# Design and Implementation of an Online Marketplace Web Application

Technologies, design decisions, and evaluation

JACOB WILLIAMSON

Department of Information Science, Drexel University

In this project, I look at a full-stack implementation of an online marketplace. Using React for the user interface, Node.js for the API, and ElasticSearch for the database, I was able to create an early prototype of features including a user interface, search engine, and suggestion engine. Docker and Docker Compose were leveraged to make deployments simple, efficient, and effective.

**References:**

No References.

## 1 INTRODUCTION

The inspiration for this topic stems from the work done in a previous class where we designed a system with a similar theme. While I used the same domain, there is was little that transferred to this project. Technology choice mostly stemmed from technologies I was interested in but had minimal hands-on experience. I was most familiar with Node.js and Express, which served as a good foundation to build an API to help facilitate the data retrieval and transformation from the database.

## 2 PURPOSE

This app served as an exercise to create a full-stack information retrieval stack from scratch. Building up a system from a lower level and refraining from using external libraries to help with the information retrieval helps to teach about all the work put in to the tools that help facilitate information retrieval at a higher level.

## 3 SYSTEM ARCHITECTURE

The system is separated into 3 main components: a frontend application, an API server, and a database. I chose this split to create an effective separation of concerns between parts of the system. The frontend's only task is to display information. It is not intended to do any data manipulation besides receiving information from the API server. All the data transformation is handled by the API server. The API server will retrieve documents from the database and then parse it into the format that the frontend expects. The database is document-based, meaning that there aren't any "tables" and "columns" like a SQL database. Instead all documents are stored in JSON.

## 4 SYSTEM FEATURES

The main features of this system revolve around searching for products and viewing the product pages. Additionally, similar products to the current selected product would be listed as "suggestions." This system can handle free-text search queries and simple Boolean search queries. When performing a free-text search, the database will look at the title, description, and tags of the items and return them. They will then be sorted in

order of ElasticSearch's internal scoring system (See: Lucene's Practical Scoring Function). Boolean queries look solely at the tags of the item.

## 5 TEST DATA

Since the rest of the system is proprietary, the test data was handcrafted as a proof of concept. As such, it would be difficult to perform a scientifically valid test of the system. However, the system can adequately handle basic search queries with relevant results. The test data was created with two categories in mind: Kitchen and Electronics. I chose these categories as there is a certain amount of overlap between the general descriptors. When looking at items in one category, I would likely find a couple of items from the other category in your suggestions. While this may not seem like retrieving only the most similar and relevant items, it would be beneficial in the business sense. Given a large enough collection of items, users would naturally click through different items they may want to buy.

## 6 FUTURE IMPROVEMENTS

Given we only had a few weeks to implement the system, there was a ton of functionality I would have liked to implement that I did not have time to. One such feature would have been tailored product suggestions. Currently, suggestions are based on the details of the current item. By tracking a user's search history, a better algorithm could be implemented that suggests more relevant items for that individual user.

Other features I'd like to revisit and implement include but are not limited to: a better UI, custom URLs for each product, user registration and authentication, and a fully functional marketplace (product upload and purchasing).