University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: 2

Date: 2024/07/21

Group Number: 16

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Anica Mok	60904232	w2b6c	anicamok@gmail.com
Eojin Lee	25508730	y3k9b	elee15@student.ubc.ca
Yifei Xia	55504575	y6u7g	yifeixia04@gmail.com

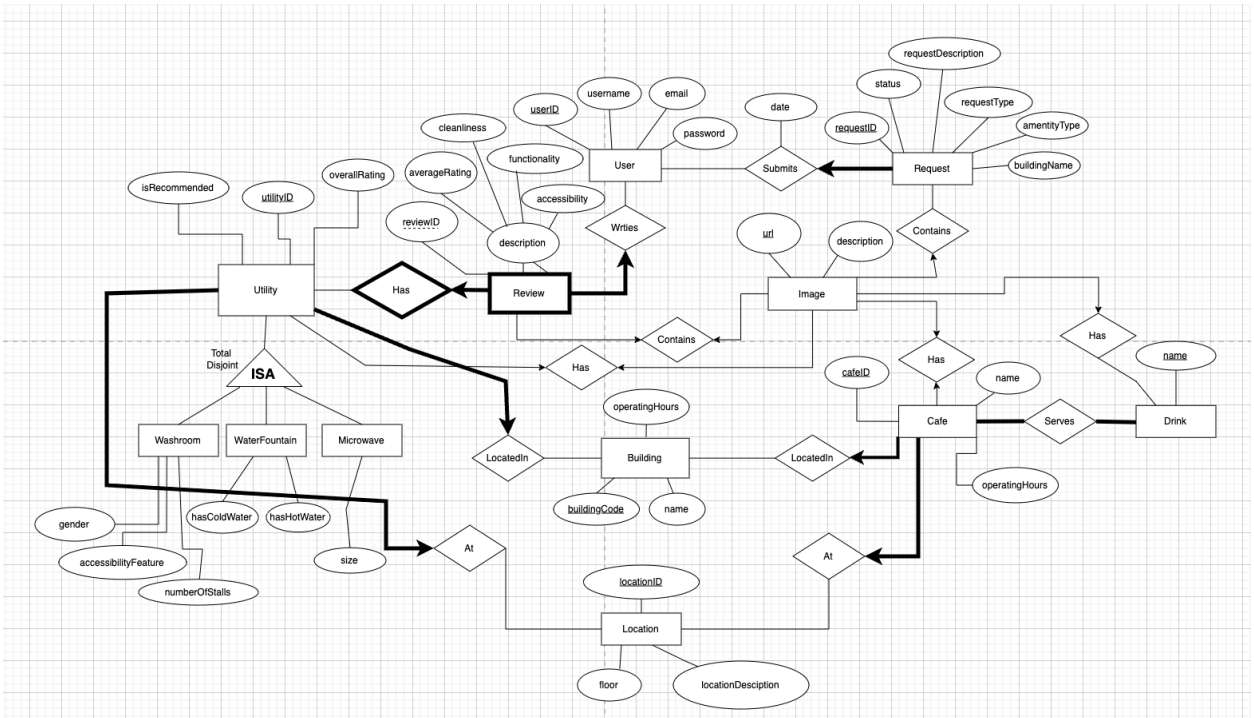
By typing our names and student numbers in the above table, we certify that the work in the attached assignment was performed solely by those whose names and student IDs are included above. (In the case of Project Milestone 0, the main purpose of this page is for you to let us know your e-mail address, and then let us assign you to a TA for your project supervisor.)

In addition, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by the Department of Computer Science and the University of British Columbia

1. Brief Summarization of Project

Our project name is “The UBC survival kit” which maps locations of various amenities such as microwaves, water fountains, washrooms, and cafes (that serve caffeinated drinks) on campus. The UBC Survival Kit aims to address the common issue faced by students in locating essential amenities within campus buildings**.**

2. ER Diagram



Changes we Made:

- **We added ISA constraints to the ISA subclasses** to make it clear that it was total disjoint – We have received feedback from the mentor to add these.

- **We have changed the naming conventions for attributes, relationships, and entities** where attributes follow lowerCamelCase, and the relationship and entities follow UpperCamelCase. This is because we wanted consistent naming conventions throughout our diagram to avoid any confusion in the future.

- **We added attributes cleanliness, functionality, accessibility and averageRating to the Review entity instead of just one rating.**
  We wanted to include a more detailed and specific rating scheme by breaking a rating into multiple sub-topics. This is to guide users to provide more meaningful feedback, helping them rate an amenity more easily and systematically. Also, certain customers might prioritize a specific quality over the other – they might be curious about a specific rating (like cleanliness) of the amenity rather than the average rating. Hence, adding these attributes will improve the reviewing process for both the contributors and the readers of the review.

- **We added operatingHour attribute to Utility** for easy querying, so that user does not have to query for the operating hour of the building separately

- **We added isRecommended to the Utility entity** . This was included to add a feature where if a certain utility has an overall rating greater than 4, then such utility will be recommended to others. This is to help users find a utility of good quality easily, without having to navigate through all the utility reviews one-by-one. This essentially makes the application more user-friendly.

- **We changed the Image entity to establish a one-to-one relationship with utilities, cafe, and reviews.** This ensures that cafes and entities are represented by one image. Additionally, we can restrict users to post one image per review. This allows each image added to our database to be much more representative and meaningful, which optimizes our data storage efficiency and simplifies the navigation through pages.

3. Translation of the ER diagram to the relational model

Utility (utilityID: INTEGER PK,
overallRating: FLOAT,
isRecommended: BOOLEAN,
buildingCode: CHAR[10] FK references Building NOT NULL,
imageURL: CHAR [50] FK references Image,
locationID: INTEGER FK references location NOT NULL,
operatingHours: CHAR[150])

Washroom (**utilityID: INTEGER** PK FK references Utility,
gender: CHAR[20], NOT NULL
numberOfStalls: INTEGER,
accessibilityFeature: CHAR[250])

waterFountain(**utilityID: INTEGER** PK FK references Utility,

hasColdWater: BOOLEAN,
hasHotWater: BOOLEAN)

microwave(**utilityID: INTEGER** PK FK references Utility,
size: CHAR [50]])

Review(reviewID: INTEGER PK,
utilityID: INTEGER FK PK references Utility,
userID: INTEGER FK references User,
cleanliness: INTEGER,
functionality: INTEGER,
accessibility: INTEGER,
overall: FLOAT
description: CHAR[250])

Location(LocationID: INTEGER PK,
floor: INTEGER NOT NULL,
locationDescription: CHAR[250] NOT NULL)

User(userID: INTEGER PK,
username: CHAR[20], NOT NULL
email: CHAR[50] CK, NOT NULL UNIQUE
password: CHAR[20] NOT NULL)

Request(requestID: INTEGER PK,
date: Date
status: CHAR[20]
requestDescription: CHAR[250],
requestType: CHAR[20] NOT NULL,
amenityType: CHAR[20] NOT NULL,
buildingName: CHAR[20] NOT NULL,
userID: INTEGER FK references User NOT NULL,
imageURL: CHAR[50] FK references Image)

Building(buildingCode: Char[10] PK,
operatingHours: Char[20],
name: CHAR[20] CK, UNIQUE NOT NULL)

Image(url: CHAR[20] PK,
description: CHAR[150])

Cafe (cafeID: INTEGER PK,
name: CHAR[50],

operatingHours: CHAR[50])

Drink (name: CHAR[20] PK)

Serves(**cafeID: INTEGER** PK, FK references Cafe,
drinkName: CHAR[20] PK, FK references Drink)

4. Functional Dependencies

Utility

- utilityID -> overallRating, buildingCode, locationID, operatingHour, overallRating, isRecommended
- overallRating -> isRecommended
- Building -> operatingHour

ISA subclasses of utility

- utilityID -> gender, accessibilityFeature, numberOfStalls
- utilityID -> hasColdWater, hasHotWater
- utilityID -> size

Review

- reviewID, utilityID -> description, userID, cleanliness, functionality, accessibility, averageRating
- cleanliness, functionality, accessibility -> averageRating

Location

- locationID -> floor, locationDescription

Building

- buildingCode -> name, operatingHours
- name -> buildingCode, operatingHours

User

- userID -> username, email, password
- email -> username, password, userID

Image

- url -> description

Cafe
- cafeID -> name, operatingHours, buildingCode, locationID

Request
- RequestID -> status, requestDescription, requestType, amenityType, buildingName, userID, imageURL

5. Normalization

The following tables are already in BCNF form as the attributes present are only defined by their primary or candidate key:

Washroom (**utilityID: INTEGER** PK FK references Utility,
gender: CHAR[20], NOT NULL
numberOfStalls: INTEGER,
accessibilityFeature: CHAR[250])

WaterFountain(**utilityID: INTEGER** PK FK references Utility,
hasColdWater: BOOLEAN,
hasHotWater: BOOLEAN)

Microwave(**utilityID: INTEGER** PK FK references Utility,
size: CHAR [50]])

Location(LocationID: INTEGER PK,
floor: INTEGER NOT NULL,
locationDescription: CHAR[250] NOT NULL)

User(userID: INTEGER PK,
username: CHAR[20], NOT NULL
email: CHAR[50] CK, NOT NULL UNIQUE
password: CHAR[20] NOT NULL)

Request(requestID: INTEGER PK,
date: Date
status: CHAR[20]
requestDescription: CHAR[250],
requestType: CHAR[20] NOT NULL,
amenityType: CHAR[20] NOT NULL,

buildingName: CHAR[20] NOT NULL,
userID: INTEGER FK references User NOT NULL,
imageURL: CHAR[50] FK references Image)

Building(buildingCode: Char[10] PK,
operatingHours: Char[20],
name: CHAR[20] CK, UNIQUE NOT NULL)

Image(url: CHAR[20] PK,
description: CHAR[150])

Cafe (cafeID: INTEGER PK,
name: CHAR[50],
operatingHours: CHAR[50])

Drink (name: CHAR[20] PK)

Serves(**cafeID: INTEGER** PK, FK references Cafe,
drinkName: CHAR[20] PK, FK references Drink)

However, the tables Utility and Review will need to go through normalization:

1) Utility

Utility (utilityID: INTEGER PK,
overallRating: FLOAT,
isRecommended: BOOLEAN,
buildingCode: CHAR[10] FK references Building NOT NULL,
imageURL: CHAR [50] FK references Image,
locationID: INTEGER FK references location NOT NULL,
operatingHours: CHAR[150])

For simplicity, we will refer to the relation like so: U(UID, O, R, B, I, L, H)

FDs:
UID -> O, R, B, I, L, H
O -> R
B -> H

O -> R violates BCNF, so we get:
U1(O, R) , U2(UID, O, B, I, L, H).

B->H still violates BCNF for U2, so we further decompose into:
U3(B, H), U3 (UID, O, B, I , L).

Thus we get tables :

U1: Rating (overallRating: float PK,
IsRecommended: boolean)

U2: Hours (**buildingCode: char[10]** PK FK references Building,
operatingHour: char [150])

U3: Utility(utilityID: Integer PK,
overallRating: float,
buildingCode: char[10] FK references building, NOT NULL,
imageURL: char[50] FK references Image,
locationID: integer FK references location, NOT NULL)

2) Review

Review(reviewID: INTEGER PK,
utilityID: INTEGER FK PK references Utility,
userID: INTEGER FK references User,
cleanliness: INTEGER,
functionality: INTEGER,
accessibility: INTEGER,
overall: FLOAT
description: CHAR[250])

For simplicity, we will refer to the relation like so: Review(RID, U, UID, C, F, A, AR, D)

Relevant FDs:
RID, U -> UID, C, F, A, SR, D
C, F, A -> AR

C, F, A -> AR violates BCNF so we decompose into:
R1(C, F, A, AR) R2(RID, U, UID, C, F, A, D).

Thus we have the tables:

R1: AverageRating(cleanliness: integer,
functionality: Integer,
accessibility: Integer,

averageRating: Float,
Primary Key (cleanliness, functionality, accessibility))

R2: Review(reviewID: Integer PK,
utilityID: integer FK PK references Utility,
userID: integer FK references User,
cleanliness: integer, NOT NULL
functionality: Integer, NOT NULL
accessibility: Integer NOT NULL
description: char[250]).

6. SQL DDL to create all the tables from item #5.

```
CREATE TABLE Location (
    locationID INTEGER PRIMARY KEY,
    floor INTEGER NOT NULL,
    locationDescription VARCHAR(250) NOT NULL
);

CREATE TABLE User(
    userID INTEGER PRIMARY KEY,
    username VARCHAR(20) NOT NULL,
    email VARCHAR(50) UNIQUE NOT NULL,
    password VARCHAR(20) NOT NULL
);

CREATE TABLE Request(
    requestID INTEGER PRIMARY KEY,
    date DATE,
    status VARCHAR(20),
    requestDescription VARCHAR(250),
    requestType VARCHAR(20) NOT NULL,
    amenityType VARCHAR(20) NOT NULL,
    buildingName VARCHAR(20) NOT NULL,
    userID INTEGER NOT NULL DEFAULT 0,
    imageURL VARCHAR(50),
    FOREIGN KEY (imageURL) REFERENCES Image(url),
    FOREIGN KEY (userID) REFERENCES User(userID) ON DELETE SET DEFAULT
);

CREATE TABLE Building(
    buildingCode VARCHAR(10) PRIMARY KEY,
    operatingHours VARCHAR(20),
    name VARCHAR(20) UNIQUE
);

CREATE TABLE Image(
```

```
    url VARCHAR(20) PRIMARY KEY,
    description VARCHAR(150)
);

CREATE TABLE Utility (
    utilityID INTEGER PRIMARY KEY,
    overallRating FLOAT,
    buildingCode VARCHAR(10) NOT NULL,
    imageURL VARCHAR(50),
    locationID INTEGER NOT NULL,
    FOREIGN KEY (imageURL) REFERENCES Image(url),
    FOREIGN KEY (locationID) REFERENCES Location,
    FOREIGN KEY (buildingCode) REFERENCES Building ON DELETE CASCADE
);

CREATE TABLE Cafe(
    cafeID INTEGER PRIMARY KEY,
    name VARCHAR(50),
    operatingHours VARCHAR(50)
);

CREATE TABLE Drink(name VARCHAR(20) PRIMARY KEY);

CREATE TABLE Serves(
    cafeID INTEGER,
    drinkName VARCHAR(20),
    PRIMARY KEY (cafeID, drinkName),
    FOREIGN KEY (cafeID) REFERENCES Cafe ON DELETE CASCADE,
    FOREIGN KEY (drinkName) REFERENCES Drink(name) ON DELETE CASCADE
);

CREATE TABLE Rating(
    overallRating FLOAT PRIMARY KEY,
    isRecommended BOOLEAN
);

CREATE TABLE Hours(
    buildingCode VARCHAR(10) PRIMARY KEY,
    operatingHour VARCHAR(150),
    FOREIGN KEY (buildingCode) REFERENCES Building ON DELETE CASCADE
);

CREATE TABLE Washroom(
    utilityID INTEGER PRIMARY KEY,
    gender VARCHAR(30),
    numberOfStalls INTEGER,
    accessibilityFeature VARCHAR(250),
    FOREIGN KEY (utilityID) REFERENCES Utility ON DELETE CASCADE
);
```

```
CREATE TABLE Microwave(
    utilityID INTEGER PRIMARY KEY,
    size VARCHAR(50),
    FOREIGN KEY (utilityID) REFERENCES Utility ON DELETE CASCADE
);

Create TABLE WaterFountain(
    utilityID INTEGER PRIMARY KEY,
    hasColdWater BOOLEAN,
    hasHotWater BOOLEAN,
    FOREIGN KEY (utilityID) REFERENCES Utility ON DELETE CASCADE
);

CREATE TABLE AverageRating(
    cleanliness INTEGER,
    functionality INTEGER,
    accessibility INTEGER,
    averageRating FLOAT,
   PRIMARY KEY (cleanliness, functionality, accessibility)
);

CREATE TABLE Review(
    reviewID INTEGER,
    utilityID INTEGER,
    userID INTEGER NOT NULL DEFAULT 0,
    cleanliness INTEGER NOT NULL,
    functionality INTEGER NOT NULL,
    accessibility INTEGER NOT NULL,
    description VARCHAR(250),
    PRIMARY KEY (reviewID, utilityID),
    FOREIGN KEY (utilityID) REFERENCES Utility,
    FOREIGN KEY (userID) REFERENCES User ON DELETE SET DEFAULT
);
```

7. Tuples

```
INSERT
INTO Image(url, description)
VALUES
("/images/iced-latte", "iced latte"),
("/images/london-fog", "london fog"),
("/images/cappuccino", "cappuccino"),
("/images/sauder-tim-hortons", "Tim Hortons (Sauder location)"),
("/images/ICICS-top-floor-male-washroom", "Male Washroom (ICICS top
floor)")
```

```
INSERT
INTO Cafe(cafeID, name, operatingHours)
VALUES
(1732, "Tim Hortons", "9:00 - 21:00"),
(3812, "Starbucks", "8:30 - 23:00"),
(8128, "Blue Chip Cafe", "9:00 - 17:00"),
(5921, "Loafe Cafe", "8:00 - 17:00"),
(7391, "JJ Bean Coffee Roasters", "8:00 - 17:00")

INSERT
INTO Drink(name)
VALUES
("Iced Latte"),
("London Fog"),
("Cappuccino"),
("Dark Roast"),
("Espresso")

INSERT
INTO Serves(cafeID, drinkName)
VALUES
("Tim Hortons", "Iced Latte"),
("Tim Hortons", "Espresso"),
("Starbucks", "London Fog"),
("Starbucks", "Dark Roast"),
("Starbucks", "Iced Latte")


INSERT
INTO Utility(utilityID, overallRating, buildingCode, imageURL, locationID)
VALUES
(0, 2.4, “BIOL”, "/images/BIOL", 0),
(1, 5.0, “ICCS”, "/images/ICICS-top-floor-male-washroom", 2),
(2, 3.2, “ICCS”, "/images/ICCS-elevator", 1),
(3, 1.9, “NEST”, "/images/NEST-garden", 3),
(4, 4.6, “NEST”, "/images/NEST-washroom", 5)

INSERT INTO Location (locationID, floor, locationDescription)
VALUES
    (0, 69, 'default location'),
    (1, 2, 'near the elevator'),
    (2, 2, 'by the men''s washroom'),
    (3, 4, 'by rooftop garden'),
    (4, 1, 'by Science One exclusive lounge');

INSERT INTO User (userID, username, email, password)
VALUES
    (0, 'defaultUser', 'd@email.com', 'password'),
    (1, 'Kyle', 'k@email.com', 'password'),
```

```
    (2, 'Sarah', 's@email.com', 'password'),
    (3, 'Maddie', 'm@email.com', 'password'),
    (4, 'Joey', 'j@email.com', 'password');

INSERT INTO Request (
    requestID, date, status, requestDescription, requestType, amenityType,
buildingName, userID, imageURL)
VALUES
    (0, '2024-07-01', ‘in-progress’,  'default', 'update', 'fountain', 'CS
building', 0, NULL),
    (1, '2023-07-01', ‘pending’,  'water fountain broke', 'update',
'fountain', 'Nest', 2, NULL),
    (2, '2024-05-01', ‘pending’,  'new cafe', 'add', 'cafe', 'fred kaiser',
3, NULL),
    (3, '2024-03-01', ‘complete,  'new neutral washroom', 'add',
'washroom', 'math annex', 4, NULL),
    (4, '2024-04-20', ‘in-progress’, 'microwave broke', 'update',
'microwave', 'biology', 1, NULL);

INSERT INTO Building (buildingCode, operatingHours, name)
VALUES
    ('BIOL', 'Mon to Fri: 7:30AM - 8:00PM, Sat/Sun/Holidays: Closed',
'Biological Sciences'),
    ('ICCS', 'Mon to Fri: 7:30AM - 5:00PM, Sat/Sun/Holidays: Closed',
'Institute for Computing'),
    ('NEST', 'Monday to Friday: 7AM - 11PM, Saturday to Sunday: 8AM -
11PM', 'AMS Student Nest'),
    ('MATX', 'Mon to Fri: 7:30AM - 5:00PM, Sat/Sun/Holidays: Closed',
'Mathematics Annex'),
    ('KAIS', 'Mon to Fri: 7:00AM - 6:00PM, Sat/Sun/Holidays: Closed', 'Fred
Kaiser');



INSERT INTO Washroom (utilityID, gender, numberOfStalls,
accessibilityFeature)
VALUES
    (10000000, 'FEMALE', 2, 'WHEELCHAIR STALL'),
    (10000001, 'FEMALE', 5, 'WHEELCHAIR STALL'),
    (10000002, 'NEUTRAL', 1, 'GRAB BAR'),
    (10000003, 'MALE', 2, 'NONE'),
    (10000004, 'MALE', 4, 'NONE');

INSERT INTO Microwave (utilityID, size)
VALUES
    (20000000, 'SMALL'),
    (20000001, 'LARGE'),
    (20000002, 'MEDIUM'),
```

```
    (20000003, 'SMALL'),
    (20000004, 'MEDIUM');

INSERT INTO WaterFountain (utilityID, hasColdWater, hasHotWater)
VALUES
    (30000000, TRUE, FALSE),
    (30000001, TRUE, FALSE),
    (30000002, TRUE, TRUE),
    (30000003, TRUE, FALSE),
    (30000004, TRUE, TRUE);

INSERT INTO AverageRating
VALUES
    (1, 1, 1, 1.0),
    (2, 2, 2, 2.0),
    (3, 3, 3, 3.0),
    (4, 4, 4, 4.0),
    (5, 5, 5, 5.0);

INSERT INTO Review
VALUES
    (1, 1, 1, 1, 1, 1, 'REALLY BAD'),
    (2, 1, 1, 2, 1, 3, 'AWFUL'),
    (3, 3, 2, 2, 3, 4, 'MEH'),
    (4, 4, 3, 5, 5, 5, 'AMAZING'),
    (5, 2, 4, 5, 3, 4, 'GREAT!');

INSERT INTO Rating (overallRating, isRecommended)
VALUES (4.5, TRUE),
(3.8, FALSE),
(4.9, TRUE),
(2.7, FALSE),
(5.0, TRUE);

INSERT INTO Hours (buildingCode, operatingHour)
VALUES ('ICCS', 'Mon-Fri: 8am-10pm, Sat: 9am-8pm, Sun: 10am-6pm'),
('BIOL', 'Mon-Fri: 7am-11pm, Sat-Sun: 8am-9pm'),
('KAIS', 'Mon-Fri: 9am-9pm, Sat: 10am-6pm, Sun: Closed'),
('NEST', 'Mon-Sat: 8am-8pm, Sun: 9am-5pm'),
('MATX', 'Mon-Fri: 6am-12am, Sat-Sun: 7am-10pm');
```