**Robert Tung, Kristina Shia, Sean Gao, James Ma**

**CPSC 437 Final Project Report**

**Yelp++: Food Recommendations for You and Your Friends**

**Project Summary**

Our team built a prototype application to help users decide what to eat. This application leverages publicly available Yelp data, along with custom user profiles, to provide recommendations tailored to a user’s tastes. By utilizing this data in tandem with machine learning algorithms, we provide a data-driven dining experience. In addition, we differentiate ourselves from Yelp and other existing services by incorporating a social network aspect to our recommendations.

**Stack:** Python, R, SQLite

**Github:** https://github.com/rtung1333/CPSC-437-Final-Project

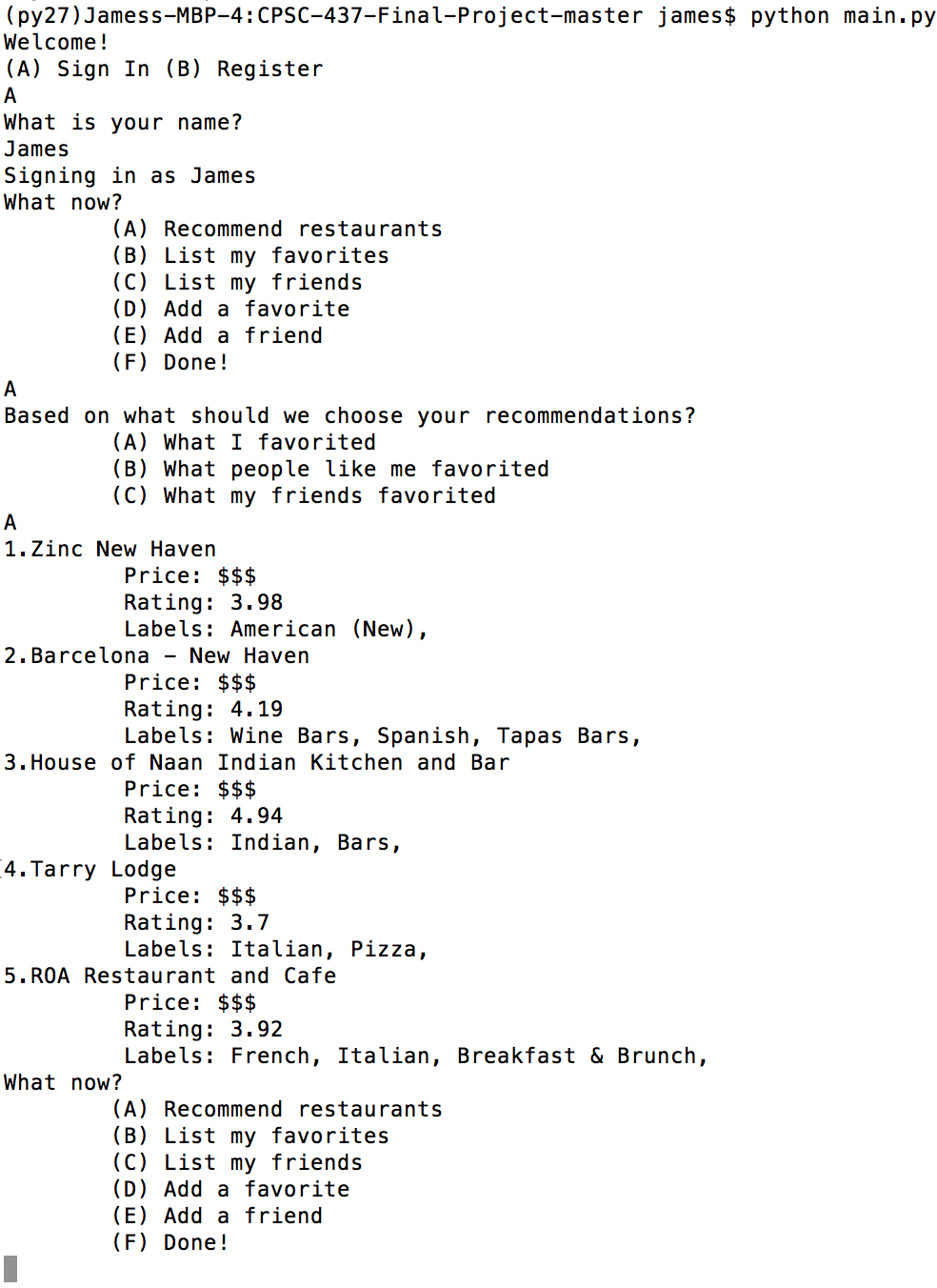
**Work split**

* Whole team
  + Coming up with database schema (relations, attributes, keys)
  + Overall project idea and direction
* Robert
  + Wrote all of computations.py
  + Recommend based on personal favorites, recommend favorites of friends, and recommend favorites of people of same age and gender
    - Full debugging of when the main uses computations and tables in config
  + A small amount of debugging and refactoring in main.py (Kristina wrote almost all of it)
  + Created some of the tables in config.py (following Kristina’s lead after she set up all that other code)
* Kristina
  + Wrote command line interface
  + Wrote database queries for outputting recommendations and user friends/favorites
  + Wrote user signin and register process
  + Wrote database queries to add friends and favorites
  + Wrote config script to initialize database and create tables
* Sean
  + Wrote classes to pull Yelp restaurant data from HTML (later scrapped for API)
  + Wrote functions to scrape all user reviews for each restaurant from HTML
  + Wrote script to perform PCA and fit attribute weights onto data to output user-specific restaurant recommendations
  + Wrote script to randomly generate and populate Users, Friends, and Favorites
* James
  + Wrote scripts to pull Yelp restaurant data from the Yelp v2 and v3 Fusion APIs
  + Cleaned and manipulated JSON data from API into intended table format
  + Wrote scripts to create a SQLite DB and insert tables into the database
  + Wrote template code for using MySQL before switching to SQLite
  + Write-up

**Functionality**

Our application is intuitive to use. It’s as simple as answering the prompts until you’re ready.

Example:



In this example, the user signed in and asked for a personalized recommendation. You can:

* Sign In or Register
* Ask for a personalized recommendation
* Ask for a recommendation based on your friends
* Ask for a recommendation based on people in the same gender and age group
* Add friends or favorites

**Limitations**

* Cannot provide recommendations if a user does not have any favorites, friends, and there are no other users in their age and gender group
* Favorites must be spelled correctly

**Main issues and challenges**

* Parsing through raw HTML for data scraping was too cumbersome and slow
  + Pivoted to use the Yelp API
* Incorporating data (ratings) that was not accessible by just the Yelp API scrape
* Intelligent similarity measure between restaurants and how to weight factors (PCA)
* Pulling multiple recommendations required tracking current sets of best restaurants as the computations ran

**Main contributions of our system**

* Unlike Yelp:
  + Our application provides specific recommendations that are informed by user preferences
  + We allow the user to “friend” other users and incorporate their preferences into our recommendation algorithms
  + We incorporate user demographics into our recommendations (age, gender)

**Selling Points**

* We allow users to obtain recommendations based on specific inputted preferences
* Peer-to-peer interactions / results -- We also incorporate allow users to obtain recommendations based on social factors:
  + Friend groups
  + Age and gender (easily extended to other personalized factors if those were added to database)
* Our application prototype is easy to set up and use
* Peer-to-peer functionalities open up a lot of possible future directions
  + Recommendations for a group of people
  + Aggregate statistics of preferences across demographic groups
  + Aggregate a user’s recommendations as a combination of own, friends’ and demographics’ by using user feedback to recommendations
* Our database design is BCNF

