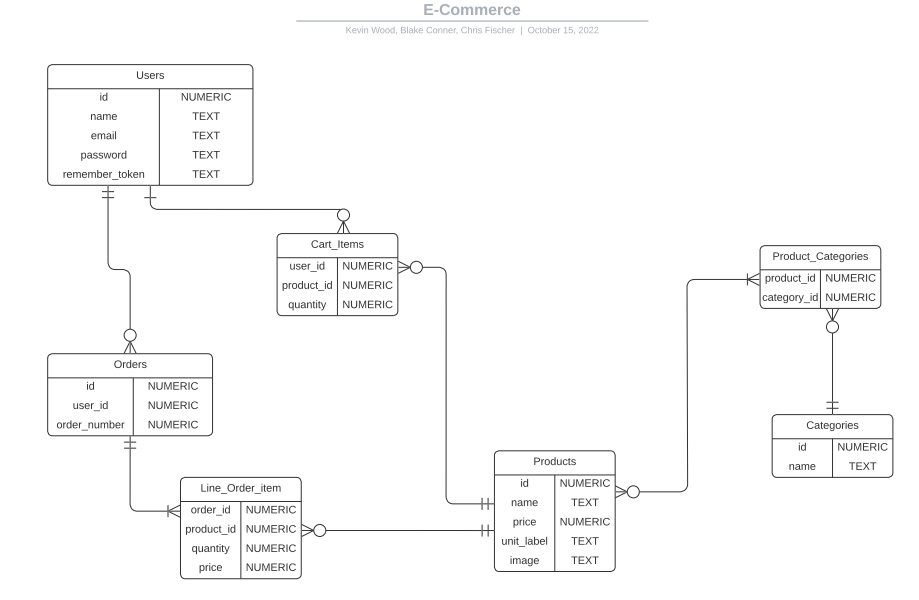
This Project constituted several different technologies and requirements for us as a team to collaborate and deliver on. We were given the task of developing a simple application that incorporated a PHP flavor, web interface and Database. In addition, utilizing any necessary collaboration tools and software. We were given this assignment towards the beginning of the semester and would have to present to an audience our working prototype before the close of the semester. To plan, we evaluated what sort of valuable applications we could look to build that might have a direct benefit to our team members in this job market. We narrowed down the focus to looking at an ecommerce website as this is prevalent web application that we all use nearly daily. We settled on building one for a grocery store as we all love food.

Define Requirements:

For this project, we would need to incorporate front end, middle, and backend technologies to build a robust application. First, we would need a web interface that the customer could access to take certain actions. These actions might include adding/removing from inventory, purchasing, reviewing receipt, and logging in. From there we would need a database to house certain details. Details such as inventory price/descriptions, login credentials, and purchase history. With those two pieces in place, we would bring them together in a framework to process the users’ actions and provide the necessary and relevant details on the appropriate page.

Designing:

The ERD Diagram for desired data flow is contained in the figure below.



Implementing:

Regarding implementation, we utilized a few different tools to build this application. We used Supabase(GIT) for version control and collaboration, Posgres(Database), Stripe API(Payment processing), Laravel(PHP), HTML and CSS(Interface & styling). We began by loading sample data into a database and building a single page to see how interactions may work. Then adding in more page details, followed by the authentication and payment process flows. Once this was achieved, we could add in additional details and functionality inside the cart and receipts.

Testing:

Testing was done throughout the development lifecycle and was performed at every new implementation step. Our goal was to ensure we were building from points of success. Whereas we were not carrying with us any technical debt, but instead moved into each new development stage with a well performing application. We worked in an agile method, looking to build and test each new component that we added to reduce any added complexities. Most of the testing involved was manual and user testing.

Deployment:

For deployment, we will be uploading our final package into a public directory, as well as in to D2L. The Radford PHP server was unavailable for us to deploy to but was accessible on all three team member’s devices and across multiple browsers.