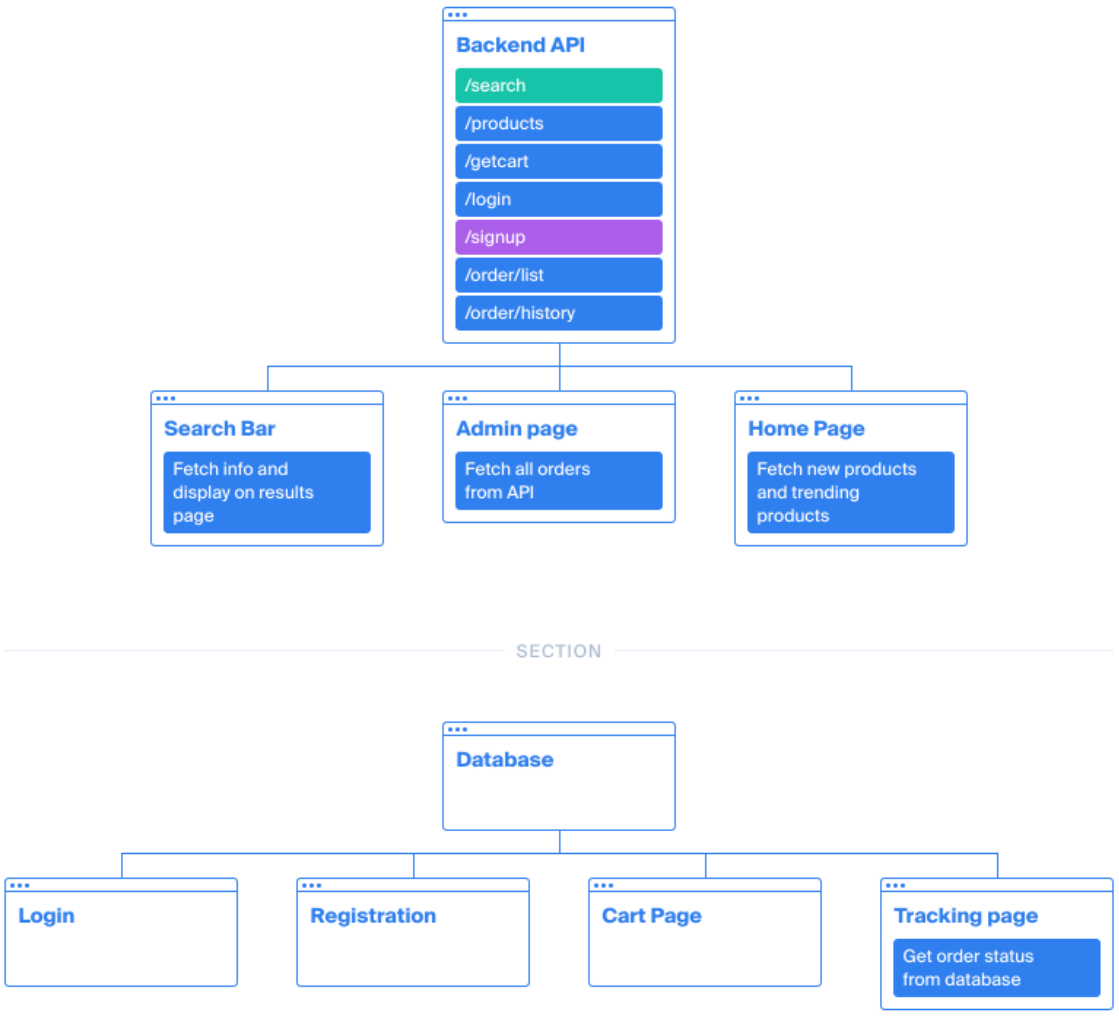
SpIcY VaNiLlA - David Chen, Jing Yi Feng, Matthew Yee, Jeremy Kwok
SoftDev
2022-12-02
Target Ship Date: 12/20/22

Components
Map



Backend Design:

1. /products - Procure a Bestbuy API key and use it to get a list of items for the front page
2. /login - login for a user

3. /signup - signup for a user
4. /getcart - return items in cart
5. /order/list - return a list of all orders currently in your shopping cart
6. /order/history - return a list of all old orders and their info
    a. /orders/<id> - return a specific order by id, provided session OR email and order ID
7. /search?q=<QUERY> - return products that match said query
8. /ip - return information about request IP
9. /email - send email/newsletters to users on the list (only authorized admins)

Frontend:

1. Login/Registration Page
    a. Allows users to login/create an account
2. Landing page
    a. Allows users to browse items and add them to their cart
    b. Use Bestbuy page for inspiration
    c. Allows users to browse through broad categories of items in addition to searching for specific ones.
3. Help/Contact Page
    a. Allows users to contact a staff member (fake)
4. Admin page
    a. Button on navbar, only visible/accessible to admins, allows them to view the status of any user's order and send newsletters to users.
5. Shopping Cart Page
    a. Lists all the items the user has in their cart at the moment
    b. Retrieves current cart items from backend database

        c. Offers an option to check out
            i. Displays dummy checkout page with 4242 4242 4242 4242 as a test credit card number
    6. Tracking page
        a. Allows users to track the status of their past orders

Database:

1. Table for tracking user login info
2. Table for users, orders, and shopping cart contents (users in both tables should preferably be linked)
3. Tables for items (sorted by category) - this could provide some sort of caching instead of hitting the Bestbuy API everytime we request

API Used:

- BestBuy API
- Radar API - a more flexible alternative to Google Maps API with up to 100,000 free requests per month and offers REST API
- IP location API - used in conjunction with Radar API
- Mailchimp API - used to send confirmation emails and or newsletters

Why Bootstrap:

- Bootstrap looks nicer and is more intuitive to use than Foundation
- Use builtin cards, buttons, navbar, carousel, etc

Organization:

- 3 Tables
    - Users Table

Username	Hashed Password

Matthew	testingtesting123
Duck	BreadIsGood

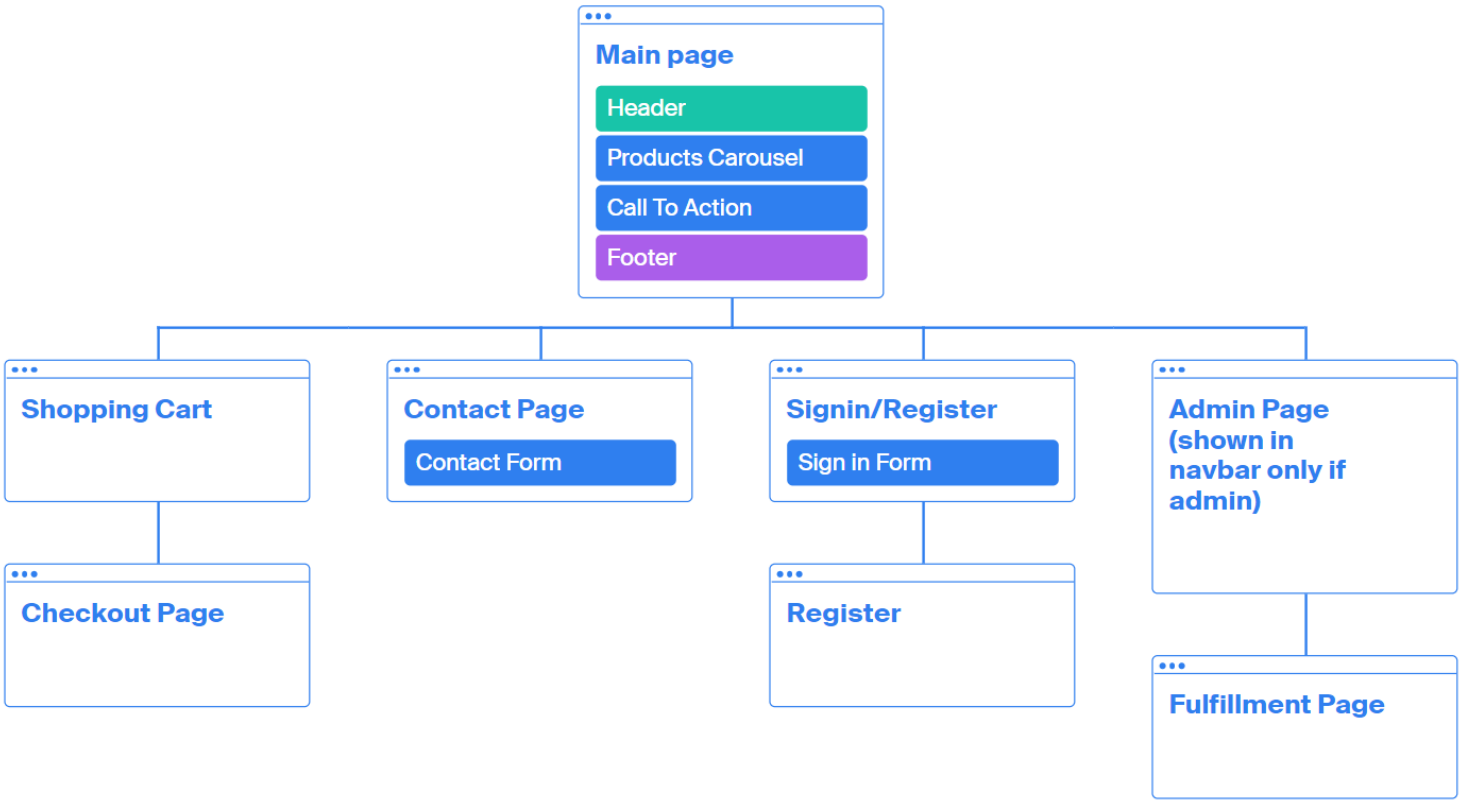
- Order History Table

Username	Shopping Cart Contents	Order History (IDs)
Matthew	[item_ID1], [item_ID2], [item_ID3]	[Order1], [Order2]
Duck	[item_ID1]	[Order3]

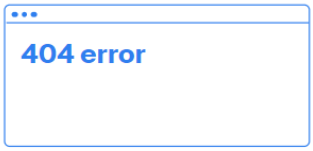
- Orders

Order ID	Product Name(s)	SKU (Product Identifier)	Order Date	Price (Total)	Status (Unshipped/Shippd)
0001	Water, Knife	itemID1 itemID2	12/5/22	$17.86	Shipped

Sitemap



SECTION



Assignments:

- P1M David - Backend API Implementation
  - Flask (serving pages, redirection, and safety checks)
  - General assistance with all backend-related tasks
- Matthew - Database Engineer
  - Storing login information
  - Storing orders and shopping cart items

    - Retrieval of data
- Jing - Bootstrap Engineer
    - Site Navigation
    - Site formatting
    - Collecting user input
- Jeremy - API Linkage
    - Pulling and displaying/implementing data from backend

To Do List (Bold if completed)

1. **Login/Logout Functionality** [<1 day]
2. Set up history and order databases [<1 day]
3. Create a homepage (all pages will be linked here) [1 day]
4. Display Product Details (start with name, image and price) [2 days]
5. Create shopping cart mechanism [1 day]
6. Create a mock checkout flow [<2 days total]
    a. Confirm shipping location with Radar API [<1 day]
    b. Add order to databases [<1 day]
    c. Send confirmation email with order details to user [<1 day]
7. View past orders page [<1 day]
8. Search functionality [1 day]
9. Sort functionality [1 day]
10. Categorize items [<1 day]
11. Admin management portal [1 day]