Project Proposal

Team Members: David Pace, Andy Tran, and Lisa Gilmore-Montero

Abstract

It is incredibly time consuming to maintain even a small amount of digital products on the web. If you purchase a large quantity of products through wholesale and you want to turn a profit, exposure online in our technological era is a big help. However, technically inclined, newfound, entrepreneurs trying to sell large amounts of products have trouble posting them online in multiple areas. These guys are our target audience, but it's still a common problem for many small-medium sized businesses as well. Entire businesses exist solely to provide a solution to this e-commerce problem, but of course they require your credit card information for their free trial. Therefore we will build our own and distribute it amongst the masses, because we are computer scientists, and we are in Advanced Web Development taught by Ted Holmnberg.

We propose to build an ecommerce website where a person can sell any type of product they want. From the customer's side, they will be able to add products to their cart and pay using PayPal's callback request API. They also can create multiple carts and wish lists and add products to them. Administrators will be able to add and remove products to the store and categorize them. Ideally, the admin will also be capable of interfacing with major online marketplaces, so that their products can be added, removed, and managed from our frontend for these online markets. By doing this, the administrator can easily manage their products in one location but still have the advantage of large amounts of traffic.

For the frontend creation, we will use a combination of HTML, Javascript (JS), CSS, and Bootstrap. The website is an object oriented structured online store with procedural front end HTML generation done through the DOM using JavaScript. For the design of the website, we will be using CSS and Bootstrap to make the website look pleasing to the user and responsive. We want to create an interface that is easy for both the clients and administrators to use and understand. For the backend, we aren't entirely sure how to approach it. It's possible to handle REST api on a server that can be used to manage a sql database, but perhaps a noSQL approach will be better, one using XML or JSON. Once we discuss MongoDB more in the class we may decide to use this approach.