**Project Design**

**Brand name:**

**TERPCravingz —Find your Terp Eatery Here**

**Business Scenario:**

TERPCravingz is a medium sized restaurant review platform that showcases restaurant information, restaurant review information, and UMD Shuttles to the restaurant of choice.

Recently, TERPCravingz started providing consultancy services to budding entrepreneurs looking to open their own restaurant near the UMD campus. These entrepreneurs looking for restaurant business opportunities nearby University of Maryland (UMD) need analysis insights for their decision making, and customers who want to find their best restaurants to visit. In addition, customers can get information about the UMD shuttles passing by the restaurant and the nearest shuttle stop to the restaurant that they want to visit. Apart from analytical and travel-searching functions, the ordinary queries are available on TerpCravingz, so that customers can look up on the restaurant review platform for reviews of restaurants and other information to which TERPcravingz offers the access for all visitors.

Generally, our entrepreneur customers have the following needs:

(1)Easy connectivity

(2)Neither too cluttered nor too deserted with neighbouring restaurants (3) Well-received, well-known and affordable restaurant themes (Eg: Breweries, Bars, Pool houses,etc)

We, the Analytical team of TERPCravingz, provide our entrepreneur clients with these business insights to guide them better about the location and restaurant theme of their anticipated restaurant business.

**Business processes/transactions:**

● There are various restaurants listed on the company’s website ( 2793 restaurants in 15 areas nearby UMD: College Park, East Riverdale, Hyattsville, Langley Park, Mount Rainier, Chillum, New Carrollton, Greenbelt, Hillandale, Takoma Park, Beltsville, Silver Spring, White Oak, Glenn Dale and Fairland). Each restaurant is described by a unique identifier, name, address (including street, city, state and postal code), phone number, location coordinates (described by the latitude and longitude in decimal format), average rating (shown as the number of stars from 0 to 5), the total number of its reviews in the database, the expensiveness of the restaurant (on a scale of 1 to 5) and whether or not it provides delivery, pick-up or restaurant reservation service (labeled as resTransactionType). Moreover, the schedule of the time when the restaurant opens and is closed from Monday to Sunday is also stored.

● Foods that a restaurant sells can fall into various categories (Mexican, Italian, etc). Each category is assigned a unique identifier. A category of food can be served by multiple restaurants.

● Each customer has a unique customer identifier and a customer name.

* A review is stored by its unique identifier, the rating (the number of stars) that a customer assesses that restaurant, the review date, and the text content of the review that they give to that particular restaurant for each visit. A customer can review for more than one restaurant. A restaurant can receive more than one review.
* Apart from restaurant information, there are 25 UMD shuttles that can take customers go to their chosen restaurants. Each shuttle have at least one station on their scheduled routes. The UMD shuttle is identified by a unique identifier, bus number, its name and the final destination.
* All bus stops where a shuttle stops for passengers to board or alight from it are saved in the database. However, each restaurant has only one nearest bus stop to itself. Restaurants can share the same nearest stop or have their own nearest stops. Shuttle stops are recorded by unique stop identifiers, names and corresponding location coordinates (latitude and longitude).

## **Entities, Attributes and Primary Keys**

* Restaurant(**resId**, resName, resAddress, -resStreet, -resCity, -resState, -resPostalCode, resPhoneNo, resLocation, -resLat, -resLong, resStars, resRevCNT, resPrice, resTransactionType, resOpenHour, -monStartTime, -monEndTime, -tuesStartTime, -tuesEndTime, -wedStartTime, -wedEndTime, -thurStartTime, -thurEndTime, -friStartTime, -friEndTime, -satStartTime, -satEndTime, -sunStartTime, -sunEndTime)
* Category(**categoryId**, categoryName)
* Customer**(cusId**, cusName)
* Shuttle(**busId**, busNo, busName, busFinalDest)
* Stop(**stopId**, stopName, stopLocation, -stopLat, -stopLong)

## **Relationships, Attributes, Degrees, Participating Entities and Constraints**

* Have: binary relationship:

1 shuttle to 1 or more stops

1 stop to 1 or more shuttles

* Belong: binary relationship:

1 restaurant to 1 or more categories

1 category to 0 or more restaurants

* Be nearest(walkTimeToRes): binary relationship

1 restaurant to 1 stop

1 stop to 0 or more restaurants

* Review(**revId**, revRating, revDate, revText): binary relationship as associative entity

1 restaurant to 0 or more customers

1 customer to 0 or more restaurants

A screenshot of a cell phone

Description automatically generated

## 

## **Project Proposal**

Team 0504-06

## **1. Relations:**

● Restaurant(**resId**, resName, resStreet, resCity, resState, resPostalCode,

resPhoneNo, resLat, resLong, resStars, resRevCNT, resPrice, resTransactionType, monStartTime, monEndTime, tuesStartTime, tuesEndTime, wedStartTime, wedEndTime, thurStartTime, thurEndTime, friStartTime, friEndTime, satStartTime, satEndTime, sunStartTime, sunEndTime, *stopId,* walktimeToRes)

● Category(**categoryId**, categoryName)

● Customer**(cusId**, cusName)

● Review(**revId**, revRating, revDate, revText, *cusId*, *resId*)

● Shuttle(**busId**, busNo, busName, busFinalDest)

● Stop(**stopId**, stopName, stopLat, stopLong)

● Belong(***resId***, ***categoryId***)

● Have(***busId***, ***stopId***)

## **2. Functional Dependency:**

● resId → resName, resStreet, resCity, resState, resPostalCode, resPhoneNo, resLat, resLong, resStars, resRevCNT, resPrice, resTransactionType, monStartTime, monEndTime, tuesStartTime, tuesEndTime, wedStartTime, wedEndTime, thurStartTime, thurEndTime, friStartTime, friEndTime, satStartTime, satEndTime, sunStartTime, sunEndTime, stopId, walktimeToRes

● categoryId → categoryName

● cusId → cusName

● revId → revRating, revDate, revText, cusId, resId

● busId → busNo, busName, busFinalDest

● stopId → stopName, stopLat, stopLong

● resId, categoryId →

● busId, stopId →

## **3. Normalization:**

● Restaurant(**resId**, resName, resStreet, resCity, resState, resPostalCode, resPhoneNo, resLat, resLong, resStars, resRevCNT, resPrice, resTransactionType, monStartTime, monEndTime, tuesStartTime, tuesEndTime, wedStartTime, wedEndTime, thurStartTime, thurEndTime, friStartTime, friEndTime, satStartTime, satEndTime, sunStartTime, sunEndTime, *stopId*, walktimeToRes) = 3NF

● Have(***busId***, ***stopId***) = 3NF

● Category(**categoryId**, categoryName) = 3NF

● Customer(**cusId**, cusName) = 3NF

● Review(**revId**, revRating, revDate, revText, *cusId*, *resId*) = 3NF

● Shuttle(**busId**, busNo, busName, busFinalDest) = 3NF

● Stop(**stopId**, stopName, stopLat, stopLong) = 3NF

● Belong(***resId***, ***categoryId***) = 3NF

## **4. Business rules:**

[R1] When a stop is deleted from the database, restaurants under its coverage and the walktime between restaurants and corresponding nearest stops should not be deleted in the database.

[R2] When a stop is changed in the database, restaurants corresponding to the nearest stop should not be updated accordingly.

[R3] When a stop is deleted from the database, the corresponding information in the shuttle should also be deleted.

[R4] When a stop is updated in the database, the corresponding information in the shuttle should be updated accordingly.

[R5] When a shuttle stops running and is deleted from the database, the corresponding stop information should be deleted.

[R6] When a shuttle line is changed in the database, the corresponding stop information should be updated accordingly.

[R7] When a restaurant is no longer in the database, the corresponding category information should be deleted from the database.

[R8]When a restaurant changes information in the database, the corresponding category to the restaurant should be changed accordingly.

[R9] When a category is deleted from the database, the corresponding restaurant should be deleted in the database.

[R10] When a category of restaurants is changed in the database, the corresponding restaurant information should be changed accordingly.

[R11] When a customer is deleted from the database, reviews that he posted online should be deleted in the database.

[R12] When a customer changes his information in the database, his review information should be changed accordingly.

[R13] When a restaurant is deleted from the database, reviews which customers wrote for it should be deleted in the database.

[R14] When a restaurant is changed in the database, reviews which customers wrote for it should not be changed accordingly.

## **5. Referential integrity:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Relation | Foreign Key | Base Relation | Primary Key | Business Rule | Constraint: ON DELETE | Business Rule | Constraint: ON UPDATE |
| Restaurant | stopId | Stop | stopId | R1 | NO ACTION | R2 | NO ACTION |
| Have | stopId | Stop | stopId | R3 | CASCADE | R4 | CASCADE |
| Have | busId | Shuttle | busId | R5 | CASCADE | R6 | CASCADE |
| Belong | resId | Restaurant | resId | R7 | CASCADE | R8 | CASCADE |
| Belong | categoryId | Category | categoryId | R9 | CASCADE | R10 | CASCADE |
| Review | cusId | Customer | cusId | R11 | CASCADE | R12 | CASCADE |
| Review | resId | Restaurant | resId | R13 | CASCADE | R14 | NO ACTION |

## **6. Describe Sample Data:**

Every table in the database has been described using a few rows of the dataset.

● **Restaurant**

Text Example:

Restaurant('WJKA3MTOMEFcPCSMaytLVw','China Golden','9127 Riggs Rd','Adelphi','MD',20783,'(301) 434-5928',39.003739,-76.970217,3,17,1,'delivery','11:00:00','23:00:00','11:00:00','23:00:00','11:00:00','23:00:00','11:00:00','23:00:00','11:00:00','23:30:00','11:00:00','23:30:00','12:00:00','22:00:00','34650',2.189471981)

**● Have**

Text Example :

Have('BID108','34650')

●  **Category**

Text Example:

Category('1028', 'Chinese')

● **Customer**

Text Example:

Customer('owl1WlncqhwJ\_TquzAKkpA', 'Karl S.')

● **Review**

Text Example:

Review('R76RNeUt\_A4hu-wQDn\_aAg', 5, '2019-06-05 14:02:14','Hi, this review is for the takeout service at China Golden. They are always pleasant and the food is good. Their prices are reasonable. Thank you!', 'owl1WlncqhwJ\_TquzAKkpA', 'WJKA3MTOMEFcPCSMaytLVw')

● **Shuttle**

Text Example:

Shuttle('BID108', '108', '108 Adelphi', 'Regents Drive Garage')

● **Stop**

Text Example:

Stop('34650', 'Metzerott Rd and Riggs Rd (Outbound)', 39.0032699000000, -76.9722580000000)

● **Belong**

Text Example:

Belong('WJKA3MTOMEFcPCSMaytLVw','1028')