University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: 2

Date: 6/12/23

Group Number: 30

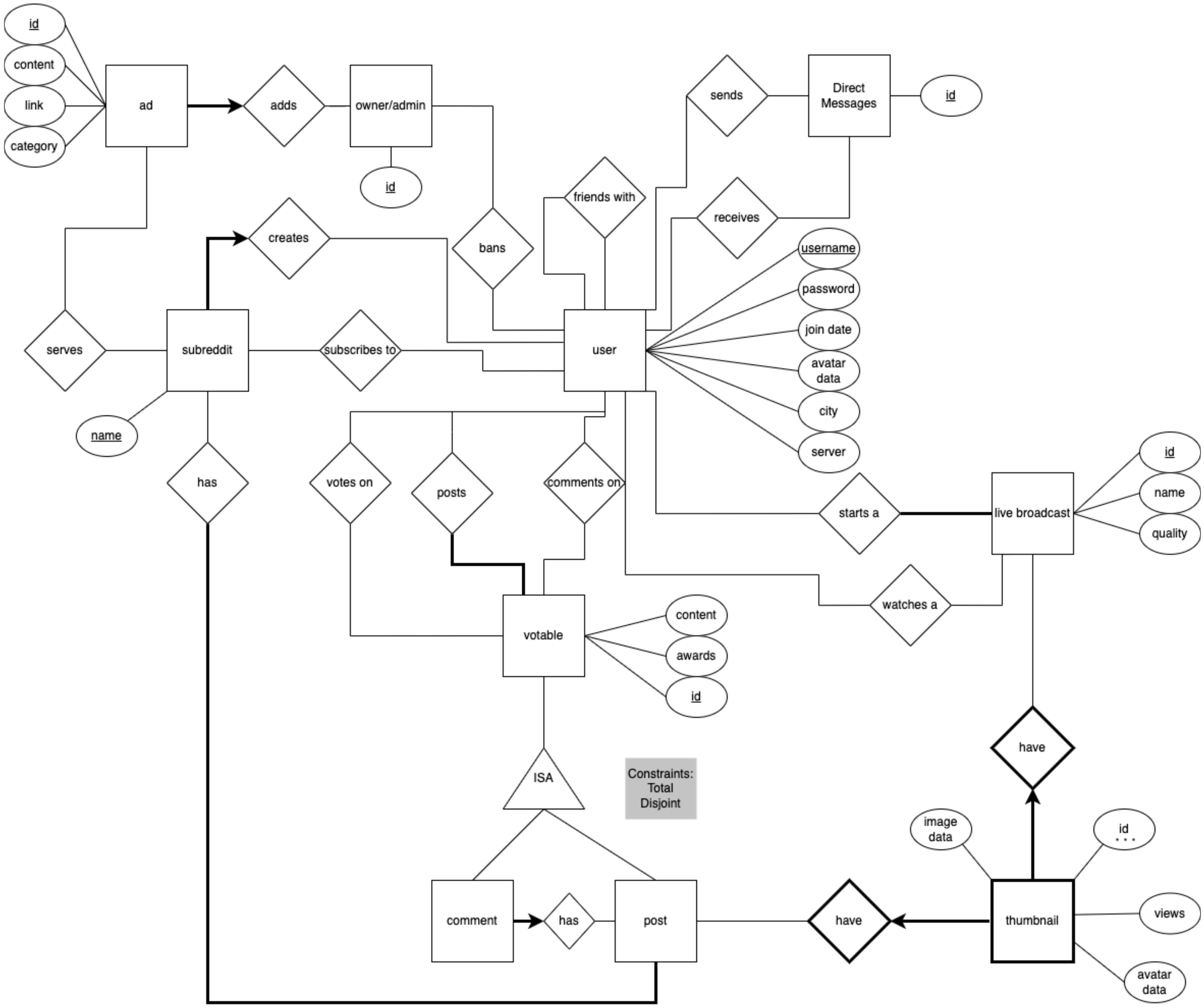
Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Logan Keener	67439018	w2m9y	lokeener@student.ubc.ca
Gavin George	38240818	n0x6y	gavinmgeorge@gmail.com
Wendy Han	51108264	c1g2b	ws.han@outlook.com

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

<u>Summary:</u>

Our project is a social media site similar to Reddit. It has users, subreddits, comments, live broadcasts, ads, messages, admins, thumbnails and posts.



Changes made:

- id added to thumbnail: TA suggestion for primary keys
- ISA constraints added: TA suggestion
- id added to live broadcast: TA suggestion
- Total participation for the “starts a” relationship: TA suggestion
- direct message id: TA suggestion for primary keys
- Total participation added through new “posts” relationship between votable and user, since every post/comment has a user behind it, but two users could post something together (similar feature on instagram)
- id added to votable
- Added participation constraint with comment since each one will have a parent post
- Post has total participation with subreddit: TA suggestion
- Subreddit has total participation with user since every subreddit has exactly one initial creator

- Total participation between ad and admin, since ads must be approved by owners before being added
- id added to ad
- added attributes city and server to user (city associated to its own regional server)
- added attributes views and popularity to broadcast (measuring the views of the broadcast, and using a rating between 1-10 for popularity level)

Schema

Message(messageId: integer, sender: char[20], receiver: char[20])

- messageId is a unique PK
- sender is foreign key for User
- receiver is foreign key for User
- all should be non-null

FriendsWith(friend1: char[20], friend2: char[20])

- friend1 and friend2 are foreign keys for different Users
- PK: (friend1, friend2)

Ban(adminId: integer, userId: integer)

- adminId is non-null
- userId is a foreign key for User, non-null
- PK: (adminId, userId)

Ad(id: integer, content: char[60], link: char[50], category: integer, adminId: integer)

- id is a unique non-null PK
- content, category, adminId and link are non-null

Serves(adId: integer, subId: char[20])

- adId is a non-null foreign key for ad
- subId is a non-null foreign key for Subreddit
- PK: (adId, subId)

Subscribes(userId: char[20], subId: char[20])

- userId is non-null foreign key for User
- subId is non-null foreign key for Subreddit
- PK: (userId, subId)

VotesOn(userId: char[20], votableId: integer)

- userId is non-null foreign key for User
- votableId is non-null foreign key for Votable
- PK: (userId, votableId)

User(username: char[20], password: char[20], joinDate: date, avatarData: char[60], city: char[20], server: char[20])

- username and password are non-null

Broadcast(liveId: integer, title: char[30], quality: integer, creatorId: char[20], views: integer, popularity: integer [1…10])

- liveId is a non-null unique PK
- title is non-null
- creatorId is a non-null foreign key for User

Thumbnail(liveId: integer, postId: integer, tnId: char[20] imageData: char[60])

- liveId is a non-null foreign key for the related Broadcast
- postId is a non-null unique weak entity foreign key
- tnId is a non-null primary key that, along with liveId OR postId (both CKs)
- PK: (liveId, tnId)

Subreddit(name: char[20], creatorId: char[20])

- name is non-null unique PK
- creatorId is a foreign key for User

Votable(votableId: integer, awards: char[30], content: char[60], creatorId: char[20])

- votableId is a non-null unique PK
- creatorId is a foreign key for User

Comment(votableId: integer, parentId: char[20], awards: char[30], content: char[60], creatorId: char[20])

- creatorId is a non-null foreign key for Comment for the user who posted
- parentId is a non-null foreign key for Votable for the comment’s parent post
- votableId is a non-null unique PK

Posts(votableId: integer, subId: integer) (to represent “has” relationship between Post and Subreddit)

- votableId is a non-null unique foreign key for Votable
- subId is a non-null foreign key for Subreddit
- PK: (votableId, subId)

Functional dependencies:

Message Table:
messageID -> sender, receiver
sender -> username

receiver -> username

FriendsWith:
friend1 -> username
friend2 -> username

Ban
userId -> username

Ad
id -> content, link, category, adminId

Serves
adId -> Ad.id
subId -> Subreddit.name

Subscribes
userId -> username
subId -> Subreddit.name

VotesOn
userId -> username
votableId -> Votable.id

User:
username -> password, joinDate, avatarData
city -> server

Broadcast:
creatorId -> username
views -> popularity

Thumbnail:
liveId -> Broadcast.id
postId -> Votable.id
tnId -> imageData

Subreddit:
creatorId -> username

Votable:
votableId -> awards, content, creatorId

Comment:
votableId -> parentId, awards, content, creatorId
parentId -> Votable.votableId

Posts:
votableId -> Votable.id
subId -> Subreddit.name

Normalization

two FDs violate 3NF (city -> server, views -> popularity)

User(username: char[20], password: char[20], joinDate: date, avatarData: char[60], city: char[20], server: char[20])

Abbreviate as R(U, P, J, A, C, S)
split based on C -> S

R1(U, P, J, A, C) R2(C, S)

Split into tables:
User(username: char[20], password: char[20], joinDate: date, avatarData: char[60], city: char[20])

- previously mentioned constraints
- city is a foreign key for Location table

Location(city: char[20], server: char[20])

- city is a non-null unique PK
- server is non-null

Broadcast(liveId: integer, title: char[30], quality: integer, creatorId: char[20], views: integer, popularity: integer [1…10])

Abbreviate as B(L, T, Q, C, V, P)
split based on V -> P

B1(L, T, Q, C, V) B2(V, P)

Split into tables:
Broadcast(liveId: integer, title: char[30], quality: integer, creatorId: char[20], views: integer)

- previously mentioned constraints
- views is a foreign key for Popular table

Popular(views: integer, popularity: integer[1…10])

- views is a unique, non-null PK
- popularity is non-null

Full Schema after Normalization:

Message(messageId: integer, sender: char[20], receiver: char[20])

- messageId is a unique PK
- sender is foreign key for User
- receiver is foreign key for User
- all should be non-null

FriendsWith(friend1: char[20], friend2: char[20])

- friend1 and friend2 are foreign keys for different Users
- PK: (friend1, friend2)

Ban(adminId: integer, userId: integer)

- adminId is non-null
- userId is a foreign key for User, non-null
- PK: (adminId, userId)

Ad(id: integer, content: char[60], link: char[50], category: integer, adminId: integer)

- id is a unique non-null PK
- content, category, adminId and link are non-null

Serves(adId: integer, subId: char[20])

- adId is a non-null foreign key for ad
- subId is a non-null foreign key for Subreddit
- PK: (adId, subId)

Subscribes(userId: char[20], subId: char[20])

- userId is non-null foreign key for User
- subId is non-null foreign key for Subreddit
- PK: (userId, subId)

VotesOn(userId: char[20], votableId: integer)

- userId is non-null foreign key for User
- votableId is non-null foreign key for Votable
- PK: (userId, votableId)

User(username: char[20], password: char[20], joinDate: date, avatarData: char[60], city: char[20])

- previously mentioned constraints
- city is a foreign key for Location table

Location(city: char[20], server: char[20])

- city is a non-null unique PK
- server is non-null

Broadcast(liveId: integer, title: char[30], quality: integer, creatorId: char[20], views: integer)

- previously mentioned constraints
- views is a foreign key for Popular table

Popular(views: integer, popularity: integer[1…10])

- views is a unique, non-null PK
- popularity is non-null

Thumbnail(liveId: integer, postId: integer, tnId: char[20] imageData: char[60])

- liveId is a non-null foreign key for the related Broadcast
- postId is a non-null unique weak entity foreign key
- tnId is a non-null primary key that, along with liveId OR postId (both CKs)
- PK: (liveId, tnId)

Subreddit(name: char[20], creatorId: char[20])

- name is non-null unique PK
- creatorId is a foreign key for User

Votable(votableId: integer, awards: char[30], content: char[60], creatorId: char[20])

- votableId is a non-null unique PK
- creatorId is a foreign key for User

Comment(votableId: integer, parentId: char[20], awards: char[30], content: char[60], creatorId: char[20])

- creatorId is a non-null foreign key for Comment for the user who posted
- parentId is a non-null foreign key for Votable for the comment’s parent post
- votableId is a non-null unique PK

Posts(votableId: integer, subId: integer) (to represent “has” relationship between Post and Subreddit)

- votableId is a non-null unique foreign key for Votable
- subId is a non-null foreign key for Subreddit
- PK: (votableId, subId)

SQL DDL To Create Tables and Insert Data:

```
-- Note: changed User table name to UserTable because User is a reserved
word
--  changed name of subreddit -> subName because name is a reserved word
-- changed name of server -> serverName because server is a reserved word
CREATE TABLE Location (
    city char(20),
    serverName char(20),
    PRIMARY KEY(city)
);

CREATE TABLE UserTable (
    username char(20) PRIMARY KEY NOT NULL,
    password char(20) NOT NULL,
    joinDate DATE,
    avatarData char(60),
    city char(20),
    FOREIGN KEY (city) REFERENCES Location(city)
);

CREATE TABLE Message (
    messageId int NOT NULL,
    sender char(20) NOT NULL,
    receiver char(20) NOT NULL,
    PRIMARY KEY(messageId),
    FOREIGN KEY (sender) REFERENCES UserTable(username),
    FOREIGN KEY (receiver) REFERENCES UserTable(username)
);

CREATE TABLE FriendsWith (
    friend1 char(20),
    friend2 char(20),
    PRIMARY KEY(friend1, friend2),
    FOREIGN KEY (friend1) REFERENCES UserTable(username),
    FOREIGN KEY (friend2) REFERENCES UserTable(username)
);

CREATE TABLE Ban (
    adminId int NOT NULL,
    userId char(20) NOT NULL,
    PRIMARY KEY(adminId, userId),
    FOREIGN KEY (userId) REFERENCES UserTable(username)
);

CREATE TABLE Ad (
    id int,
    content char(60) NOT NULL,
    link char(50) NOT NULL,
    category int NOT NULL,
    adminId int NOT NULL,
    PRIMARY KEY(id)
);

CREATE TABLE Subreddit(
    subName char(20),
    creatorId char(20),
    PRIMARY KEY (subName),
    FOREIGN KEY (creatorId) REFERENCES UserTable(username)
);

CREATE TABLE Serves (
    adId int NOT NULL,
    subId char(20) NOT NULL,
    PRIMARY KEY(adId, subId),
    FOREIGN KEY (adId) REFERENCES Ad(id),
    FOREIGN KEY (subId) REFERENCES Subreddit(subName)
);

CREATE TABLE Subscribes (
    userId char(20) NOT NULL,
    subId char(20) NOT NULL,
    PRIMARY KEY (userId, subId),
    FOREIGN KEY (userId) REFERENCES UserTable(username),
    FOREIGN KEY (subId) REFERENCES Subreddit(subName)
);

CREATE TABLE Votable(
    votableId int,
    awards char(30),
    content char(60),
    creatorId char(20),
    PRIMARY KEY (votableId),
    FOREIGN KEY (creatorId) REFERENCES UserTable(username)
);

CREATE TABLE VotesOn(
    userId char(20) NOT NULL,
    votableId int NOT NULL,
    FOREIGN KEY (userId) REFERENCES UserTable(username),
    FOREIGN KEY (votableId) REFERENCES Votable(votableId),
    PRIMARY KEY (userId, votableId)
);

CREATE TABLE Popular(
    views int,
    popularity int NOT NULL CHECK (popularity BETWEEN 1 AND 10),
    PRIMARY KEY (views)
);

CREATE TABLE Broadcast(
    liveId int,
    title char(30) NOT NULL,
    quality int,
    creatorId char(20),
    views int,
    PRIMARY KEY (liveId),
    FOREIGN KEY (creatorId) REFERENCES UserTable(username),
    FOREIGN KEY (views) REFERENCES Popular(views)
);

CREATE TABLE Thumbnail(
    liveId int,
    postId int NOT NULL,
    imageData char(60),
    PRIMARY KEY (postId),
    FOREIGN KEY (liveId) REFERENCES Broadcast(liveId)
```

```
);

CREATE TABLE Comment(
    votableId int,
    parentId int NOT NULL,
    awards char(30),
    content char(60),
    creatorId char(20),
    PRIMARY KEY (votableId),
    FOREIGN KEY (parentId) REFERENCES Votable(votableId),
    FOREIGN KEY (creatorId) REFERENCES UserTable(username)
);

CREATE TABLE Posts(
    votableId int NOT NULL UNIQUE,
    subId char(20) NOT NULL,
    FOREIGN KEY (votableId) REFERENCES Votable(votableId),
    FOREIGN KEY (subId) REFERENCES Subreddit(subName),
    PRIMARY KEY (votableId, subId)
);

-- populate Location
INSERT INTO Location VALUES ("vancouver", "server 1");
INSERT INTO Location VALUES ("toronto", "server 2");
INSERT INTO Location VALUES ("quebec", "server 3");
INSERT INTO Location VALUES ("ny", "server 4");
INSERT INTO Location VALUES ("boston", "server 5");

-- populate UserTable
INSERT INTO UserTable VALUES ('gavin', '100', DATE("2017-06-15"), "some
user data", "vancouver");
INSERT INTO UserTable VALUES ('bob', 'pass1', DATE("2018-06-15"), "some
user data2", "toronto");
INSERT INTO UserTable VALUES ('helen', 'pass2', DATE("2019-06-15"), "some
user data3", "vancouver");
INSERT INTO UserTable VALUES ('ben', 'pass3', DATE("2020-06-15"), "some
user data4", "vancouver");
INSERT INTO UserTable VALUES ('robert', 'pass4', DATE("2021-06-20"), "some
user data5", "ny");
```

```
-- populate Message
INSERT INTO Message VALUES (1, 'gavin', 'helen');
INSERT INTO Message VALUES (2, 'gavin', 'bob');
INSERT INTO Message VALUES (3, 'gavin', 'ben');
INSERT INTO Message VALUES (4, 'robert', 'helen');
INSERT INTO Message VALUES (5, 'robert', 'ben');

-- populate FriendsWith
INSERT INTO FriendsWith VALUES ('gavin', 'helen');
INSERT INTO FriendsWith VALUES ('gavin', 'bob');
INSERT INTO FriendsWith VALUES ('gavin', 'ben');
INSERT INTO FriendsWith VALUES ('robert', 'ben');
INSERT INTO FriendsWith VALUES ('ben', 'bob');

-- populate Ban
INSERT INTO Ban VALUES (1, 'helen');
INSERT INTO Ban VALUES (1, 'bob');
INSERT INTO Ban VALUES (1, 'ben');
INSERT INTO Ban VALUES (2, 'robert');
INSERT INTO Ban VALUES (2, 'gavin');

-- populate Ad
INSERT INTO Ad VALUES (1, 'buy the stuff', 'buy the stuff.com', 10, 1);
INSERT INTO Ad VALUES (2, 'buy other stuff', 'buy other stuff.com', 11,
1);
INSERT INTO Ad VALUES (3, 'buy cereal', 'buy cereal.com', 12, 1);
INSERT INTO Ad VALUES (4, 'circlespace', 'circlespace.com', 13, 2);
INSERT INTO Ad VALUES (5, 'vpn', 'vpn.com', 14, 2);

-- populate Subreddit
INSERT INTO Subreddit VALUES ('cats', 'gavin');
INSERT INTO Subreddit VALUES ('dogs', 'helen');
INSERT INTO Subreddit VALUES ('ubc', 'bob');
INSERT INTO Subreddit VALUES ('movies', 'gavin');
INSERT INTO Subreddit VALUES ('anime', 'ben');

-- populate Serves
INSERT INTO Serves VALUES (1, 'cats');
INSERT INTO Serves VALUES (2, 'dogs');
INSERT INTO Serves VALUES (3, 'ubc');
INSERT INTO Serves VALUES (4, 'movies');
INSERT INTO Serves VALUES (5, 'anime');

-- populate Subscribes
INSERT INTO Subscribes VALUES ('helen', 'cats');
INSERT INTO Subscribes VALUES ('bob', 'dogs');
INSERT INTO Subscribes VALUES ('ben', 'ubc');
INSERT INTO Subscribes VALUES ('robert', 'movies');
INSERT INTO Subscribes VALUES ('gavin', 'anime');

-- populate Votable
INSERT INTO Votable VALUES (1, 'none :(', 'wow a cool votable', 'helen');
INSERT INTO Votable VALUES (2, 'none :(', 'wow a cool votable2', 'bob');
INSERT INTO Votable VALUES (3, 'shiny', 'wow a cool votable3', 'ben');
INSERT INTO Votable VALUES (4, 'none :(', 'wow a cool votable4',
'robert');
INSERT INTO Votable VALUES (5, 'none :(', 'wow a cool votable5', 'gavin');

-- populate VotesOn
INSERT INTO VotesOn VALUES ('gavin', 1);
INSERT INTO VotesOn VALUES ('gavin', 2);
INSERT INTO VotesOn VALUES ('gavin', 3);
INSERT INTO VotesOn VALUES ('bob', 1);
INSERT INTO VotesOn VALUES ('helen', 1);

-- populate Popular
INSERT INTO Popular VALUES (100, 5);
INSERT INTO Popular VALUES (80, 4);
INSERT INTO Popular VALUES (50, 3);
INSERT INTO Popular VALUES (5, 1);
INSERT INTO Popular VALUES (10, 2);

-- populate Broadcast
INSERT INTO Broadcast VALUES (1, 'awesome broadcast 1', 2, 'gavin', 100);
INSERT INTO Broadcast VALUES (2, 'awesome broadcast 1', 3, 'gavin', 80);
INSERT INTO Broadcast VALUES (3, 'awesome broadcast 1', 4, 'gavin', 50);
INSERT INTO Broadcast VALUES (4, 'awesome broadcast 2', 4, 'bob', 5);
```

```sql
INSERT INTO Broadcast VALUES (5, 'awesome broadcast 3', 4, 'helen', 10);

-- populate Thumbnail
INSERT INTO Thumbnail VALUES (1, 1, "this is some image data");
INSERT INTO Thumbnail VALUES (2, 2, "this is some image data");
INSERT INTO Thumbnail VALUES (3, 3, "this is some image data");
INSERT INTO Thumbnail VALUES (4, 4, "this is some image data");
INSERT INTO Thumbnail VALUES (5, 5, "this is some image data");

-- populate Comment
INSERT INTO Comment VALUES (10, 1, 'none :(', '1st post', 'gavin');
INSERT INTO Comment VALUES (11, 2, 'none :(', 'thread from 1st post?',
'gavin');
INSERT INTO Comment VALUES (12, 1, 'none :(', '3rd post', 'bob');
INSERT INTO Comment VALUES (13, 1, 'none :(', '4th post', 'gavin');
INSERT INTO Comment VALUES (14, 1, 'none :(', '5th post', 'gavin');

-- populate Posts
INSERT INTO Posts VALUES (1, 'cats');
INSERT INTO Posts VALUES (2, 'cats');
INSERT INTO Posts VALUES (3, 'cats');
INSERT INTO Posts VALUES (4, 'dogs');
INSERT INTO Posts VALUES (5, 'dogs');
```