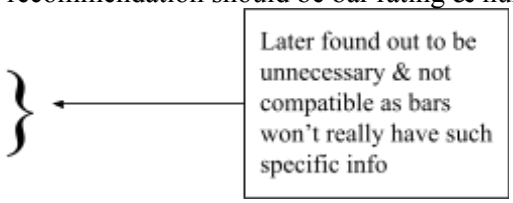
Project 4: Clubbing in times of COVID

Group Members: Shawn, Mustafa, Anuj, Ernest, Ankita

Sprint 1

Requirements Engineering

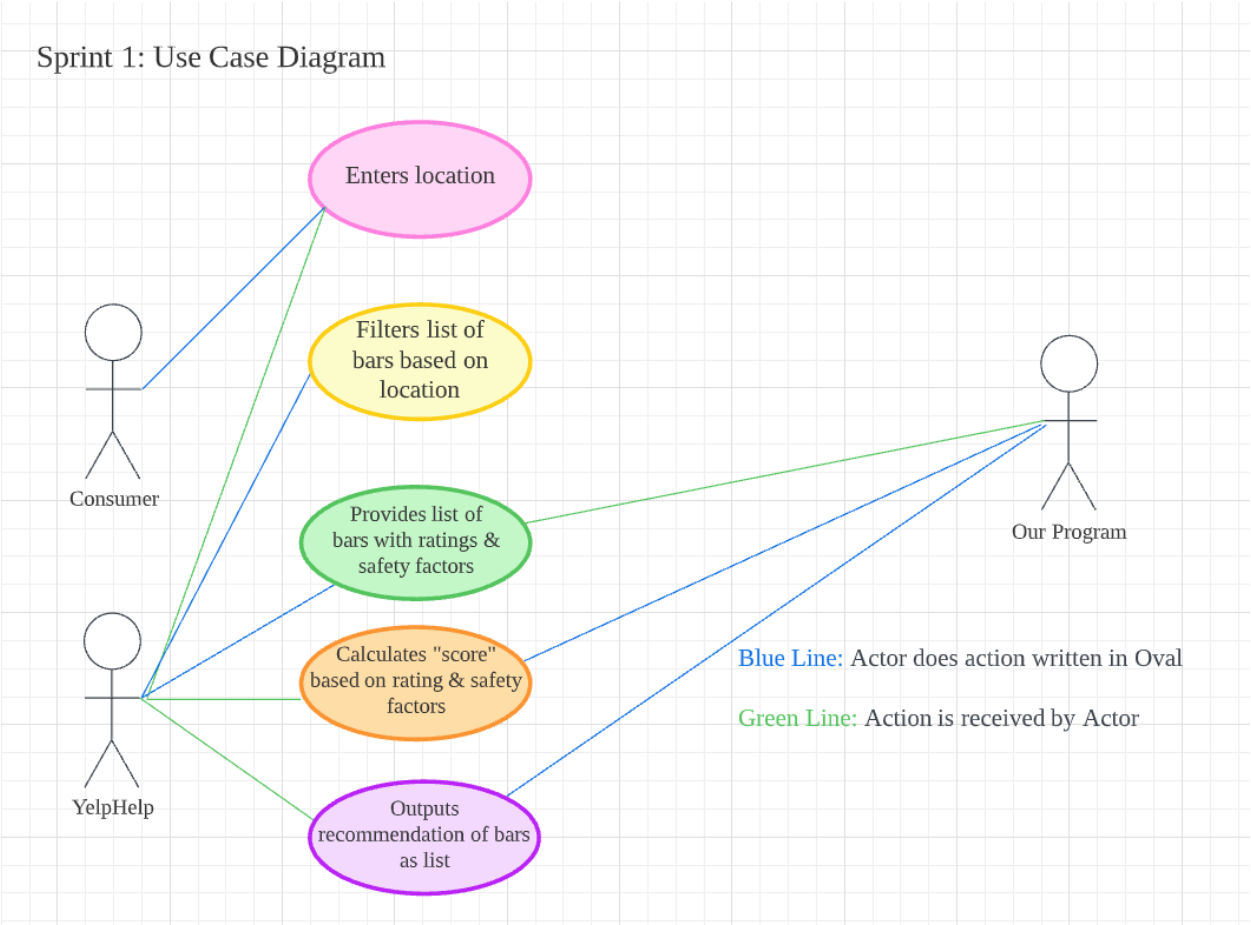
- **Functional Requirements**
  - The factors that decide the recommendation should be bar rating & number of people at the bar.
  - Ratings - consumer side
    - Atmosphere
    - Menu
  - Occupancy
    - Current
    - Maximum
  - Bars should be able to tell the count of people at a given moment
  - Way to balance the ratings/occupancy (safe-to-cool ratio)



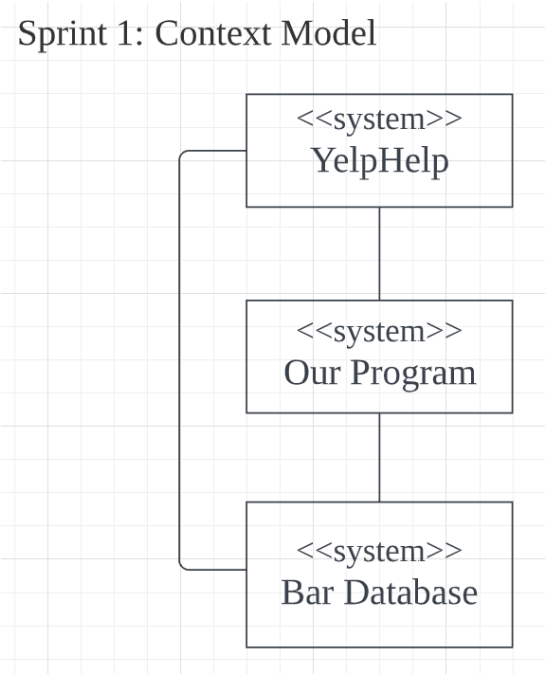
- **Non-Functional Requirements**
  - It is expected that our program gets no input from the YelpHelp user or any bars. The sole interaction is with YelpHelp.
  - Our program is built according to the assumption that YelpHelp will be able to provide occupancy statistics of each bar in its database.
- **Usability Requirements**
  - It is assumed that our program is responsible for giving a score to each bar which is out of 100. YelpHelp will not expect any further information.
  - The YelpHelp user also should be able to make a choice based on the scores out of 100. No threshold is provided which specifically makes a bar recommendable or not (ie. if the recommendation score is above 50%, then it is recommended).
  - Should bars be able to use this ranking to see how they perform and show this service on their own page?
  - Should users of this service be able to input their own recommendations or ratings?
  - Will users be able to rate the quality of the service itself?
- **Requirements Elicitation**
  - It is not possible or necessary to gather & utilize actual customer data (specific expectations of users using YelpHelp) as there is to be no interaction between our program and the user.
  - It is expected that YelpHelp will provide our program with only the relevant bars, that is, a list of bars that have been filtered out based on the user input of desired distance/location on YelpHelp.

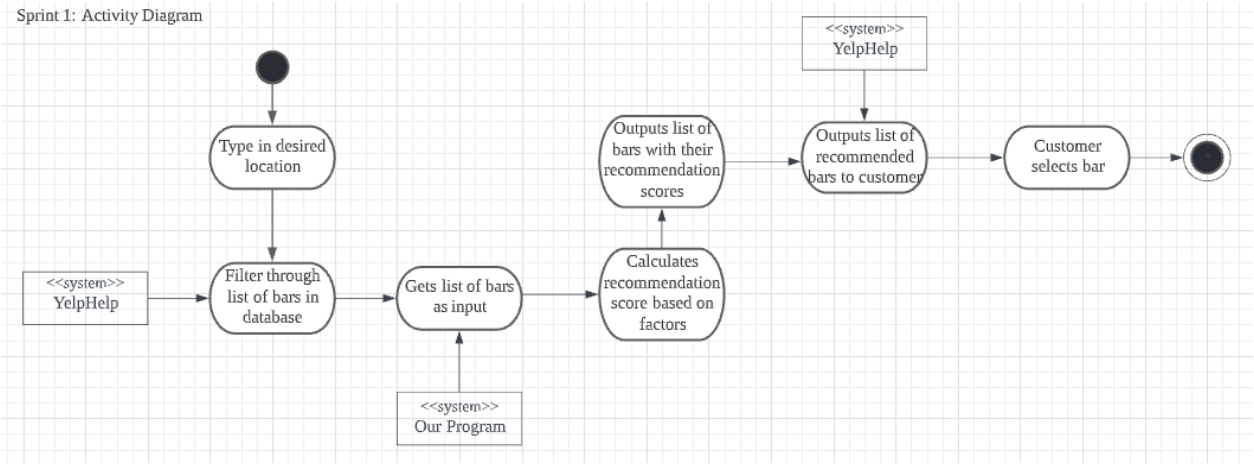
System Modeling

- Input list of bars with ratings and occupancy factors
- Output list of bars ranked by algorithm
- It is assumed that there is no interaction between the customer and our program. Our program's only interaction is with YelpHelp which interacts with the user.
    - It is assumed that YelpHelp will get an input by the user of the desired location and that YelpHelp will then filter through all the bars in its database to only give our program the ones that are within the inputted distance/location.
- It is assumed that YelpHelp will store each bar's information: its official rating (on its personal website or through YelpHelp), its maximum occupancy, and its current occupancy.

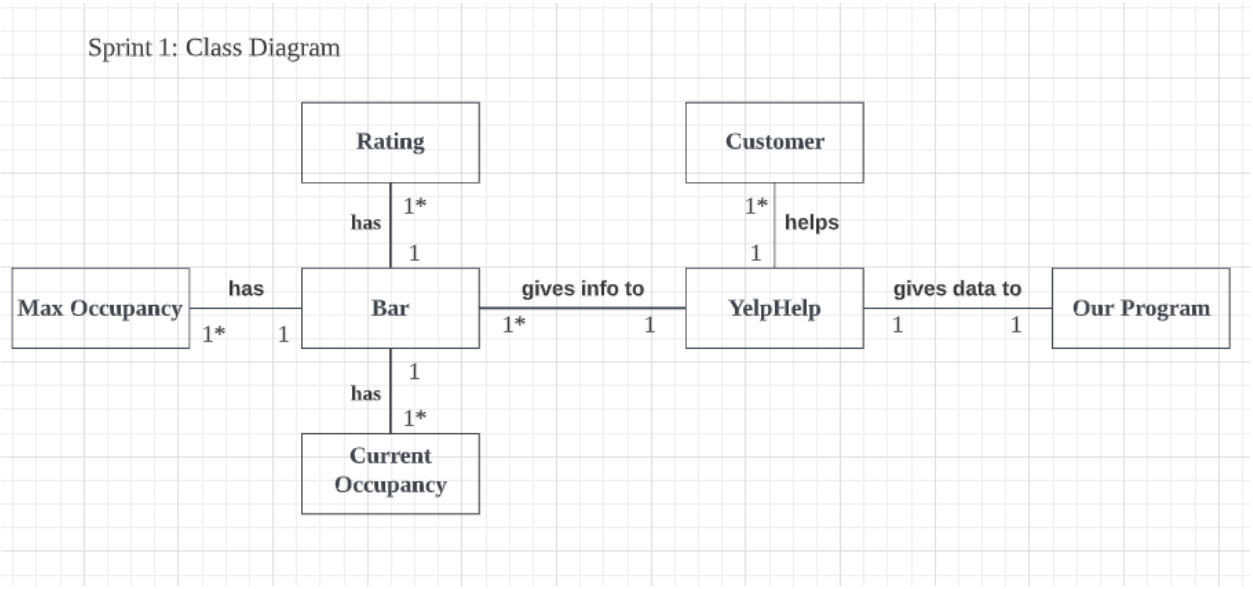


Sprint 1: Context Model

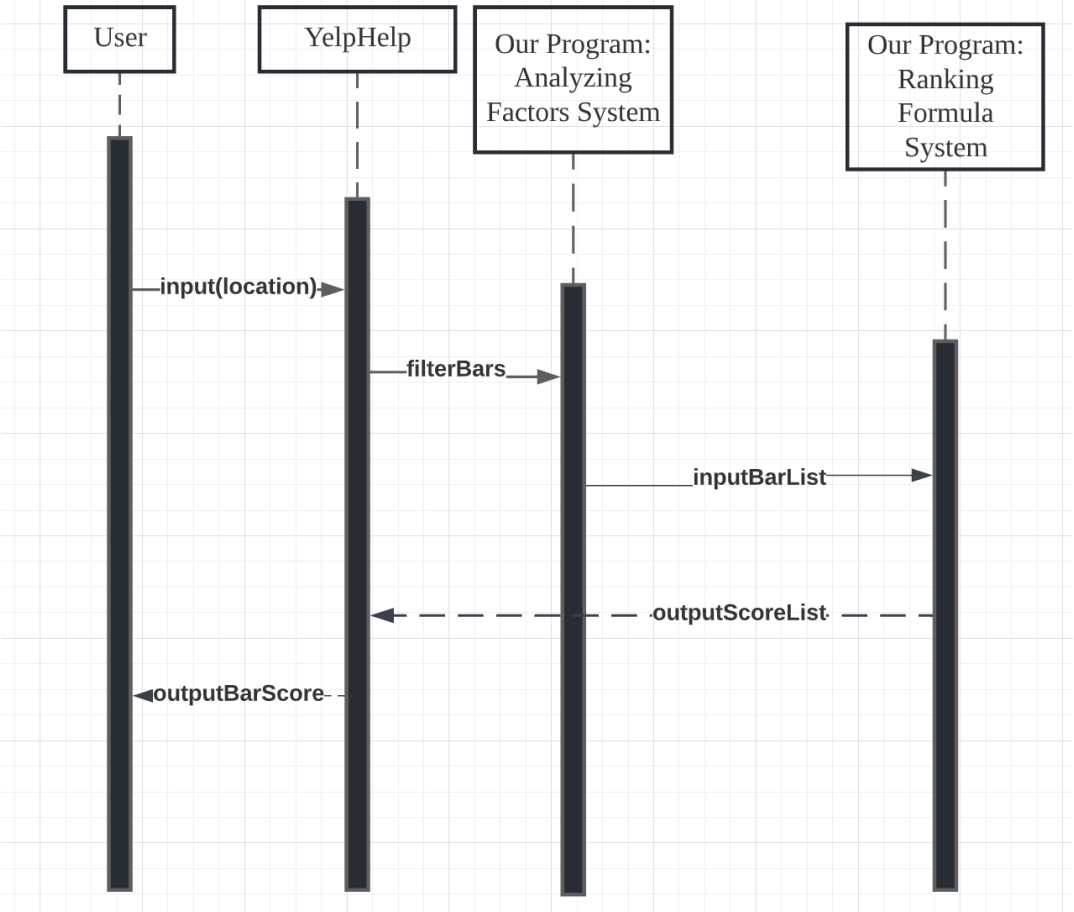




Sprint 1: Class Diagram

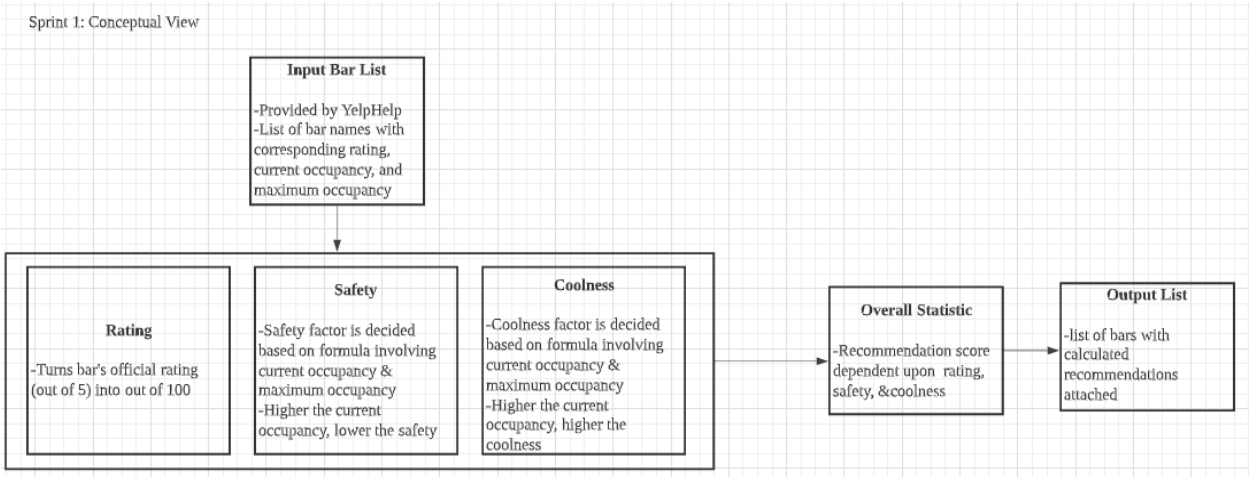


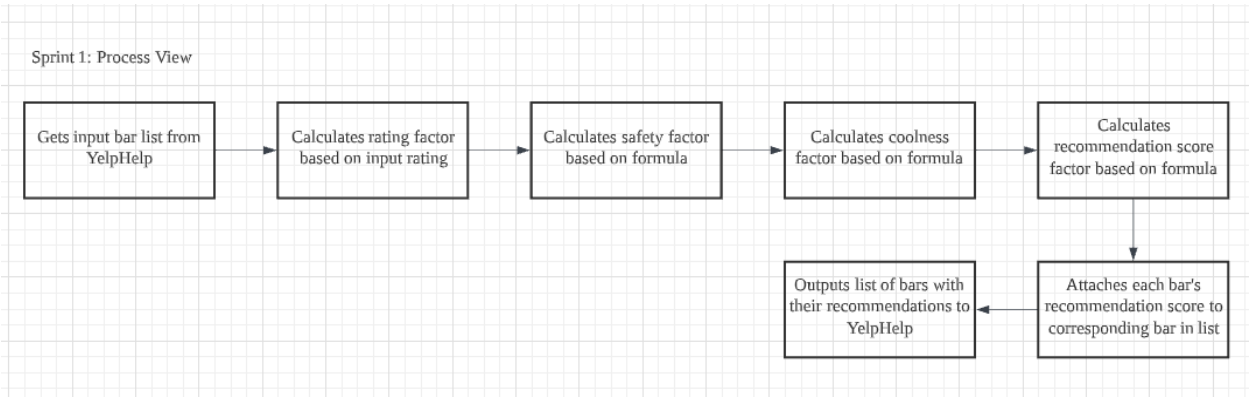
Sprint 1: Sequence Diagram



Architecture

- Test-first development
    - Create test cases for the data set or use public data sets (bars? restaurants?)
- Regarding the formula, it is assumed that the input rating parameter is out of 5. Safety and coolness are both out of 100 and equally balance out the score.





Design & Implementation

Most of us know Java however, Python is used for data management & helps with designing code specific to the problem so Python is a better choice. Also, Java algorithms can be easily shifted to Python.

The calculations in Sprint 1 were brainstormed as a group, given that we would take 3 inputs: rating, current occupancy and maximum occupancy. The safety is calculated as (1-current occupancy/max occupancy)*100 and coolness is calculated as current occupancy/max occupancy * 100. The output is (safety + coolness)/2 (an average of the two factors).

Based on the algorithm, it can be seen that the number of people in the bar goes two ways, both positively and negatively. If the current occupancy is close to the maximum occupancy, then the safety factor goes low as there might be too many people in the bar. However, this also shows that the bar is popular which implies that it has a high “coolness”. However, here, coolness is also measured by the bar’s official rating which is assumed to be out of 5.

Test Cases: Input Parameters - Rating, Current Occupants, Total Occupants

In order to get a more suitable test case size, we used uniform random test data– a rating generated between 1-5, a random current occupancy from 1 to the total occupancy, and a random total occupancy from 100 to 1000.

One issue is that it may not reflect real world data (tends to be normal biased towards top (survival effect)). However, this allows for a variety of testing so that it can be seen whether the implemented formulas work, so it’s not a problem even if the numbers are not specifically realistic.

[[4.46, 323, 529], [2.01, 344, 757], [4.29, 46, 196], [3.56, 570, 836], [2.01, 892, 941], [2.4, 43, 312], [1.86, 159, 160], [3.25, 925, 995], [2.53, 475, 520], [3.2, 209, 807], [4.85, 553, 887], [2.43, 73, 495], [4.26, 5, 544], [4.25, 267, 657], [3.0, 19, 395], [1.32, 265, 841], [4.19, 63, 290], [4.27, 194, 818], [2.84, 445, 834], [3.84, 379, 776], [1.41, 229, 683], [1.55, 614, 670], [3.89, 809, 913], [2.9, 547, 803], [4.53, 51, 714], [3.24, 82, 130], [1.53, 441, 705], [4.86, 42, 207], [4.97, 144, 984], [3.76, 181, 563], [3.79, 187, 194], [4.25, 23, 669], [3.7, 96, 110], [4.83, 267, 851], [4.7, 434, 891], [3.97, 59, 313], [3.62, 152, 842], [2.7, 167, 257], [2.12, 465, 535], [2.49, 328, 682], [2.92, 216, 989], [4.73, 274, 540], [1.09, 340, 597], [3.01, 98, 217], [3.72, 197, 277], [1.63, 867, 981], [2.62, 503, 919], [2.4, 150, 567], [3.44, 121, 416], [4.62, 499, 956], [3.04, 479, 561], [1.17, 73, 425], [4.09, 137, 643], [2.91, 69, 103], [2.39, 112, 199], [2.45, 77, 108], [2.92, 129, 999], [3.06, 128, 374], [4.18, 76, 125], [4.59, 185, 514], [3.65, 339, 369], [4.52, 682, 850], [4.38, 304, 449], [3.11, 162, 504], [1.72, 773, 786], [2.92, 406, 851], [2.93, 111, 383], [1.16, 813, 861], [3.61, 518, 808], [4.68, 664, 852], [4.01, 275, 831], [2.84, 461, 843], [4.4, 93, 514], [3.25, 490, 800], [2.9, 409, 864], [4.88, 17, 535], [3.33, 748, 963], [3.97, 110, 193], [4.74, 429, 529], [4.0, 260, 422], [2.49, 177, 225], [3.39, 136, 452], [2.17, 636, 931], [1.4, 371, 421], [1.2, 485, 575], [2.16, 449, 904], [1.57, 933, 977], [3.48, 197, 586], [3.79, 117, 780], [1.99, 150, 371], [3.71, 51, 638], [1.48, 543, 860], [1.2, 176, 558], [4.65, 608, 974], [1.62, 151, 413], [2.03, 405, 507], [3.72, 119, 159], [2.99, 484, 526], [3.11, 22, 415]]

Updated Test Cases

The test cases were updated with a normal distribution. Bar names were also added for some authenticity.

[['GIARDINO RISTORANTE', 1.42, 121, 667], ['ART BAR', 2.41, 74, 523], ['CAFE 212/COLUMBIA CATERING KITCHEN - ALFRED LERNER HALL', 2.65, 311, 837], ["DOMAND'S ITALIAN GOURMET DELI & CATERERS", 4.15, 242, 480], ["GOTTE'S CAFE", 2.2, 401, 429], ['NOOR MARTS INC', 3.35, 342, 711], ['AEGEA GYROS AND PIZZA', 2.32, 132, 158], ['MEAL MART', 4.39, 782, 964], ['THE ORGANIC GRILL', 2.36, 583, 754], ['PHILLIP MARIE', 2.51, 264, 657], ['UPLAND', 3.28, 202, 276], ['BROOKLYN SWEET SPOT', 2.72, 20, 179], ['CAFE CAPRI', 3.72, 373, 402], ['FRANCIS LEWIS PASTRY SHOP', 1.08, 155, 304], ['BEDFORD KITCHEN & WINE BAR', 2.92,

132, 782], ['EAST SIDE BAGEL CAFE', 2.74, 32, 166], ['RAMEN LAB', 4.7, 100, 532], ['ANI PIZZA', 2.87, 505, 870], ['US FRIED CHICKEN & P1ZZA', 4.26, 341, 479], ['BENVENUTO CAFE', 1.61, 24, 376], ['PURE FOOD & WINE', 1.89, 157, 314], ["TEDDY'S BAR AND GRILL", 4.01, 428, 705], ['HOT SICHUAN', 2.53, 386, 685], ['NASHA RASHA', 4.02, 398, 405], ['CERTE CATERING', 1.55, 98, 654], ['PANDA EXPRESS #2634', 2.74, 678, 825], ['ISLAND CZ CAFE', 4.3, 134, 142], ['MINI MUNCHIES PIZZERIA', 3.69, 807, 828], ['OLLA WINE BAR', 4.62, 253, 632], ['VILLAGE CROWN', 1.12, 74, 833], ['BEST NEW CHINA', 3.47, 75, 948], ['AZOGUENITA BAKERY & RESTAURANT', 1.71, 4, 872], ["GABBY'S COFFEE SHOP", 1.97, 607, 870], ['LIU WRAC WELLNESS CENTER', 2.92, 189, 234], ["BAGEL BOB'S", 4.72, 407, 446], ["MURPHY'S PUB & RESTAURANT", 3.67, 21, 191], ["ANITA'S ROTI SHOP LLC.", 3.19, 40, 370], ['10TH AVENUE COOKSHOP', 2.6, 169, 694], ['WAVERLY RESTAURANT', 3.3, 31, 567], ['NUEVO HORIZONTE (HERBALIFE)', 3.24, 125, 212], ['GRACE II CHINESE RESTAURANT', 4.12, 89, 419], ['RE SETTE', 2.85, 577, 672], ['CORNER GRIND', 3.55, 240, 337], ['HI LIFE BAR & GRILL', 3.78, 245, 992], ['HOJA SANTA', 1.23, 79, 379], ['VICTORY GARDEN', 1.24, 398, 909], ['KEENANS', 1.34, 14, 419], ['HEART OF INDIA', 1.4, 346, 405], ["MR. Q'S GRILL", 2.11, 652, 965], ['KMR', 3.9, 576, 918], ['DON BURRITO RESTAURANT', 1.27, 73, 551], ['BOULEY', 2.49, 140, 263], ['BELLA PIZZA', 4.35, 52, 149], ['HOMESTYLE FOOD SERVICES (ST. BARNABAS HIGH SCHOOL)', 3.73, 139, 166], ['BLACK DOOR', 2.32, 180, 694], ['V-BAR AND CAFE', 3.89, 124, 361], ["CAVALLO'S PIZZERIA", 1.0, 159, 276], ['UGLYDUCKING', 3.37, 328, 817], ['AMLA CUISINE OF INDIA', 1.35, 385, 401], ['B & H RESTAURANT', 3.35, 359, 557], ['KO EP, LLC', 1.86, 538, 713], ['C.M. COFFEE SHOP', 1.23, 448, 754], ["SUN'S KITCHEN", 4.07, 356, 961], ['WHISKEY', 4.57, 7, 838], ['414 HOTEL', 2.91, 260, 487], ['LIBATION', 2.63, 390, 738], ['ELYNE RESTAURANT', 1.22, 543, 613], ['LA CABANA JARABACOA RESTAURANT', 3.91, 66, 183], ['SUBWAY OF BROADWAY', 1.61, 343, 708], ['HAWA SMOOTHIE', 1.75, 326, 595], ["GRANDMA'S HOUSE", 1.75, 829, 917], ['WOODROWS', 4.4, 382, 412], ['CASTILLO DE JAGUA 3', 3.49, 234, 699], ['MANHATTAN WEST', 2.74, 297, 669], ['PINCHE TAQUIERA', 3.44, 174, 770], ['FRONT TOWARD ENEMY', 2.22, 90, 197], ['CREDIT SUISSE B-2= CAFE', 3.35, 226, 349], ['UNION SQUARE SPORTS & ENTERTAINMENT AT THEATRE FOR A NEW AUDIENCE', 3.63, 363, 859], ['GREEN SUSHI BISTRO', 1.94, 124, 379], ['MANGAL KEBAB TURKISH RESTAURANT', 1.9, 516, 854], ['COCO TEA', 3.93, 168, 837], ['HUNAN MANOR RESTAURANT', 4.18, 118, 173], ["ARTHUR'S TAVERN", 3.93, 681, 985], ["DELILAH'S STEAKS", 4.28, 391, 398], ['CITY COLLEGE CAFETERIA', 4.94, 559, 857], ['KENYON & KENYON CAFETERIA', 1.31, 501, 596], ['MK KARAOKE', 2.22, 77, 293], ["QUIZNO'S SUBS", 4.27, 291, 375], ['SFOGLIA RESTAURANT', 4.37, 273, 276], ['26 SEATS', 4.63, 531, 779], ["PACINI'S PIZZERIA", 3.25, 264, 626], ['BAGELS AND MORE', 1.95, 11, 123], ['DONUT CONNECTION', 2.26, 543, 819], ['YUMMY SUSHI & FALAFEL INC', 2.04, 139, 694], ['KIRKLAND & ELLIS LLP', 2.59, 444, 763], ["PUFFY'S TAVERN", 4.53, 409, 558], ['NEW SAIGON', 1.53, 597, 880], ['GOLD MINE CAFE', 4.73, 326, 727], ["MANGANARO'S HERO BOY", 3.55, 222, 484], ['KASHKAVAL GARDEN', 2.03, 457, 794]]

Software Testing

The test output for Sprint 1 is on GitHub. The output shows all the bars that were inputted along with their scores out of 100. The bars are not ranked in any way so the order in which they came in is the order in which they come out.

Sprint Backlog
***organized neatly in SCRUM List below which elaborates on both sprints**

- Write code? Shawn & Anuj
    - Make a mathematical model of coolness and safety
        - Algorithm Ideas:
            - Bar score = bar rating/5 * safety rating
            - Population of bar and rating combined make coolness rating
            - Safety rating between 1 and 100 (higher rating = more safe)
            - Safety rating and coolness rating combined make total score
            - Total score between 1 and 100 (higher score = more recommended bar)
    - Coolness function = rating*popularity
    - Safety function = 1-popularity/max occupancy
    - Rank based on coolness & safety
- Nail down requirements? - Ankita
    - Go back to slides, use techniques to write a requirements draft
- Come up with test cases? Ernest Chiu
    - Bars, not restaurants
- Come up with a performance rubric (Mustafa)
    - Judge how well our ranking algorithm performs on the data sets

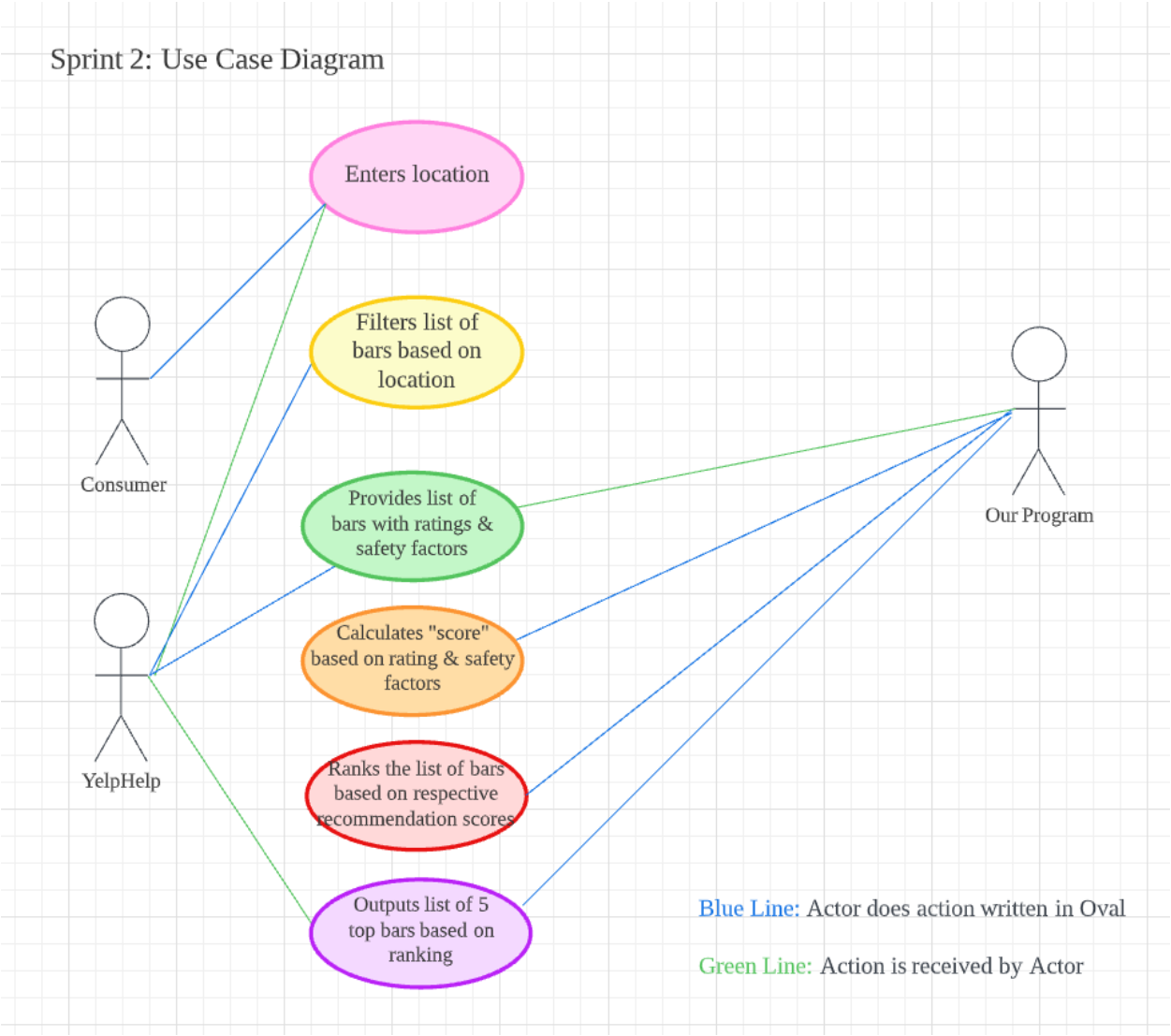
Sprint 2

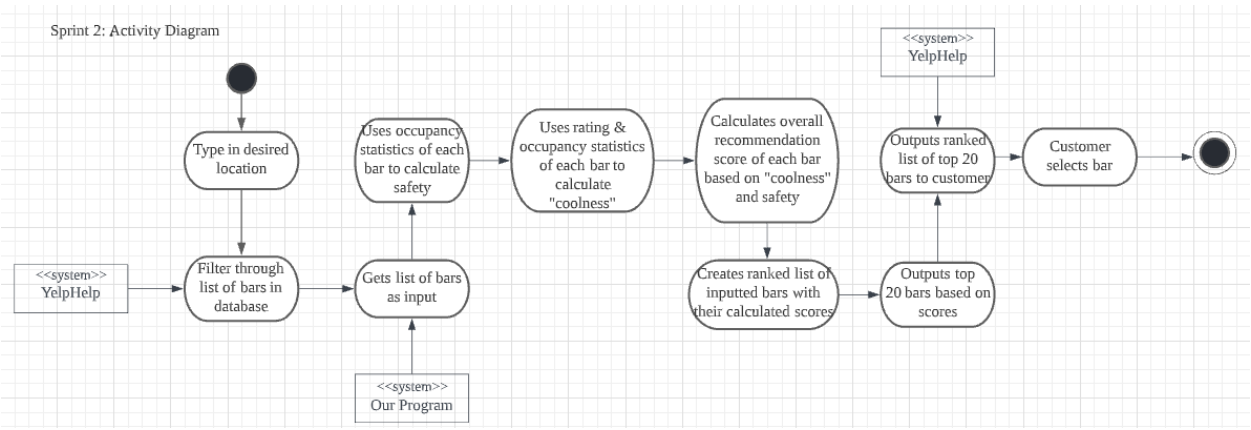
Requirements Engineering

- **Functional Requirements**
    - The factors that decide the recommendation should be bar rating & number of people at the bar.
        - Rating - official rating of the bar as stored on YelpHelp
        - Safety - based upon formula relating current occupancy & maximum occupancy of each bar
        - Popularity - based upon formula relating current occupancy & maximum occupancy of each bar
            - Should be inversely proportional to Safety
    - In the previous sprint, the formulas used are not perfectly accurate in that they assign lower scores for bars that are moderately safe (safety is 40% or better) when the overall ratings are very high.
        - Therefore, the same formula cannot be used for all the bars without discrimination.
        - Different formulas based on popularity should be used for different scenarios of safety (ie. when safety < 25% or safety > 80%).

    - Instead of just giving out scores for all the bars in the list, our program should play a more active role in the actual recommendation process by ranking the bars in order and providing the top 20 bars that it recommends.
- **Non-Functional Requirements**
    - It is confirmed that our program will not need any other information from YelpHelp other than bar's official rating (whether it is on YelpHelp or personal rating on its website) and occupancy statistics.
    - Our program will not operate differently for bars with specific COVID policies. In other words, safety will not be dependent upon the current pandemic situation so that the program will be relevant for longer.
- **Usability Requirements**
    - No interaction between the individual YelpHelp user and our program
    - YelpHelp is to be able to input a list of bars into our program continuously with any updated ratings and occupancy statistics and our program should update each time.
- **Requirements Elicitation**
    - Same as before
    - It is expected that YelpHelp doesn't expect our program to base our evaluation based on bars' additional information such as design, menu, atmosphere, etc.

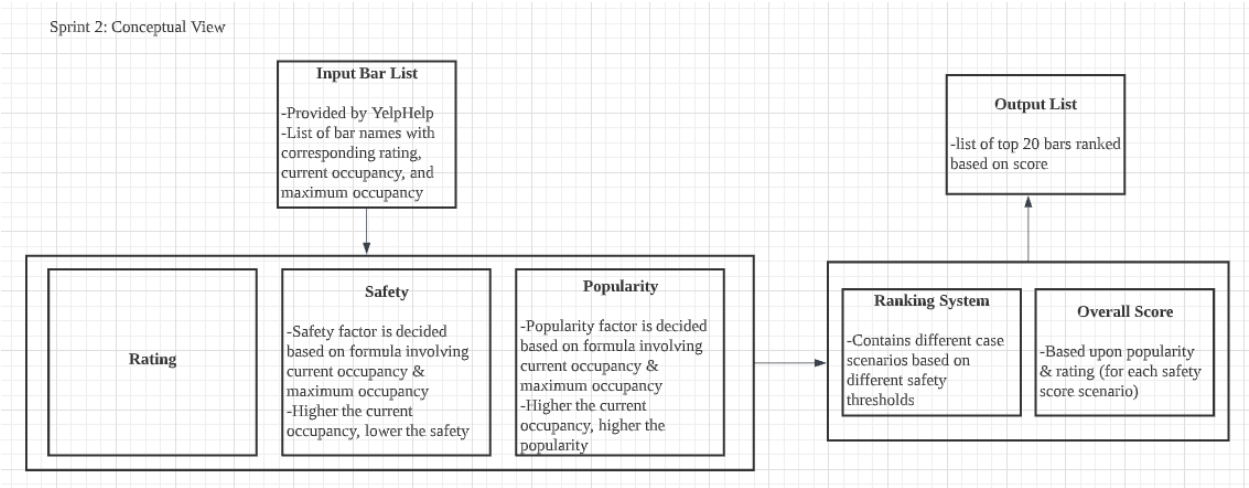
System Modeling



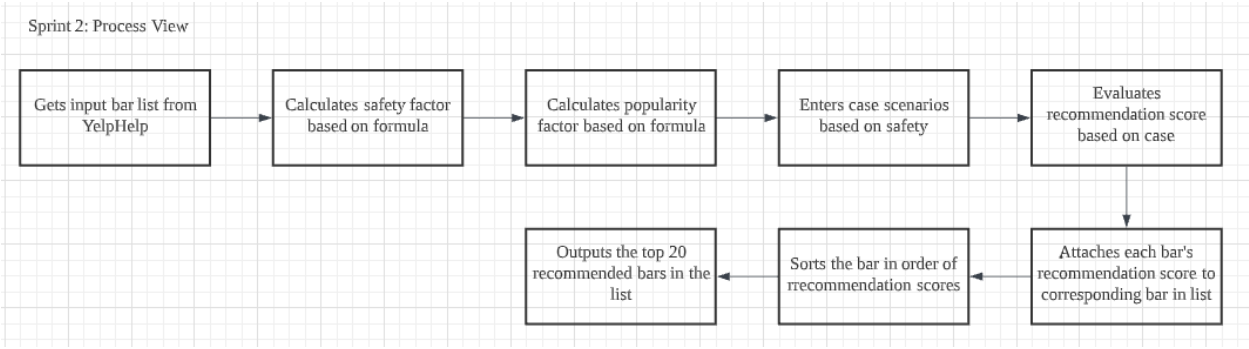


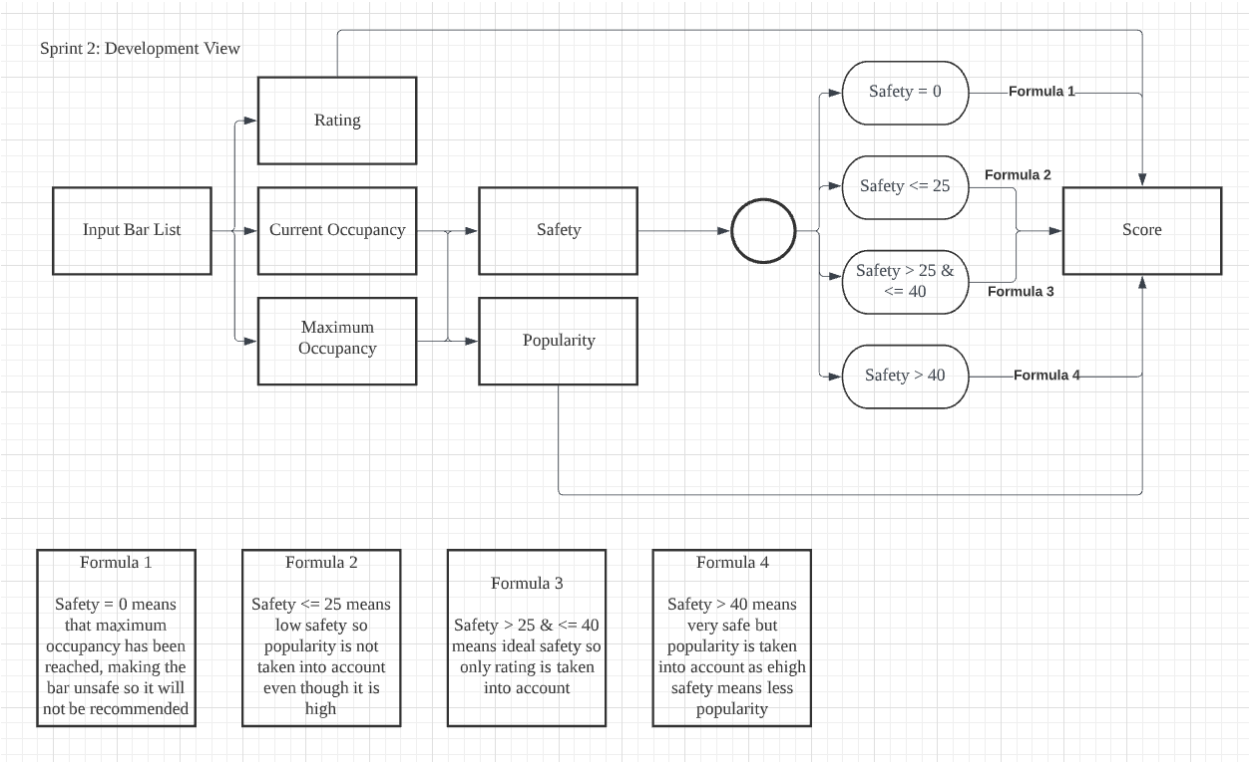
Architecture

Sprint 2: Conceptual View



Sprint 2: Process View





Design & Implementation

The Sprint 2 code on GitHub reflects the changes made to allow for the different safety thresholds. The best way to accomplish this (as referred to briefly in the Development View of the Architecture) was decided to be if-else statements. Here, popularity was made different from coolness from Sprint 1. Popularity reflects the same meaning as coolness but it isn't based upon the official rating at all. However, both popularity and rating get used for the formulas in the different scenario cases. The algorithm will contain different case scenarios as follows:

1. The first would be if a bar is completely unsafe, meaning that the bar has reached full capacity, making the safety factor 0. These bars will automatically get a score of 0, as we would not recommend going.
2. The second would be a bar that has 0 people currently there. Although this bar is not very cool, depending on the rating of the bar, the maximum score a bar like this could get is 20/100.
3. The third would be a bar that has a low safety rating, specifically below 25, meaning that the bar would have more than 75% capacity filled. The score of these bars would be based on the safety as well as the rating.
4. The fourth is the ideal bar. We decided to make the ideal bar, a bar that is not only safe, but is also popular. Bars that are at least 60% capacity, and below 75% capacity (safety is greater than 25 and less than or equal to 40) are considered ideal, as there is plenty of room, and there is also a

good amount of people there to mingle. Scores would solely be based off of the rating for these bars.

5. The last would be a bar that is not as cool. The score of these bars would be based on the popularity and the rating of the bar. Any bar with a safety factor greater than 40, meaning there is less than 60% filled, will be scored by this.

As our code reflects different case scenarios based on safety thresholds, the test cases were also generated with that in mind. Down below, the different types of cases used in creating test cases are elaborated upon.

Updated Test Cases (Sprint 2)
This has an updated data format.
The code for the generation of the test cases is on GitHub under the file testcases.py:
This update now uses targeted test cases

- Cases 1-50 are red flags– too many (>90%) occupants.
- Cases 51-100 are high rating, low occupancy.
- Cases 101-150 are low rating, high safe occupancy.
- Cases 151-200 are high rating, high safe occupancy.
- Cases 201-290 are uniform random data.
- Cases 291-300 are illegal. (more people than total)

“Bar name|number|number|number|”
“Bar name2|number|number|number|”
CHINA HOUSE KITCHEN | 4.18 | 830 | 878
J AND L DELI | 1.16 | 511 | 555
BALZEM | 1.1 | 836 | 850
ALLORO | 3.69 | 743 | 743
[Test Cases omitted for brevity, final version on github]
MOMOFUKU MILK BAR STORE | 4.26 | 390 | 252
SHAMAS DELI | 4.02 | 568 | 346
GEORGIA&ALIOU'S TINY TREATS CAFE | 4.0 | 1469 | 967
HIROKO'S PLACE | 4.57 | 1209 | 772
DELICIA DONA MARIA RESTAURANT | 4.04 | 526 | 402
STARRY BAKERY & CAFE | 4.01 | 1064 | 716

Software Testing
The test output for Sprint 2 is on GitHub. Unlike Sprint 1’s output, it shows only the top 20 bars which are ranked along with their scores out of 100.

Evaluation of Requirements & Design

- The design of the program is not too complex and easy to understand. Safety is given high precedence in this algorithm as one formula isn’t applied to all the bars.

- The revised algorithm using if-else statements is a great improvement to the previous Sprint 1 code which treated all bars equally in that it took an average of safety and coolness which doesn't reflect the variety of test cases that were provided. Bars that are realistically safe were not given good scores because their safety score was not as high as other bars. Bars that were safe were given better scores in Sprint 1 but this is not an accurate model as although they might have been safe, they didn't reflect high popularity.
- The program fits the requirements in that it factors in both safety and popularity. The bars that are ranked near the top do have a substantial amount of people in the bar but not too much to make the safety factor too low. However, they have enough people to justify its popularity and high ranking.
- The code does sort out the best bars in the list. For example, the top recommended bar "Plaza Deli" has a rating of 4.98 (out of 5) and has a safety measurement of about 32 (out of 100) so it's relatively safe. However, along with the high rating, it is quite popular as it fits into the ideal safety threshold. Therefore, its score is based solely on its rating.
    - This makes more sense than the output for Sprint 1's code where "Plaza Deli" is allotted the low score of 50 since the formula took an average of both safety and coolness. According to that formula, regardless of relative safety, it doesn't take into consideration different scenarios so technically, "Plaza Deli' has a lower safety giving it a low score.
    - Another example is "Grand Noodle House" which has a safety of about 38 but was given the low score of 40 in the Sprint 1 output. This score doesn't reflect its high popularity. In Sprint 2, it is given a better score of 68 which is based upon its rating.
- The bars in the top 20 list all fulfill the ideal safety thresholds which is good for the user as it indicates that the recommended bars are safe and popular and does not spurn either of these factors.
- The diagrams reflect the changes made from Sprint 1 to Sprint 2 and give an overview of how the code will work.
- Moreover, a vast improvement is made in Sprint 2 compared to Sprint 1 in that the bars are not listed in an arbitrary order. As they are ordered and ranked, it makes it easier for YelpHelp to show the user which bars are most recommended instead of having the user scroll and find it.
- The test cases used were designed thoroughly well as the values were randomized and reflected all the different cases that we wanted to test. It was not as if there was one bar which was completely perfect and deserved a 100 out of 100 which would not be fully realistic.
- It is possible that the program could be improved further by factoring in more parameters such as the bar's size which would play a role in safety (as even if maximum capacity is close to being reached, a larger size would still make it safe).
- Moreover, the program might have been designed to include the YelpHelp user's personal desires but it didn't seem practical to analyze a variety of bar characteristics as the program is being utilized by YelpHelp for the sole purpose of giving a recommendation. It isn't directly being used by the bars or the customers.

SCRUM List for Sprints 1 & 2

<table>
<tr><th></th><th>Mustafa</th><th>Shawn</th><th>Anuj</th><th>Ankita</th><th>Ernest</th></tr>
<tr><td>Sprint 1</td><td colspan="5">Planning Stage
Requirements of the program
The program’s main function is to take data from YelpHelp and output a recommendation value for a bar
The program had to take into account both the safety and the quality of the bar
A formula is needed to balance both factors
Delegation of Tasks
Requirements Engineering, System Modeling & Architecture - Ankita
Coding - Shawn, Anuj & Mustafa
Scrum List & Performance Evaluation Metrics - Mustafa
Test Cases - Ernest</td></tr>
<tr><td>Sprint 2</td><td colspan="2">In charge of developing the algorithm
Realized after the 1st sprint of code that the code did not take all scenarios into account in that it didn’t perfectly allot scores based on the safety and popularity values
Instead of just one formula for calculating score based on rating, safety, and popularity, came up with more formulas which are to be applied for bars which fulfill certain cases</td><td colspan="2">Wrote up all the requirements for each of the sprints
Finished system modeling & architecture for the 1st sprint
Will revise the diagrams to match the greater complexity of the 2nd sprint’s code</td><td>Found test cases to use on our algorithm</td></tr>
</table>