Restaurant Guide

# General Concept

As described in our first hand-in for this project, we wanted to create somewhat of a typical restaurant-database application but with a focus on local and small businesses and emphasis on user-interaction.  
The application allows for all kind of restaurants and food-stalls to be entered. Businesses can add categories to their information in order to be easier to find and to allow recommendations to be given to the users.   
Furthermore, they can add a description, location and opening hours to their data, which will then be displayed on the detail page of that business.

Users can rate and review businesses; reviews are visible to all other users and the average rating of all reviews makes up the official rating of the business.   
Users can add other users as their friends to check out all of their reviews. Since the application aims to focus on a local circle of businesses, a lot of interaction is expected between the users.

# Database Schema

**user** (user\_id, name, review\_count, member\_since, average\_stars);

**business** (business\_id, naname, adress, city, state, postal\_code, stars, reviews, is\_open, description, hours)

**review** (review\_id, user\_id, business\_id, stars, date, text); *Foreign Keys:*user\_id from user(user\_id); business\_id from business(business\_id);

**categories** (business\_id, category); *Foreign Keys:*business\_id from business(business\_id);

**friends** (user\_id, friend); *Foreign Keys:*user\_id from user(user\_id); friend from user(user\_id)

The user and business tables hold basic information about every user that has signed up and any business that was created within the application. They are connected through the review table, which links a user to a business via a review they wrote about it. Lastly we created the categories and friends tables. The information stored in them used to be part of the business and user tables respectively, however, the transfer of this information into separate tables allows us to work with atomic values.

# Data acquisition

Disclaimer: Unfortunately, due to the current circumstances and the shortened time we were not able to collect the data as planned. Our original method is explained below.  
First, all restaurants in the vicinity of Mahidol University and the dormitories are registered by the team by hand.   
Then we ask our friends and students how they find the restaurants we have selected and collect reviews. This data will be digitalized by us. In this step we also create user profiles for the participants of the survey.  
Before a student can participate, he or she will of course be informed about his or her rights and data protection. As this is personal data, data protection is an important part of our application.

Due to the shortened project period, we have decided to collect restaurant data as described above and in a realistic way. All other data is computer generated or imaginary.

# Queries

We tried to put a lot of thought into what kind of functionalities we would require from this kind of application.   
Of course, if this was a large-scale project, the user would want to be able to look for restaurants with almost any combination of filters. However, due to the limited time, we had to boil it down to some basic functions.

**Top Businesses**

The most obvious functionality we want to offer to the user is to look for the ‘best’ businesses.   
To do this, the user will be shown the businesses with the highest star-ratings. Since there may be a big number of businesses with a similar rating, the user can choose himself how many he wants to be shown.   
In the finished application, sorting by the rating would be included in almost any other filter-option automatically.

**All Reviews (of a business)**

This would most likely be used on the detail-page of one single business. The reviews are what the star-rating is based on, so it’s important that when a user is considering to visit a certain business, he can check out their page and get insights about what other customers thought of it.   
Of course, he should also be able to see who a specific comment was left by.

**Business by category**

The category-system is an easy way for the user to find businesses he might be interested in visiting.   
He can enter a category he likes (for example “international”, “cheap”, “healthy”, etc.) and only have businesses recommended to him that belong to that category.  
In future versions of the application, we would expand the function by letting the user filter by many categories at once.

**View all friends (of a user)**

This function would be implemented in the application in the logged-in user’s profile page, allowing them to see all of their friends in one list. It makes sense to offer the users a possibility to connect with their friends so that they can view the establishments frequented by them, as well as check out reviews left by them.

**Favorite categories**

This function aims to find out the user’s four favorite categories by checking all the businesses he has rated well (4 or 5 stars) and their respective categories, then counting which 4 of the categories where present the most in all of those businesses.  
In the final version, this function would be further expanded into a recommendation algorithm to suggest new businesses to the user.

***Recommendation Algorithm***Initially intended to be one of the main features (and the most complex one) of our application, we were sadly forced to leave it out due to the shortage of time.   
The algorithm would have considered different factors such as the user’s favorite categories, other businesses the user enjoyed and perhaps which businesses his friends or users with a similar taste enjoyed the most, to try and find the “ideal” business for the user to visit.

# Usage

As there was no time to prepare a proper interface, the queries can be used via a simple console interface upon executing *tables.py* after creating an appropriate database.