
# Product Guide

By Prateek Swain, Steven Takeshita, Lucas Irvine, Dominic Whyte, Antony Toron

# A. User Guide

## 1. Introduction/Purpose

LottoDeal is a revolutionary new platform that combines the psychological appeal of lotteries with an e-commerce marketplace. Each seller is matched with thousands of interested micro-bidders who can buy virtual “lottery tickets” for any item they desire. An interested bidder can simply bid $1, $2, or $5 to contribute towards the item’s selling price, where the bidder’s chances of winning are proportional to the amount they bid. When the item reaches its goal, it is lotteried off to a lucky winner. This helps buyers get a chance to own items they desire at a fraction of its cost, while sellers can utilize economies of scale to sell their item as fast as possible.

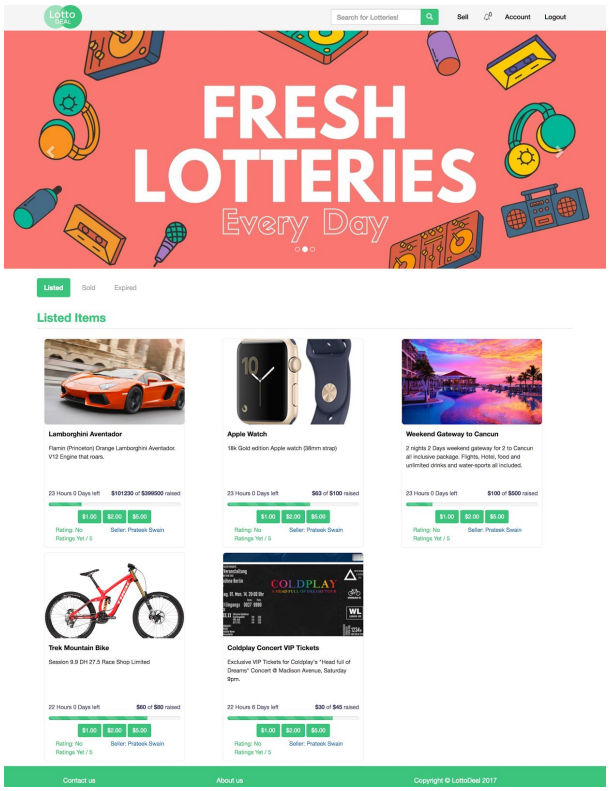
## 2. Features

### 2.1. Facebook Login

When you log into our site at www.lottodeal.us, you can start browsing immediately. However, to bid, sell, leave reviews, and get better item suggestions, you must login to LottoDeal using Facebook. On the top right corner of the navigation bar exists a button, “Login”. When you click it, a Facebook authentication modal drops down. We use Facebook because 1) it is universal, easy to use, and hassle-free 2) it provides us with demographic information. To logout of the site, you press the same button, which now says “Logout”.

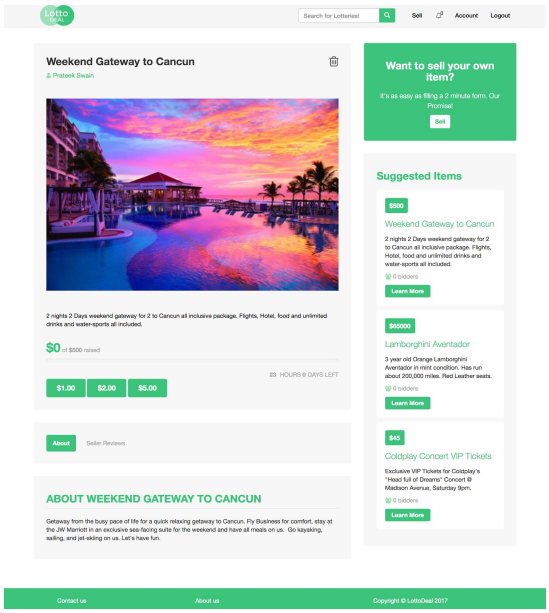
### 2.2. Home Page

When you log into our website, you are welcomed by our home page. Below the banner, you will see three tabs: Listed, Sold, Expired. Your default view is listed which shows all the items currently active. Sold Items shows items that were sold and expired items displays items that failed to reach their goal before their expiration date.




### 2.3. Item Card

Each Item has its own card with a compressed picture (which can be clicked on to open a detailed item page), along with a title (which can also be clicked for the item page) and a short description. At the bottom of each card is the relevant information: price of the item, a progress bar showing what percent of the goal has been reached, and time left till the product expires. There are three buttons to instantly bid on the item ($1, $2, and $5) which uses Stripe to process the payment. Below these buttons, we display the seller’s name (linked to his profile) as well as his rating.



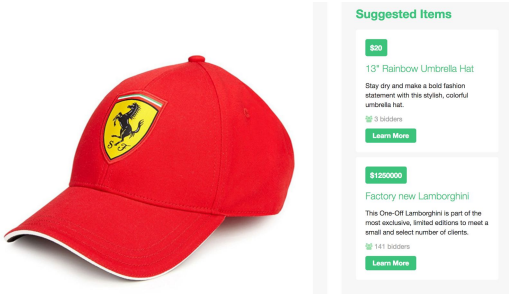

## 2.4. Product Page

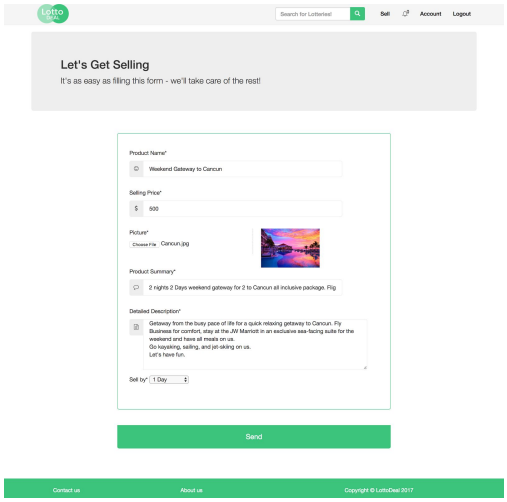
On clicking on the image or title of an item card, you're redirected to the product page. This shows all the information provided on the item card more clearly, along with a high-resolution image of the product. Right below this are two tabs: "About" and "Seller Reviews". By default "About" is chosen which lets you read the more in-depth description for the given item. You can also select the "Seller Reviews" tabs to conveniently read the reviews of the seller to verify his authenticity.

On the right side of the page, there is a button to redirect you to the sell page. Below that are the suggested items displayed. If you are the seller, you have the additional functionality of deleting the item on the product page by clicking the trashcan icon (top right corner just above the picture) which is only visible to you.

## 2.5 Suggested Products

On the product page, we display suggested products custom tailored to each person. The recommendations are based upon your demographic information from Facebook as well as other data points we analyse to give you recommended items that you might be interested in.

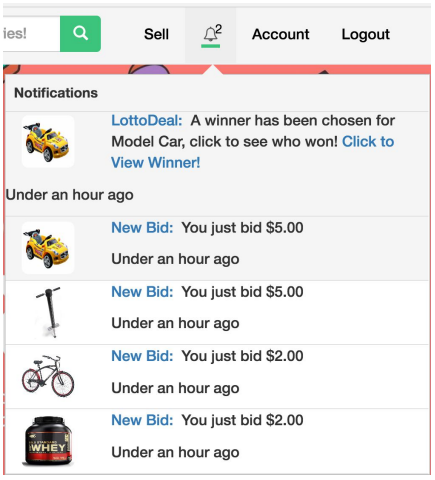



## 2.6 Sell Page

On the navigation bar is a button called "Sell". On clicking this, you are redirected to a sell page. It's a simple form where you fill out an item's information including title, price, short description, detailed description, expiry of item (1 day, 1 week, 1 month, 30 seconds), and upload a picture. You can instantly view the picture you uploaded next to the upload picture field. On pressing submit, your item is posted on the homepage with an individual product page custom generated for it.

## 2.7. Notifications

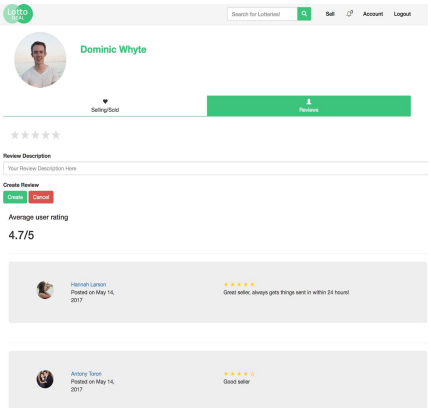



The notification option is displayed with a bell on the navigation bar. It has a number next to it (slightly above it on the top right) to show unread notifications. Unread notifications are slightly darker in shade to differentiate them. There are three kinds of notifications you can get:

1. When you bid on an item. The image of the item, along with info about your bid (amount etc.) is displayed
2. When an item you bid on has a winner: On clicking the link in the notification, a graphic pulls up a modal and shows the name of the winner.
3. When an item you bid on expired/was deleted: Notification lets you know the item expired and that you've been fully refunded.

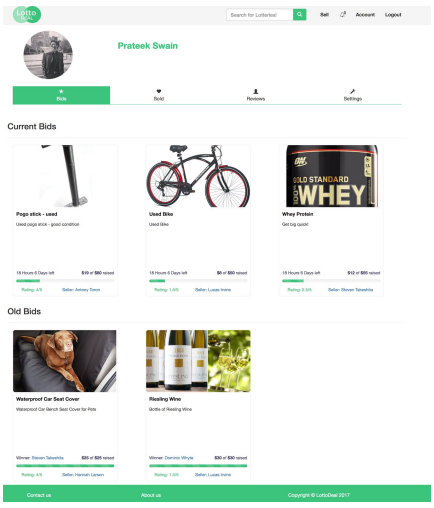
## 2.8. Search

There is a search filter on the navigation bar which allows a user to filter items on the homepage. It compares the text input with the title of each card to narrow down results.




## 2.9. Reviews

On a user profile (linked through the item card or Product Page), you can select the "Reviews"' tab and see previous reviews left for the person as well as their star rating out of five. You can also fill out your own review which includes 1) a star rating out of five 2) a description of your experience. It will be listed on the seller's "Reviews" tab and will change his rating according to the new average.




## 2.10. Profile

There are two kinds of profiles: Your private profile and other people's public profile. A person's public profile can be accessed from the item card/product page of items they're selling. It shows their currently listed as well as sold items under the "Selling/Sold" tab. All their reviews are also available to be seen (as well as added by you) under the reviews tab. You can view your personal profile by clicking "Account" on the navigation tab. In your personal profile, you can view all your current and expired bids on items under the "Bids" Tab. You can see your sold items and currently listed items under "Sold" tab, read reviews people left you and your rating under the "Reviews" tab. Lastly, you can

change your email address (where we send you emails for when an item you bid on has a winner, or the item has been deleted or expired) in the “Settings” tab.

## B. Developer Guide

The below guide will help a developer understand the implementation and system of LottoDeal.

### Overview

LottoDeal was developed using the MEAN stack, with additional libraries being used.
The technologies can be divided into:

1. Front-End: HTML (Bootstrap), CSS, jQuery, AngularJS
2. Middleware: NodeJS, Express
3. Back-End: MongoDB

#### 1.2 First-tier: Front-end

The first layer is the client side or front-facing web application. The majority of the front end was written using HTML and CSS, incorporating AngularJS onto the web application, allowing for easily updated DOM elements and dynamically changing data displayed to the users. HTML was used to provide the basis for all of the web pages and with a combination of Bootstrap and CSS, we made the website responsive and clean. jQuery was used for being lightweight and allowing for quick DOM manipulation in the javascript.

On the front end, we use AngularJS to create dynamically loaded fields such as items and reviews. The bulk of the dynamic population of data on the site was done in an AngularJS module located in a file called serverModule.js to modularize all of ajax calls that the front end would initiate to get and post from the server and database (utility functions were in utilsModule.js). Each js file can call services in the server Module and populate the HTML using Angular to load the dynamic content.
The HTML and CSS can be classified as:

- ➔ item.html -- displays the item with appropriate information and suggested items
- ➔ user.html -- parses the reviews and the current items being sold by a user
- ➔ profile.html -- pareses reviews, current items being bid on, sold, and settings
- ➔ index.html -- homepage with all the items being sold, currently selling, and expired
- ➔ sell.html -- page where a user can list an item to sell
- ➔ contact.html -- contact page so that feedback can be submitted
- ➔ about.html -- learn more about the creators of the project
- ➔ main.css -- CSS styling for all of the HTML

Each page had its own controller as defined by angularJS (in a corresponding JS file) in which we could call the appropriate function from services imported from serverModule.js to populate the page with corresponding items or reviews, allowing for a more modular way to distribute reused code.

For the Facebook Login, we use Facebook’s API so that we can parse their unique ID’s, gender, and age. Facebook has created a great API for easy access to login and logout so that we can outsource all of the authentication system to a third party. Users will know that they are safe because their profiles will be kept secret by Facebook. Moreover, we have increased security because we use Facebook’s access

tokens, rather than solely their ID's, so that people cannot adjust the JS and input their own ID. We validate the tokens on the backend to make sure that the actual user is accessing their own information and a hacker is not trying to impersonate another user.

### 1.3 Second-tier: Server

#### NodeJS and ExpressJS

NodeJS is a JavaScript runtime environment that executes JavaScript code server-side. Express in particular, a lightweight and fast web-development framework for NodeJS, provided an easy way to set-up a web-service.
If you are using a Mac, to install Node.js, make sure you have homebrew installed on your computer. If you do not, type the following into your command prompt to install homebrew:

```
$ /usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

Next, in a command prompt type:

```
$ brew install node
$ npm install
```

For Windows users, Node can be installed directly from the Node.js website: https://nodejs.org/en/download/.
For package managing, use Node Package Manager (NPM), installed similarly to NodeJS.

One example of using NodeJS is when a user of our site tries to bid on an item: this creates an Ajax HTTP request to the server. The HTTP request is parsed by the server and the corresponding database call is initiated with the appropriate data being returned. The Express server code is located in server.js and handles all of the get and post requests from HTTP, while additional modules for notifications, lottery, and suggestions are imported to server.js. To run the server, write in the command prompt:

```
$ node server.js
```

#### Stripe

Stripe allows companies to accept payments over the internet in a secure way. Therefore, no user payment information was stored on our database, as Stripe has their own infrastructure to prevent fraud. Although Stripe is not a long-term solution for LottoDeal because of the transaction fees that exist, it provided an easy mechanism for all of the payment processing functions that we needed, including charging a user when they bid on an item and refunding all users when an item is deleted or expires.

#### SendGrid

SendGrid is a platform for sending emails to users. We use SendGrid in conjunction with our notification system to make sure that users receive confirmation of any payments they make, as well as whether they have won the item.

### 1.4 Third-tier: Database

The final layer is the database, for which we use MongoDB. Mongo is regarded as a scalable database, which is important considering that many items can be posted on the page so our database should be able to handle large amounts of information without being slow. Mongo also allows for the structure to change as the program runs and can do so asynchronously, which is helpful considering that we have many fields to fill and they do change when a user adds a bid or an item is deleted. This particular database also has great integration with NodeJS and AngularJS, allowing for a consistent codebase and technology stack.

To make the development of the database easier, we used a library called mongoose, which is a mongo object model for node.js. We used this because it makes developing mongo documents and collections a lot simpler by employing schemas. We also included the mongoose calls in server.js.

To install mongoose and mongo for Mac users, open a command prompt and type:

```
$ brew install mongodb
$ npm install mongoose
```

For windows users: https://docs.mongodb.com/manual/tutorial/install-mongodb-on-windows/

For LottoDeal, we created three schemas, one for the users, items, and images. In each, the below are the example structures with their respective categories given a more full description. By using mongoose, the values can be easily accessed in Node.

#### User Schema

fullName: Name of user
fbid: Facebook ID of user
email: Email of user
pictureURL: The URL of the profile picture on Facebook of the user
bids: A dictionary of bids, where the key is the item ID the bid was placed on and the value is the amount of the bid
reviews: An array of reviews on the user, where each review contains the userID of the person who reviewed them, the number of stars they were given, a review description, and the date the review was posted
userInfo: UserInfo of the user, including the age and gender of the user taken from Facebook
notifications: An array of notifications

#### Item Schema

title: Title of the Item
price: How much they are selling it for
datePosted: When the item was posted
expirationDate: When the item expires

amountRaised: How much money has been raised already
bids: Array filled with each user that bidded on the item and relevant data
shortDescription: short description of the item that will be displayed on the front page
longDescription: longer description of the item that is displayed on its item page
img: Stores the compressed version of the item image (compressed using the Jimp library)
sold: has the item been sold
expired: is the item expired
sellerID: the seller of the item's Facebook ID
sellerName: the full name of the seller
winnerID: the Facebook ID of the winner
winnerName: the full name of the winner

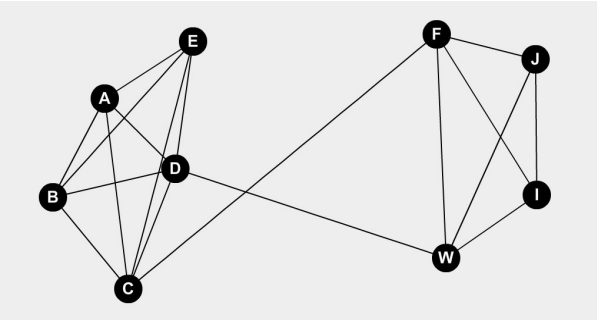
### Image Schema

itemID: the Mongo generated item ID that is associated with the item that the image is of
img: the non-compressed image of an item

These schemas are used to create objects of the corresponding type whenever new items, images, or users are created on the database. Therefore, we easily manipulate these data structures whenever requests are parsed by the server. Because of the asynchronous nature of Mongo, we are able to handle many calls at once and the server can still run quickly.

## 1.5 Implementation Details

### Suggestions




The suggested items graph was created using a graph API called cytoscape. Cytoscape is a JavaScript library for network visualization which allowed us to construct the graphs of users and items so that the suggestions algorithm could be run on the created graphs.

The suggested items algorithm works by creating two complete weighted graphs connected to each other also with weighted edges. Our algorithm computes suggested items for a user based on demographic similarities that that user shares with other users, as well as based on the similarities between items that they bid on. Therefore there is a complete user graph which calculates the similarity of people with others and assigns the edges that weight. Next, the items complete graph is created by examining the titles of the items using the string-similarity library (uses Dice's algorithm) and assigning the edges this weight. Finally, we add edges between the two graphs if a bid occurs between the two groups and this new edge is weighted by the bid amount. Finally, we find the shortest path between the given user and items using dijkstra's algorithm. We then display these items on the suggested items whenever a person clicks on an item.

### 1.6 Debug Mode

Currently, all of the code is in full production mode. Therefore, if the developer want to debug it locally, all of the URL's in which they use the IP address (http://162.243.121.223:8000/), must be changed to https://localhost:8000/. This will run the node server locally and allow the developer to see the console log. To help debug notifications, add yourself as an admin in the ADMIN_FBIDS array in communications.js using your facebook id.

### Testing Script

There is a testing infrastructure located on the backend to check major functionality, as well as some corner cases that may occur during the site working. This testing script was created using mocha tests, which are particularly useful in testing MongoDB and Express, because the mocha test framework can run tests asynchronously. This means that as soon as a test completes successfully, it can be marked as passed without worrying if it completes before another test. This testing script can be run by going to the LottoDeal-Backend repository where the server.js file is located and running:

```
$ npm test
```

(assuming that the server is running at that moment). Note that if the tests are not run, this means that the server is likely not accepting requests from the origin of the testing calls, which will just require updating settings on your local server to allow requests from null origins (localhost). When the tests run, several tests are created and run asynchronously which should add items to the database, as well as users, but are deleted after they are run to make the tests non-invasive. Functionality that is tested includes, but is not limited to: user creation, item creation, bidding on items, deleting items, deleting items that have bids, deleting users that have bid on items, etc. This testing script was designed to be run on server updates, to ensure that important functionality on the website is working properly on the backend.