University of British Columbia, Vancouver
Department of Computer Science

CPSC 304 Project Cover Page

Milestone #: ____2____

Date: _____Oct 19_____

Group Number: ____119_____

Name	Student Number	CS Alias (Userid)	Preferred E-mail Address
Stuart Chen	48414957	r4v4j	stuartcc6@gmail.com
Allan Xing	28532901	p7i4r	allanx01@students.cs.ubc.ca
Eric Fu	57440844	y2c8w	ericfu55@student.ubc.ca

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.)

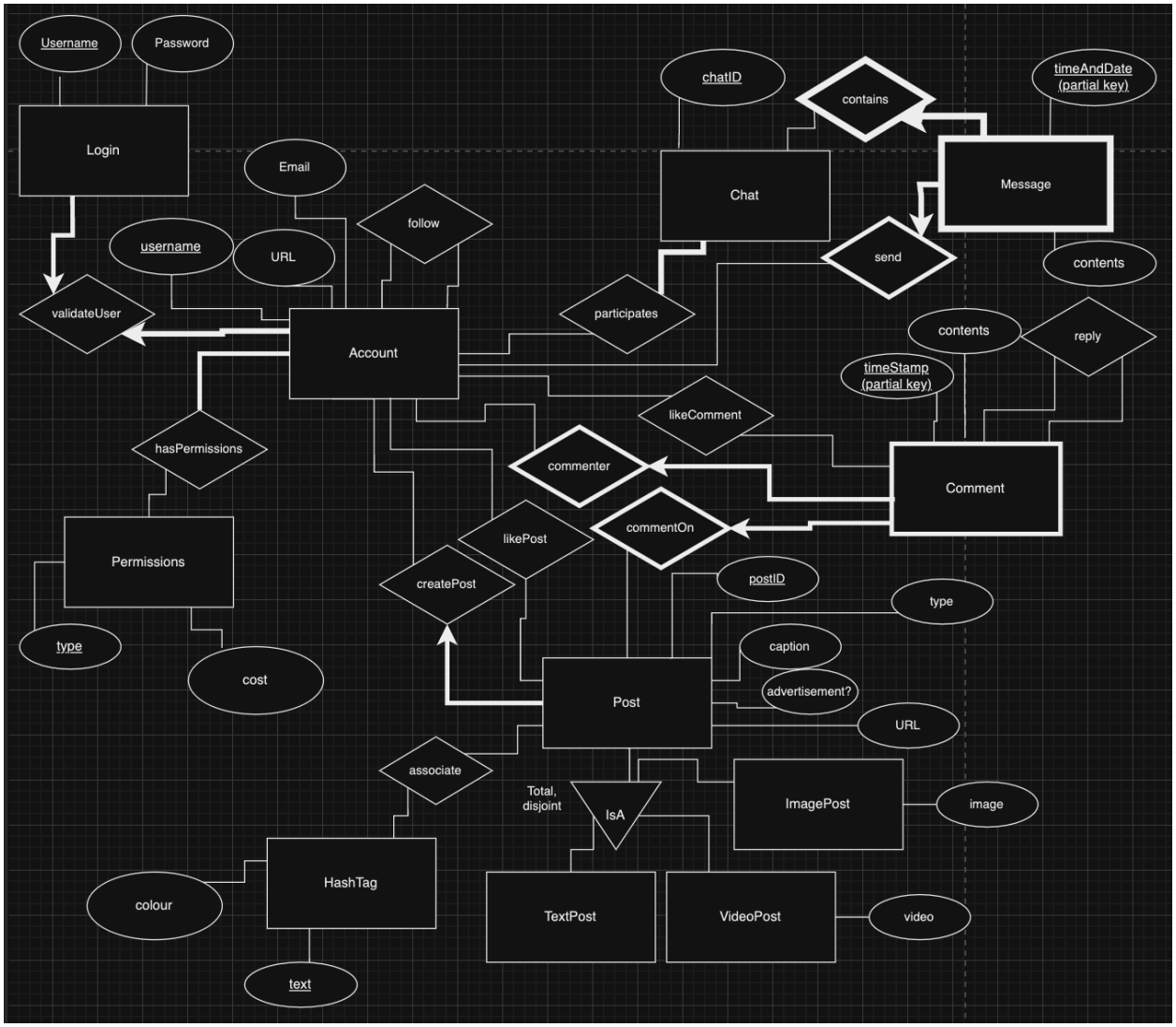
In addition, we indicate that we are fully aware of the rules and consequences of plagiarism, as set forth by the Department of Computer Science and the University of British Columbia

University of British Columbia, Vancouver
Department of Computer Science

Summary:

We're making a social media platform similar to applications like Instagram, Twitter/X, and Facebook. The primary function is to enable users to share and interact with posts, through liking or commenting. Other functions include messaging, advertised posts, and changing permissions of users (e.g. through blocking or subscribing to premium features).

ER diagram:



Changes since milestone 1:

- We merged permissions and submissions into one entity set (now called Permissions), as suggested by our mentor; this is a good change because it combines two similar entities and leaves us with less things to worry about in the big picture.
- We also removed the attribute "status" from Account because we realized that the information intended for "status" was already encapsulated by Permissions

- Also, we made it so that Login is uniquely identified by Username instead of Username + Password, since all usernames in the system should be unique.
- Added contents attribute for Message (the text that is contained in the message)
- We removed the relation useHashTag since that relation does not add any valuable functionality to our desired result.
- Added the attribute "type" to Post, which determines whether it is an ImagePost, VideoPost, or TextPost. This was to make it easier for us to retrieve information from these tables.

Schema:

IMPORTANT NOTES:
- We used the term "string" or "number" for many domains in this section, but we changed them to the appropriate VARCHAR, CHAR, INTEGER, etc. in the DDL tables below.)
- For all entity sets/relations that only have 1 candidate key, we only mentioned it as a Primary key and not a Candidate key.
    - E.g. Account has multiple candidate keys, so we listed them out explicitly. Whereas hasPermissions has only one candidate key so we simply listed it as the Primary key.

- Account(username: string, email: string, URL: string)
    - Primary key: username
    - Candidate keys: username, email, URL
    - Foreign keys: perms = Permissions.type NOT NULL
- Relationship: hasPermissions(user: string, perms: string)
    - Primary key: user
    - Foreign keys: user = Account.username
        - perms = Permissions.type NOT NULL
- Relationship: follow(follower: string, following: string)
    - Primary key: follower + following
    - Foreign keys: follower = Account.Username, NOT NULL
        - following = Account.Username, NOT NULL
- Login(Username: string, Password: string)
    - Primary key: Username
- Relationship: validateUser(accountUser: string, loginUser: string)
    - Primary key: accountUser, loginUser
    - Foreign keys: accountUser = Account.username UNIQUE, NOT NULL
        - loginUser = Login.Username UNIQUE, NOT NULL
- Permissions(type: string, cost: number)
    - Primary key: type
- Hashtag(text: string, color: string)
    - Primary key: text
- Relationship: associateHashtag(postID: number, hashTag: string)

        - Foreign keys: postID = Post.postID NOT NULL
            - hashTag = hastTag.text
- Post(postID: number, URL: string, caption: string, advertisement?: number, createdBy: string, type: number)
    - Primary key: postID
    - Candidate keys: postID and URL
    - Foreign key: createdBy = Account.username UNIQUE, NOT NULL
- Relationship: likePost(postID: string, acc: string)
    - Foreign keys:
        - postID = Post.postID
        - acc = Account.username
- IsA: TextPost(postID: number, content: string)
    - Primary key: postID
    - Foreign key: postID = UNIQUE, NOT NULL
- IsA: VideoPost(postID: number, content: BLOB)
    - Primary key: postID
    - Foreign key: postID = Account.username UNIQUE, NOT NULL
- IsA: ImagePost(postID: number, content: BLOB)
    - Primary key: postID
    - Foreign key: postID = Account.username UNIQUE, NOT NULL
- Weak entity: Comment(commentsOn: number, commenter: string, timeStamp: timestamp, contents: string)
    - Primary key: postID and timeStamp
    - Foreign keys: commentsOn = Post.postID
        - commenter = Account.username
- Relationship: reply(post: number, repliedTo: string, reply: string)
    - Primary key: post, repliedTo, and reply
    - Foreign keys: post = Post.postID
        - repliedTo = Comment.timeStamp
        - reply = Comment.timeStamp
- Relationship: likeComment(commentPost:number , commentTimeStamp: timestamp, acc: string)
    - Primary key: commentPost, commentTimeStamp, and acc
    - Foreign keys: commentPost = Post.postID
        - commentTS = Comment.timeStamp
        - acc = Account.username
- Chat(chatID: number)
    - Primary key: chatID
- Relationship: participates(chatID: number, acc: string)
    - Primary key: chatID
    - Foreign keys: chatID = Chat.chatID
        - acc = Account.username
- Weak entity: Message(chatID: number, account: string, timeAndDate: string)

    - Primary key: chatID and account and timeAndDate
    - Foreign key: chatID = Chat.chatID UNIQUE, NOT NULL
        - account = Account.username UNIQUE, NOT NULL

Functional Dependencies

Entities FD:

- Account(username: string, perms: string)
    - username -> perms
- Login(Username: string, Password: string)
    - Username -> Password
- Permissions(type: string, enables/disables: string, cost: number)
    - type -> enables/disables
    - type -> cost
- Hashtag(text: string)
- Post(postID: number, URL: string, caption: string, advertisement?: boolean, createdBy: string, type: number)
    - postId -> URL, createdBy, caption, advertisement?, type
    - URL -> postId, createdBy, caption, advertisement?, type
    - caption -> advertisement?
- TextPost(postID: number, content: string)
    - postId -> content
- VideoPost(postID: number, content: blob)
    - postId -> content
- ImagePost(postID: number, content: blob)
    - postId -> content
- Comment(commentsOn: number, commenter: string, timeStamp: string, contents: string)
    - commentsOn, commenter, timeStamp -> contents
- Chat(chatID: number, participant: string)
- Message(chatID: number, account: string, timeAndDate: string, content: string)
    - chatID, account, timeAndDate -> content

Relationship FDs:

- hasPermissions(user: string, perms: string)
    - No non-trivial FDs
- follow(follower: string, following: string)
    - No non-trivial FDs
- validateUser(accountUser: string, loginUser: string)

    - accountUser -> loginUser
    - loginUser -> accountUser
- associateHashtag(postID: number, hashTag: string)
    - No non-trivial FDs
- likePost(post: string, acc: string)
    - No non-trivial FDs
- reply(post: number, comment: string, reply: string)
    - post, comment -> reply
- likeComment(commentP:number , commentTS: string, acc: string)
    - No non-trivial FDs
- participates(chatID: number, acc: string)
    - No non-trivial FDs

Normalization

Due to efficient data design, all other tables are in 3NF or BCNF.

Post(postID: number, URL: string, caption: string, advertisement?: boolean, createdBy: string, type: string)

- FDs:
    - postId -> URL, createdBy, caption, advertisement?, type
    - URL -> postId, createdBy, caption, advertisement?, type
    - caption -> advertisement?
- This is not in 3NF or BCNF, since caption is not a superkey for the entity set
    - Decompose into Caption(<u>caption</u>, advertisement?) and Post(<u>postID</u>, URL, caption, createdBy, type)
        - The resulting two entity sets are in 3NF and BCNF

New tables:
Caption(caption: string, advertisement?: boolean)

- Primary key: caption

Post(postID: number, URL: string, caption: string, createdBy: string, type)

- Primary key: postID
- Candidate keys: postID and URL
- Foreign key: createdBy = Account.username UNIQUE, NOT NULL
-

University of British Columbia, Vancouver
Department of Computer Science

SQL DDL

```
CREATE TABLE Account (
        username VARCHAR(20) PRIMARY KEY,
        email VARCHAR(63),
        URL VARCHAR(255)
)

CREATE TABLE hasPermissions (
        user VARCHAR(20) PRIMARY KEY,
        perms VARCHAR(20) NOT NULL DEFAULT “normalUser”,
        FOREIGN KEY (user) REFERENCES
                Account(username),
                ON DELETE CASCADE
        FOREIGN KEY (perms) REFERENCES
                Permissions(type),
                ON DELETE SET DEFAULT
)

CREATE TABLE follow (
        follower VARCHAR(20),
        following VARCHAR(20),
        PRIMARY KEY (follower, following)
        FOREIGN KEY (follower) REFERENCES
                Account(username),
                ON DELETE CASCADE
        FOREIGN KEY (following) REFERENCES
                Account(username),
                ON DELETE CASCADE
)

CREATE TABLE Login (
        username VARCHAR(20) PRIMARY KEY,
        password VARCHAR(20)
)

CREATE TABLE validateUser (
        accountUser VARCHAR(20),
        loginUser VARCHAR(20),
        PRIMARY KEY (accountUser, loginUser),
        FOREIGN KEY (accountUser) REFERENCES Account(username),
                ON DELETE CASCADE
```

```
        FOREIGN KEY (loginUser) REFERENCES Login(Username),
                ON DELETE CASCADE
)

CREATE TABLE Permissions (
        type VARCHAR(20) PRIMARY KEY,
        cost INT NOT NULL
)

CREATE TABLE Hashtags (
        text VARCHAR(20) PRIMARY KEY,
        color CHAR(6)
)

CREATE TABLE associateHashtag(
        postID INT,
        hashTag VARCHAR(20),
        FOREIGN KEY (postID) REFERENCES Post(postID),
        FOREIGN KEY (hashTag) REFERENCES Hashtag(text)
)

CREATE TABLE Caption (
        caption VARCHAR(280) PRIMARY KEY,
        advertisement? BOOLEAN
)

CREATE TABLE Post (
        postID INT PRIMARY KEY,
        URL VARCHAR(255),
        caption VARCHAR(280),
        createdBy VARCHAR(20) NOT NULL,
        type INT,
        FOREIGN KEY (createdBy) REFERENCES Account(username)
)

CREATE TABLE likePost(
        postID INT PRIMARY KEY,
        acc VARCHAR(20) PRIMARY KEY,
        FOREIGN KEY (postID) REFERENCES Post(postID),
                ON DELETE CASCADE
        FOREIGN KEY (acc) REFERENCES Account(username),
                ON DELETE CASCADE
)
```

University of British Columbia, Vancouver
Department of Computer Science

```
CREATE TABLE TextPost(
        postID INT PRIMARY KEY,
        content: VARCHAR(280),
        FOREIGN KEY (postID) REFERENCES Post(postID)
                ON DELETE CASCADE
)

CREATE TABLE VideoPost(
        postID INT PRIMARY KEY,
        content: BLOB,
        FOREIGN KEY (postID) REFERENCES Post(postID)
                ON DELETE CASCADE

)
CREATE TABLE ImagePost(
        postID INT PRIMARY KEY,
        content: BLOB,
        FOREIGN KEY (postID) REFERENCES Post(postID)
                ON DELETE CASCADE

)

CREATE TABLE Comment(
        commentsON INT PRIMARY KEY,
        commenter VARCHAR(20),
        timeStamp TIMESTAMP PRIMARY KEY
        contents: VARCHAR(280)
)

CREATE TABLE reply(
        post INT PRIMARY KEY,
        repliedTo TIMESTAMP,
        reply TIMESTAMP,
        FOREIGN KEY (post) REFERENCES Post(postID),
                ON DELETE CASCADE
        FOREIGN KEY (repliedTo) REFERENCES Comment(timeStamp),
                ON DELETE CASCADE
        FOREIGN KEY (reply) REFERENCES Comment(timeStamp)
                ON DELETE CASCADE

)
```

```
CREATE TABLE likeComment(
        commentPost INT PRIMARY KEY,
        commentTimeStamp TIMESTAMP PRIMARY KEY,
        account VARCHAR(20) PRIMARY KEY,
        FOREIGN KEY (commentPost) REFERENCES Post(postID),
        FOREIGN KEY (commentTimeStamp) REFERENCES Comment(timeStamp),
        FOREIGN KEY (account) REFERENCES Account(username)
)

CREATE TABLE Chat (
        charID INT PRIMARY KEY
)

CREATE TABLE participates (
        chatID INTEGER PRIMARY KEY,
        acc VARCHAR(20) DEFAULT “user0”,
        FOREIGN KEY (chatID) REFERENCES
                Chat(chatID),
                ON DELETE CASCADE
        FOREIGN KEY (acc) REFERENCES
                Account(username),
                ON DELETE SET DEFAULT
)

CREATE TABLE Message (
        chatID INT,
        account VARCHAR(20) DEFAULT “user0”,
        timeAndDate TIMESTAMP,
        PRIMARY KEY (chatID, account, timeAndDate),
        FOREIGN KEY (chatID) REFERENCES Chat(chatID)
                ON DELETE CASCADE
        FOREIGN KEY (account) REFERENCES Account(username)
                ON DELETE SET DEFAULT
)
```

University of British Columbia, Vancouver
Department of Computer Science

Insert Statements

```
INSERT INTO Account(username, email, URL) VALUES
('user1', 'test1@gmail.com', 'http://www.example1.com'),
('user2', 'test2@gmail.com', 'http://www.example2.com'),
('user3', 'test3@gmail.com', 'http://www.example3.com'),
('user4', 'test4@gmail.com', 'http://www.example4.com'),
('user5', 'test5@gmail.com', 'http://www.example5.com');

INSERT INTO Permissions(type, cost) VALUES
('normalUser', 0),
('premiumUser', 10),
('admin', 20),
('editor', 5),
('viewer', 0);

INSERT INTO hasPermissions(user, perms) VALUES
('user1', 'normalUser'),
('user2', 'premiumUser'),
('user3', 'admin'),
('user4', 'editor'),
('user5', 'viewer');

INSERT INTO follow(follower, following) VALUES
('user1', 'user2'),
('user2', 'user3'),
('user3', 'user4'),
('user4', 'user5'),
('user5', 'user1');

INSERT INTO Login(username, password) VALUES
('user1', 'pass1'),
('user2', 'pass2'),
('user3', 'pass3'),
('user4', 'pass4'),
('user5', 'pass5');

INSERT INTO validateUser(accountUser, loginUser) VALUES
('user1', 'user1'),
('user2', 'user2'),
('user3', 'user3'),
('user4', 'user4'),
```

```
('user5', 'user5');

INSERT INTO Hashtags(text, color) VALUES
('Hashtag1', 'FF0000'),
('Hashtag2', '00FF00'),
('Hashtag3', '0000FF'),
('Hashtag4', 'FFFF00'),
('Hashtag5', 'FF00FF');
INSERT INTO associateHashtag(postID, hashTag) VALUES
(1, 'Hashtag1'),
(2, 'Hashtag2'),
(3, 'Hashtag3'),
(4, 'Hashtag4'),
(5, 'Hashtag5');

INSERT INTO Caption(caption, advertisement?) VALUES
('This is caption 1', TRUE),
('This is caption 2', FALSE),
('This is caption 3', TRUE),
('This is caption 4', FALSE),
('This is caption 5', TRUE);

INSERT INTO Post(postID, URL, caption, createdBy, type) VALUES
(1, 'http://example1.com', 'This is caption 1', 'user1', 0),
(2, 'http://example2.com', 'This is caption 2', 'user2', 1),
(3, 'http://example3.com', 'This is caption 3', 'user3', 2),
(4, 'http://example4.com', 'This is caption 4', 'user4', 0),
(5, 'http://example5.com', 'This is caption 5', 'user5', 1);

INSERT INTO likePost(postID, acc) VALUES
(1, 'user1'),
(2, 'user2'),
(3, 'user3'),
(4, 'user4'),
(5, 'user5');

INSERT INTO TextPost(postID, content) VALUES
(1, 'This is a text post 1'),
(2, 'This is a text post 2'),
(3, 'This is a text post 3'),
(4, 'This is a text post 4'),
(5, 'This is a text post 5');
```

University of British Columbia, Vancouver
Department of Computer Science

```
INSERT INTO VideoPost(postID, content) VALUES
(1, 'BINARY_DATA'),
(2, 'BINARY_DATA'),
(3, 'BINARY_DATA'),
(4, 'BINARY_DATA'),
(5, 'BINARY_DATA');

INSERT INTO ImagePost(postID, content) VALUES
(1, 'BINARY_DATA'),
(2, 'BINARY_DATA'),
(3, 'BINARY_DATA'),
(4, 'BINARY_DATA'),
(5, 'BINARY_DATA');

INSERT INTO Comment(commentsON, commenter, timeStamp, contents) VALUES
(1, 'user1', 2023-10-10 00:00:00, 'This is a comment 1'),
(2, 'user2', 2023-10-10 00:00:01, 'This is a comment 2'),
(3, 'user3', 2023-10-10 00:00:02, 'This is a comment 3'),
(4, 'user4', 2023-10-10 00:00:03, 'This is a comment 4'),
(5, 'user5', 2023-10-10 00:00:04, 'This is a comment 5');

INSERT INTO reply(post, repliedTo, reply) VALUES
(1, 2023-10-10 00:00:00, 2023-10-10 00:00:01),
(2, 2023-10-10 00:00:01, 2023-10-10 00:00:02),
(3, 2023-10-10 00:00:02, 2023-10-10 00:00:03),
(4, 2023-10-10 00:00:03, 2023-10-10 00:00:04),
(5, 2023-10-10 00:00:04, 2023-10-10 00:00:05);

INSERT INTO likeComment(commentPost, commentTimeStamp, account) VALUES
(1, 2023-10-10 00:00:00, 'user1'),
(2, 2023-10-10 00:00:01, 'user2'),
(3, 2023-10-10 00:00:02, 'user3'),
(4, 2023-10-10 00:00:03, 'user4'),
(5, 2023-10-10 00:00:04, 'user5');

INSERT INTO Chat(chatID) VALUES
(1),
(2),
(3),
(4),
(5);

INSERT INTO participates(chatID, acc) VALUES
```

(1, 'user1'),
(2, 'user2'),
(3, 'user3'),
(4, 'user4'),
(5, 'user5');

INSERT INTO Message(chatID, account, timeAndDate) VALUES
(1, 'user1', 2023-10-10 00:00:00),
(2, 'user2', 2023-10-10 00:00:01),
(3, 'user3', 2023-10-10 00:00:02),
(4, 'user4', 2023-10-10 00:00:03),
(5, 'user5', 2023-10-10 00:00:04);