ShiptApp

January 2, 2018

1 SHIPT API APP-Aysun Far

1.1 Capibilties of API:

This API can be used by both customers and vendor. It has the ability:

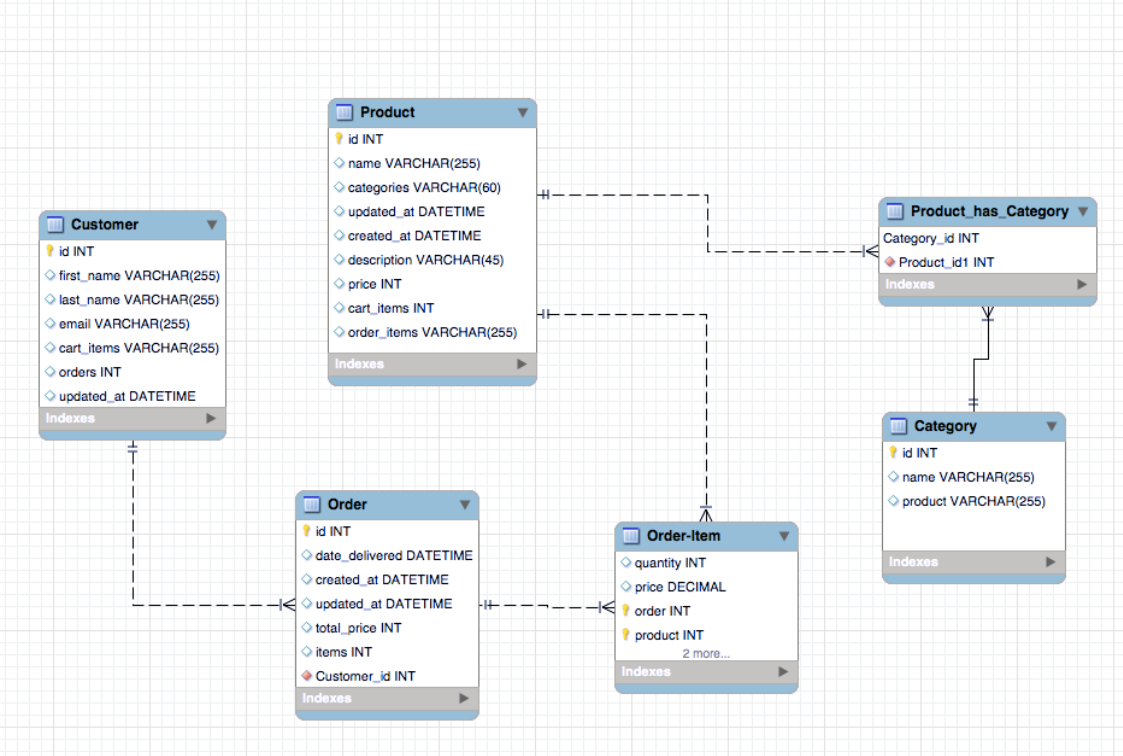
- Vendor:
  - Add category (add several categories).
  - Add new products and assign several categories to the product which it belongs to.
  - It also has the capability to add new customers, and orders.
  - Generate the sales summary report and save it as .csv file.
- Customer:
  - Customers can use the API to submit their orders, including several product in each request.
  - Display the order history for each customer.

1.2 Technogies Used in API:

- I build the API by using the Flask framework. I was going to write the SQL query but due to fear of SQL injection, I decided to write in Flask which is more secure.
- I made ERD diagram, just to have an understanding how I want to build the app. I used many to many relationship for product and category. The rest of relationship are one to many.

```
In [1]: from IPython.display import Image
        Image(filename='Screen.png')
```

```
Out[1]:
```



2 Getting Started

The requirements to run this app is to install: Flask and SQLAlchemy.

Open two terminals, once navigating to two folder where it was saved. Please have one terminal run this command:

- export FLASK_CONFIG=development
- export FLASK_APP=app.py
- flask run

The app will default to port 5000.

We want to generate the database. Please have another terminal, run these commands:

- chmod +x commands.sh
- ./commands.sh

Go to your browser and type:

- http://127.0.0.1:5000/categories - GET all the Categories.
- http://localhost:5000/customers - GET all the customers
- http://localhost:5000/customers/1 - GET the id of customer 1.
- http://localhost:5000/orders - GET all the Orders placed by the customer

- http://localhost:5000/orderitems - All order items.
- http://localhost:5000/products - GET all the products in the database.
- http://localhost:5000/ordersummary/recieve/ Download the report of order summary.

3 Usage (with curl tool)

To add new customer:

```
In [ ]: curl http://localhost:5000/customers -H "Content-Type: application/json" -X POST -d '{
```

To add new categories

```
In [ ]: curl http://localhost:5000/categories -H "Content-Type: application/json" -X POST -d '
```

To add new product

```
In [ ]: curl http://localhost:5000/products -H "Content-Type: application/json" -X POST -d '{"
```

To add new orders

```
In [ ]: curl http://localhost:5000/orders -H "Content-Type: application/json" -X POST -d '{"pr
```

An API endpoint that accepts a date range and a day, week, or month and returns a breakdown of products sold by quantity per day/week/month.

```
In [ ]: curl http://localhost:5000/ordersummary -H "Content-Type: application/json" -X GET -d
```

4 Additional Questions:

5 We want to give customers the ability to create lists of products for a one-click ordering of bulk items. How would you design the tables, what are the pros and cons of your approach?

The many to many relationship approach would be that I would basically have an Order table which contains orders placed by multiple customers (who are listed in the Customers table), and a customer may place more than one order. The Products table contains the individual products I sell, which are part of many orders in the Order table. One order may include one instance (or more than one instance). The Many-to-Many approach does hide an entity. For this reason, identifying and adding the hidden entity to the model may eliminate Many-to-Many relationships. This usually ends up with an extra entity being added, which then has a One-to-Many relationship with the existing entity. I would also add a Payment table and Address table will be created to persist the payment method details and address information of the customers where they will have one to many relationship.

6 If Shipt knew the exact inventory of stores, and when facing a high traffic and limited supply of a particular item, how do you distribute the inventory among customers checking out?

To distribute inventory among customers when facing high traffic, there are two different approaches. One approach is to satisfy one customer by placing and satisfying all their orders. For example if there are 2 customers one customer ordered 5 apple, and another ordered 10 apples. But our stock only has 6 apples, we can give it to the customer who ordered 5 apples. There is another approach which is not the best but we can porpotionally distribute it amongst the two customers, to satisfy both customers. We will also indicate that the item is out of stock and it will be saved for backorder.