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

***User Stories***

*Include the user flow of your app for each type of user (admin, regular user, and guest users, for example). From start to finish, how will each user group interact with the app?*

*Mapping user journeys help identify all of the possibilities within the app from a user’s perspective. Use User stories to identify all app features.*

Storyboards for each type of user are shown in the E-commerce Storyboard PDF.

***Frontend features***

*Determine all the app features that the browser is responsible for. Define those features. Examples may include: login view, general view, user input, game view, etc. Note: Be descriptive!*

The idea for the website front-end is having three different views. The guest, login, and admin views are going to be different. The guest view will let the user filter and browse the products, add the products to the cart, and buy the products. The user view will have similar features, but an added advantage is persistent data such that the website keeps track of repetitive details such as the address, billing info, purchases, etc. The admin view will allow management of the website: adding, removing, and updating the products. When updating the product, an admin will have the capability of adding additional variables to the product. For example, one product may have its weight, but another may not have it as a variable at all. The admin will ideally also have the ability to autonomously import products from our site onto other mainstream markets like Amazon, Craigslist, and Ebay. If there are additional variables for a product that those markets require when importing products, there will be a placeholder to communicate that to the admin. For products that have been imported onto these other markets, we want to allow the admin to manage them from our site. For example, the admin will need to see the reviews, statistics, purchases, and other informational data given by the markets to sellers.

***Backend features***

*Determine all the app features that the server is responsible for. Define those features. Examples may include: authenticate user, manage shopping cart, broadcast chat, etc. Note: Be descriptive!*

The backend will have to handle user account creation, deletion, and modification. Once an account is created we must handle authentication. When a user is authenticated, they will need to be presented with their receptive frontends. For the administrator view we want to display mangitorial options. For this our backend must also be capable of creating, removing, and altering products and categories. In order to import these products to other marketplaces however, we will need to interface with the marketplaces using an API made available by them. Little research shows markets like Amazon do not offer these capabilities. In that case we would need to automate using their frontend in order to achieve importing products. In theory, web scraping and API usage should give us all the capabilities we need to make this project a reality. Then our only concern are captcha images; this may very well limit our project’s intentions. Aside from the admin backend, the user must be able to add, remove, and modify products in their shopping carts. Additional shopping carts can be created by the user as well, we are coining the term wishlists, but as far as the backend is concerned, the model for a shopping cart and wishlist are the same thing, just with different name variables. For the filtering feature, we will need to organize our data and implement industry sorting algorithms based off input. Guests are going to need the same options as a user, except they will not be able to create additional shopping carts or wishlists for example. In order to handle purchasing, we can implement Paypal’s callback API so that we are told if the money was received or if the purchase was canceled. From here we can handle the proceeds however we like. Ideally we will export purchases into our database and then into a spreadsheet, but if the product being sold is simply digital, like a NFT, we can entirely automate the buyer receiving the product. For the context of this project, where products aren’t always just digital ones, a spreadsheet seems like the best option.

***Database features***

*Determine all the app features that the database is responsible for. Define those features. Examples may include: data storage details, user data, shop items, etc. Note: Be descriptive!*

Because we are creating an online store, we must keep track of plenty of state variables for the users in our database. Their shopping carts, the products in those shopping carts, and as well as all the user’s personal information such as passwords. For security purposes our persistent data will not be stored in plain text. Additionally, the database will need to save the current state of the store itself so that it can dynamically be loaded at runtime. It will have to keep track of what products are available and the categories they are in. For this we need that each product and category state must be saved. Also, because we are importing and managing products on other servers, we must keep track of the information they give in persistent memory as well. That would be information such as the reviews, amount sold, and other general statistics. Finally, we need to store purchases made. This will contain not only the user that made the purchase, but also the products and their quantities.