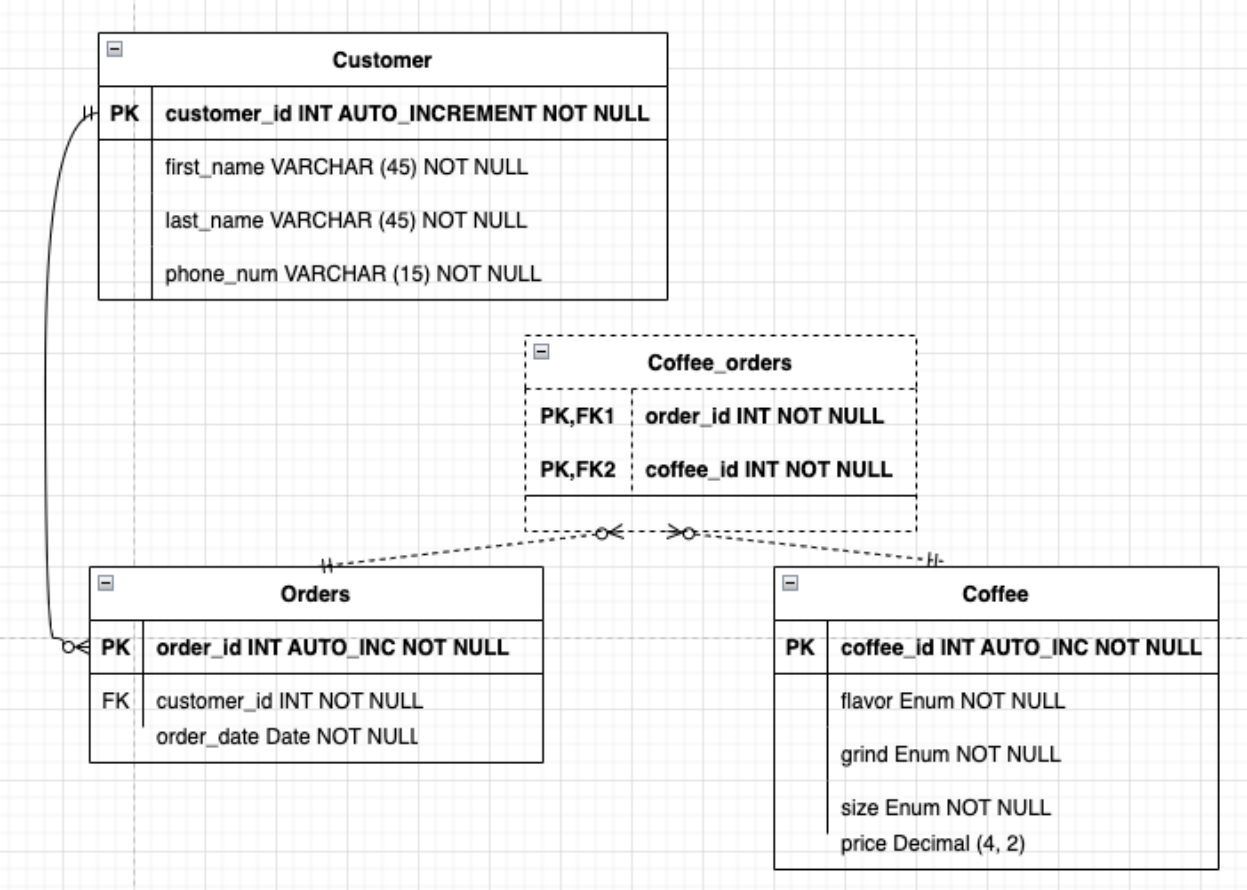
Coffee Subscription Service



GitHub Link:

https://github.com/joshuatmello/PromineoTechFinalProject

Project Participants:

Josh Mello

Executive Summary:

I LOVE coffee. There is actually local coffee roasting company that I really enjoy and they are very much a start-up, operate-out-of-their-garage type of business. They sell their product at the farmer's market but do not have a website and have not expanded beyond the farmer's market or local deliveries right now. My thought is that it would be fun to create and implement a RESTful Web API with full CRUD operations on a MySQL database that could ultimately be provided to this company in the long-term, and after I finish working through my "Stretch Goals". Since they only sell whole bean or ground coffee of a variety of types, I thought it would be a good way to approach this project. Who knows, if it works out, I might be able to get some free coffee out of it!

Initial Features:

Entities:

Customer, Order, Coffee, Coffee_Order

Customer: customerID; firstName; lastName; phone; email

Order: orderID; customerID; orderTotal

Coffee: coffeeID; flavor, size, grind, price

CoffeeOrder: FK's from Order, Coffee, OrderRequest

Relationships:

One-to-many relationship:

customer to order

Many-to-many relationship:

coffee to order

the join table Coffee_Order connects the Order and Coffee tables in a many-to-many relationship

The goal of my API will be to be able to perform the following operations:

CRUD on customer

CRUD on order

Coffee- GET

CoffeeOrder- GET

Stretch Goals (to be completed if time allows, or after graduation):

In the future I think it would be good to add the following:

-a way for collecting money

-an option for subscribing beyond the weekly subscription (i.e. a way to choose to subscribe for delivery every 2, 4, or 6 weeks).

Probably added through a Subscription Table (On/Off; Boolean) with Duration (enum 2-4-6)

Final Requirements:

Project Requirements:

- Database design which contains at least 3 entities and 3 tables
- Contains all CRUD operations (Create, Read, Update & Delete)
- **Each entity** should have **CRUD operations** with one entity having all 4 CRUD operations (Create, Read, Update & Delete).
- Contains at least 1 one-to-many relationship
- Contains at least 1 many-to-many relationship with one or more CRUD operations on this relationship
- **Required**: **REST Web API Server** tested through Swagger, Postman or AdvancedRestClient (ARC) or a front-end client.