TEAM - TEEMO

Clothing Part Picker

https://github.com/namanshenoy/326Project

Team Overview

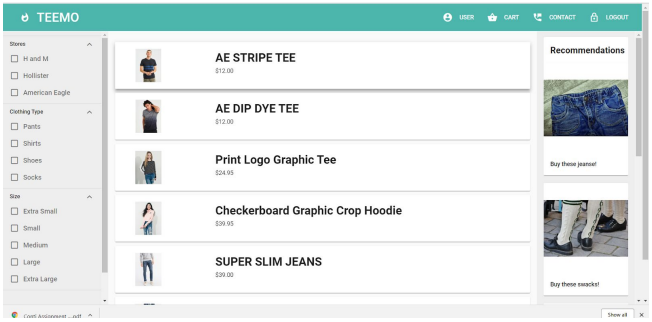
Team Member Name	Github Usernames
Brandon Conti	conti96
Corey Campbell	huglyfe
Corey Collins	Corey-Collins
James Hubert	JamesHubert
Naman Shenoy	namanshenoy

Overview

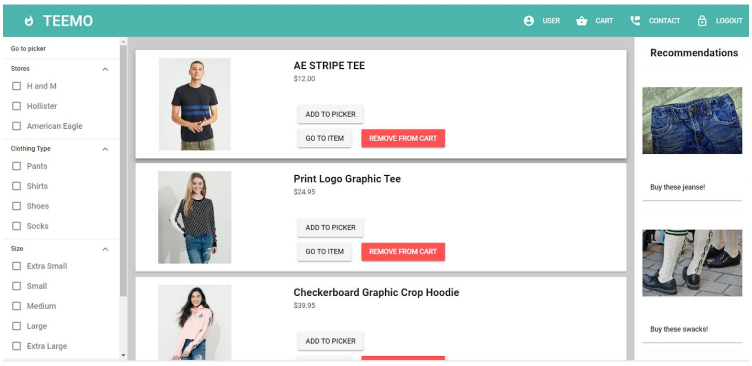
Our application will help users choose clothing items and outfits for the best prices possible by aggregating different stores. The site will make money through the referral links when a user buys a piece of clothing. This allows clothing part picker to be brand and site neutral, giving the site an incentive to give the user the best deal and outfit possible. This site will use APIs from CJ Affiliates in order to retrieve clothing data. The user can then search through the listings and add them to a cart which saves them to the database. The user can then search through their history and cart and find the best store to buy those items.

User Interface

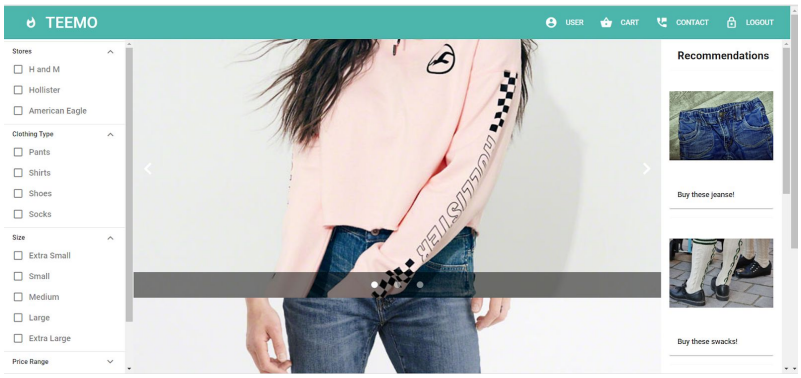
Our application has a profile settings page, a home page, a coupon page, a detailed item page, a cart management page, a “part picker” page, and a contact page. Each user is able to customize their experience using the site, searching for products based on stores, prices and clothing types. The detailed item view shows all of the relevant data about an item such as it’s price in all stores and specifications. The user is able to add items to their cart and go to a page to see all of the items in their cart and manage them. The part picker page allows users to view items they selected in order to build and outfit or compare items up close.



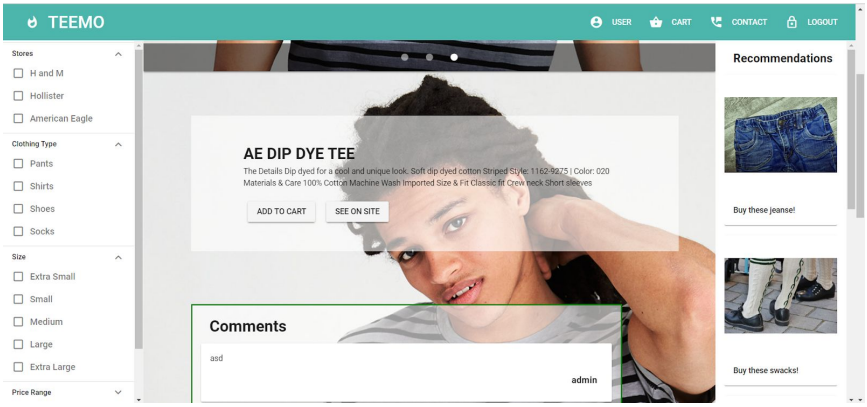
Our HomePage



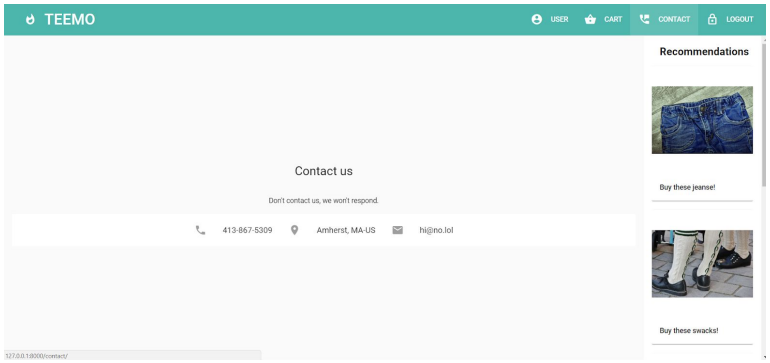
Our Cart Page



Part Picker Page



Item Details Page



Contact us page

Data Model

In order to function, our website requires data about the stores whose products we promote; the products that those stores sell (one to many for stores); coupons from the same sites (one to many for stores); user profile data, history (many 2 many for user profiles), and cart data (many 2 many for user profiles). The products are obtained from CJ Affiliates and the coupons are scraped from the web. All other data is user created or hard coded for the site. Attached on the next page is ERD that was provided by Django. The important links within the site are the landing page, the search page, the item detail page, the coupon page, and the user page. Our web framework passes through the Django template engine, passes minimal data with context framework. Dynamic data is handled through a Django REST Framework and XHR requests with axios.

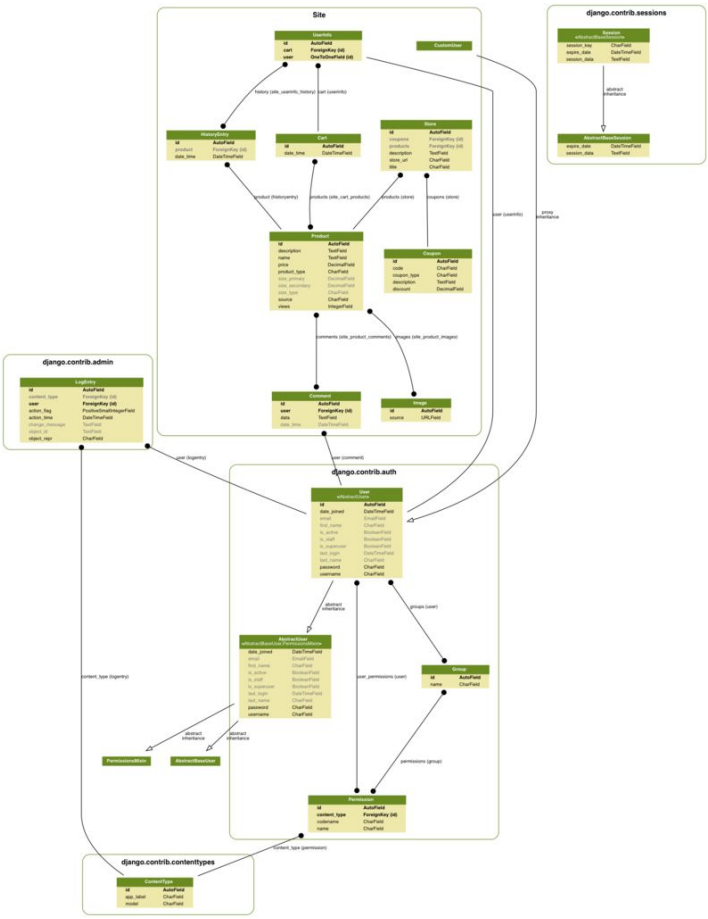


Figure 1. Datagram of Our Website

URL Routes/Mappings

URL	Description	Authentication
admin/	Used for site administration	Django administrator account needed
home/	The homepage for the site	Any

product/	A description page for a product	Any
comments/	Comments for a product	Any
cart/	Used to view the items in a user's cart	Normal account or higher required
cart/add/	Used to add an item to the user's cart	Normal account or higher required
cart/remove/	Used to remove an item to the user's cart	Normal account or higher required
products/	Used to view all of the products for a specific site	Any
checkItem/	API used to check if an item exists in the user's cart	Normal account or higher required
user/	Used to view account details	Normal account or higher required
signup/	Used to create a normal account	Any
accounts/	Django accounts for Register API (XHR)	Any
^ajaxLogin/	Used to log users in	Any
cloth_picker/	Used for the clothing part picker functionality	Normal account or higher required

Figure 2. Table of URL routes for Clothing Part Picker

Authentication/Authorization

For our login we are using an XHR request to communicate with the server. We are using Django permission classes in order to verify the user is logged in for pages that require authentication. We use a Login / Logout form to provide the login and logout functionality to the user. Registration is used in a form, and automatically creates a cart object for each user to ensure that each user has a unique cart. CartView also has permission_classes set to [IsAuthenticated], and the cart ensures that when you add/remove a product that it doesn't add/remove it from a different user's cart. In order for a user to request to add a comment we use a function (AddCommentAPI (request of session info, data [the comment])) which returns 201 or 404 based on whether posting was successful or not. Our website fetches comments by using another function we made (GetCommentsAPI(request of page id)), this returns comments[string] or 404 (error). Any unregistered user will only have access to view clothes. The cart, add to cart, user, and clothing picker links will not be available unless they are logged in and registered.

Team Choice

For our team choice we have decided to use vue.js for our front end. It makes the code for the front end far simpler to maintain and update. This is because Vue.js makes integration with javascript far easier. Our site is highly dynamic and requires lots of XHR requests, making vue.js an obvious choice for our team submission.

Conclusion

Throughout the semester our team has learned a lot about the web design process. We learned the basics of Django, Vueify, and Vue JS. Our team also had a lesson in how to collaborate with a large group on projects. We all learned git and how to use slack. Scheduling was a constant battle and is something that we all came away from with a better

knowledge of. After scheduling, the knowledge gap between group members was our greatest challenge to overcome. We had lots of skills within the team but they were spread between the different members. Setting up environments was a large challenge. Everyone had a unique set up and getting them all on the same page was difficult. Transitioning to Vue JS was more of a challenge than we anticipated, however we were able to make it happen before the due date. We all would have liked to have more web experience before the class. Having more python knowledge before taking this class would have also helped us make sense of Django faster. We also would have liked to have a unified environment before we started the project. This would have made our start up time more efficient. Despite this, we feel like our team accomplished the goals that we set out for at the beginning of the semester.