Project Name: Dopest Dish

Members:

Sarah Wadley smw16c

Alec Amico aza16

Michael Tafuri mat12j

**Introduction:**

What is your application?

Our application is a restaurant dish review application that allows users to review single dishes at restaurants in contrast to reviewing the restaurant as a whole.

Why do you want to choose this application, instead of others, as the course project (motivation)?

We chose this application instead of others because we love food. Also, we recognized the existence of apps like “Yelp” or “Zomato” which provide rankings for the restaurant as a whole. The importance of an individual dish or food item is lost with these rankings. For example, a wings restaurant may also have good burgers. Differentiating between the individual items is a factor that is overlooked.

What are the key components in your project?

The key components in this project are being able to review dishes from various restaurants with a star rating out of 5 as well as a text description of why the user decided to give that rating.

**Database details:**

How do you design the database?

We designed out database with four main entities. We have users, reviews, menu items, as well as restaurants. In our database, many reviews can come from one user. Also, many reviews can exist for one menu item. A restaurant can sell many items from the menu. Because menu items may have similar names at different restaurants, it is a weak entity with the foreign key set as the restaurant ID because one restaurant will not have two different menu items with the same exact name.

What is your E-R model and relational model?

Diagram

Description automatically generated

Table

Description automatically generated

What are the tables you include in the database?

CREATE TABLE "dishes\_dish" (

"id" integer NOT NULL,

"title" varchar(50) NOT NULL,

"price" decimal NOT NULL,

"description" text NOT NULL,

"numReviews" integer NOT NULL,

"resID\_id" varchar(50) NOT NULL,

PRIMARY KEY("id" AUTOINCREMENT),

CONSTRAINT "menuItem" UNIQUE("title","resID\_id"),

FOREIGN KEY("resID\_id") REFERENCES "restaurants\_restaurant"("name") DEFERRABLE INITIALLY DEFERRED

);

CREATE TABLE "reviews\_review" (

"id" integer NOT NULL,

"review\_text" text NOT NULL,

"date\_posted" datetime NOT NULL,

"author\_id" integer NOT NULL,

"dish\_id" integer NOT NULL,

"restaurant\_id" varchar(50) NOT NULL,

"rating" integer unsigned NOT NULL CHECK("rating" >= 0),

PRIMARY KEY("id" AUTOINCREMENT),

FOREIGN KEY("dish\_id") REFERENCES "dishes\_dish"("id") DEFERRABLE INITIALLY DEFERRED,

FOREIGN KEY("author\_id") REFERENCES "auth\_user"("id") DEFERRABLE INITIALLY DEFERRED,

FOREIGN KEY("restaurant\_id") REFERENCES "restaurants\_restaurant"("name") DEFERRABLE INITIALLY DEFERRED

);

CREATE TABLE "restaurants\_restaurant" (

"name" varchar(50) NOT NULL,

"totalReviews" integer NOT NULL,

"cuisine" varchar(2) NOT NULL,

PRIMARY KEY("name")

);

CREATE TABLE "profiles\_profile" (

"id" integer NOT NULL,

"image" varchar(100) NOT NULL,

"user\_id" integer NOT NULL UNIQUE,

PRIMARY KEY("id" AUTOINCREMENT),

FOREIGN KEY("user\_id") REFERENCES "auth\_user"("id") DEFERRABLE INITIALLY DEFERRED

);

CREATE TABLE "locations\_location" (

"address" varchar(50) NOT NULL,

"numReviews" integer NOT NULL,

"name\_id" varchar(50) NOT NULL,

PRIMARY KEY("address"),

CONSTRAINT "uniqueLocation" UNIQUE("name\_id","address"),

FOREIGN KEY("name\_id") REFERENCES "restaurants\_restaurant"("name") DEFERRABLE INITIALLY DEFERRED

);

How do you design your database to accommodate functional dependencies?

**Dishes**

id -> title, price, description, numReviews, resID

**Reviews**

id -> review\_text, date\_posted, author\_id, dish\_id, restaurant\_id, rating

**Restauarants**

name -> totalReviews, cuisine

**Locations**

address -> numReviews, name\_id

**Profiles**

id -> image, user\_id

Are your tables in BCNF or 3NF? How about other constraints?

Our tables are in BCNF because every key is a super key of the table. For example, our dish ID attribute can retrieve each piece of information for the dish: the title, price, description, numReviews, and resID.

**Functionality details:**

What are the basic functions and advanced functions in your database and how do you implement them?

The basic functions are adding a review to a currently existing dish, adding a dish to the database, and adding a restaurant. Users also have the ability to create an account and add a profile picture.

**Implementation details:**

What languages and platform you've chosen to do the implementation?

Python 3

HTML5

SQLite

How do you implement the front-end Web interface and the application logic?

The front-end web interface is implemented with HTML and the application logic is implemented with Python. Python modules are also used to enhance the user experience. We use the module Pillow as our imaging workhorse. We use Django’s Crispy Forms for form rendering and retrieving entered user information.

How does the front-end Web interface interact with the backend database?

The front-end web interface interacts through the Django framework. The HTML web interface passes information to the Python functions which then query our database for the desired results. The query is passed back to our functions in Python and then to HTML to be presented to the user.

Students are encouraged to upload their code to some online repositories, such as  
Dropbox, Github, Bitbucket, and provide the link in the report.

https://github.com/lifeunsubscribe/dopestDish

**Experiences:**

What have you learnt from this project?

Designing a database system with many parts can be challenging. Working in teams during a pandemic is difficult when there is no prior training. As undergraduate students, many of our projects before this past year, we have been able to meet face to face with our respective teams and this allows the team to always be in sync. Another thing that we learned was that companies like Yelp do not have their APIs freely available. We were not able to link their API with our database but that does not change the functionality of our system. We were mainly going to utilize Yelp’s API to populate the restaurant information as well as the menu items. To solve this issue, restaurant information and menu items are user submittable.

How have you solved hard problems in this project?

Most of our hard problems have been solved through re-watching lectures and reviewing the slides from lectures. Also, YouTube videos have been quintessential for describing how to set up our Django server and writing code in HTML for our less web development-savvy team member.

How to extend your project to more advanced, mature systems in the future?

One way we could extend our project is to create a mobile application for Android and iOS. Hopefully in the near future, we will be able to populate restaurants again and having our application and database of reviewed menu items can help make a user’s dining experience easier and more enjoyable. Another way we can extend our project is by paying for the Yelp API so that user’s do not have to enter the dish information themselves. We believe that this will make the system more accessible to everyone and will allow our functions within this project to have a much greater scope.  
  
**References:**

Any resource you have turned to for help during the implementation of your project.

https://www.youtube.com/watch?v=F5mRW0jo-U4

https://www.youtube.com/watch?v=UmljXZIypDc

https://www.youtube.com/watch?v=UB1O30fR-EE