Pages

- **(/)** - Home page
- **(/products)** - Page to view all products. Will contain components to view the top/lowest selling products.
- **(/products/missing**) - Page to view all the products that went unaccounted for. Will contain a component for financial analysis.
- **(/products/returns**) - Page to handle data involving the return of products. Will contain a component return analysis.
- **(/products/add_product)** - Page that contains a form to add a product into the system.
- **(/products/edit_product**) - Page to retrieve current product info and ability to alter it
- **(/order**) - Page to view all products that are eligible for re-order.
- **(/vendors**) - Page to view all vendor information. Will contain a component to add a vendor.

METHODS TO RETRIEVE INFORMATION

- **get_all_products()** - return a list of all the products present in the inventory with the following information: [id, name, company, selling price, product_weight concatenated with the weight_measurement, remaining quantity].
- **get_product()** - Use a form with user-input data to retrieve a specific product by its name from a db query with the same information format as above. Use regex to return any products that match the pattern of the input-string. If the product does not exist, notify the user via an error.
- **get_vendors()** - Return all the information in the vendors table
- **get_top_sellers()** - Return the top 20 products on the criteria of having the highest integer in “amount_sold” from the Transactions table. Return using the same format used in get_all_products().
- **get_lowest_sellers()** - Return the bottom 20 products on the criteria of having the smallest integer in “amount_sold” from the Transactions table. Return using the same format used in get_all_products().
- **get_products_to_order()** - Return a list of all the products that are at or below their threshold amount. Return in the following format as a row: [product_id, product_name, product_weight, remaining_quantity, amount_below_threshold]

METHODS TO ENTER/UPDATE INFORMATION

- **add_product()** - Using the form available in the (products/add_product) webpage, input the required information in the data fields. Perform data validation:

  1) By first checking if all the fields contain the correct data/data type.

  2) Then verify that the product does not already exist in the database using the product name and product weight (ensure data is formatted properly to be checked against DB). If the product already exists, notify the user and offer to update the product instead.

  3) Once validation is completed notify the user that the product has successfully been added into the system.

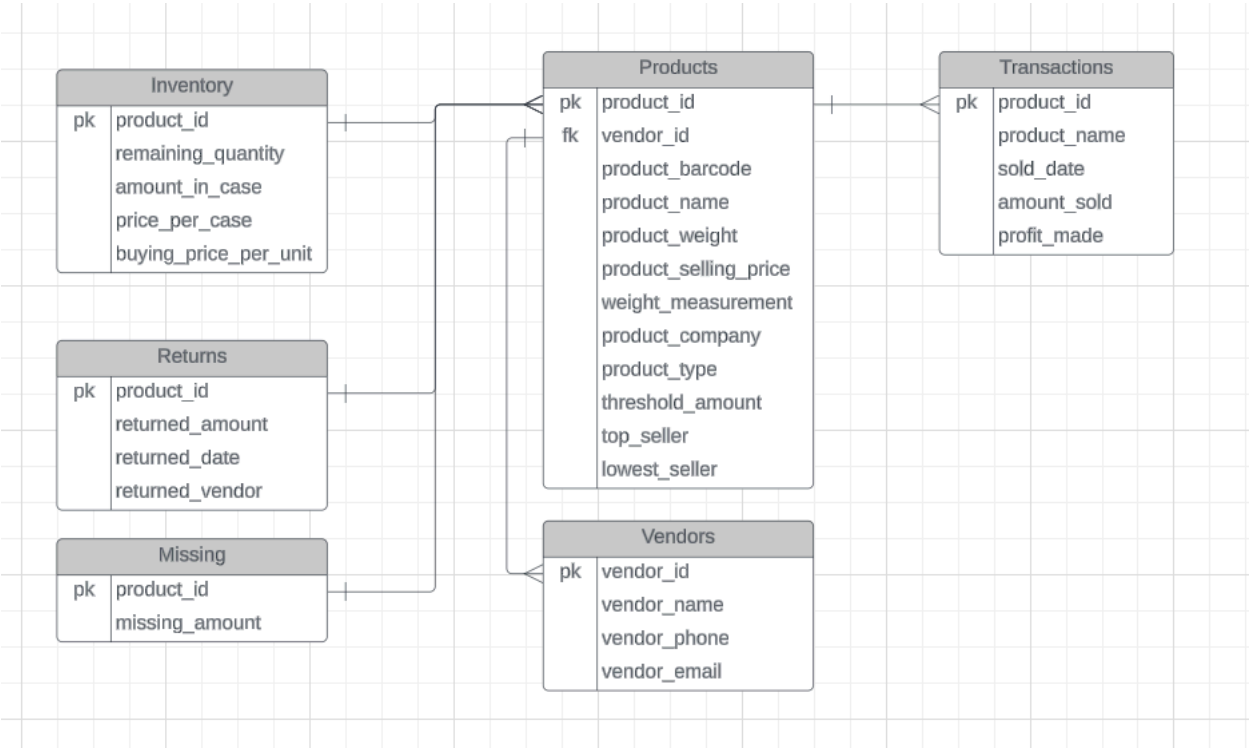
- **update_product()** - Using the list of products in the (/products) web page we can add a component/button to each row that will pass all the information of the given product to the (/edit_product) webpage. The user can then alter the form on this web page to update the product as needed.

- **remove_vendor()** - Using a component at the end of each row in the existing list of products, mark a product (row) as ready to delete. Prompt the user to verify that they want to delete the product.

- **return_product()** - Using the (products/edit_products) page we can edit the quantity of a product. This can be accomplished by creating/using a component to handle this for us, simply have the user input the amount to be returned. Validate that this amount doesn't impact the total quantity of said product such that its quantity will go below 0.

- **add_vendor()** - Using the form in the (/vendors) web page, let the user input data into the fields. Validate the information and add it to the vendors database.
- **remove_vendor()** - Using a component at the end of each row in the existing list of vendors, mark a row as ready to delete. Prompt the user to verify that they want to delete the vendor.

Database Schema/Tables



USING MySQL

Edgecase Handling

- Need to handle cases of missing/damaged products (products that won't be transacted). This can be accomplished by manually resetting the present amount within the inventory via the (/edit_product) web page. This ensures account for this real-world scenario whilst maintaining data accuracy.