Edona Mujaj U32335536
Selin Aslangul

The final report should include the final schema, additional assumptions that you make during the implementation, and the limitations of your system (what functions are implemented and what are not)

Some limitations and assumptions of our system:
Assumption 1: User will only attempt to like a photo once
Assumption 2: assumption that is made in our application is that photos are only added to pre existing albums. This may also count as a limitation of our system as you cannot upload a photo and create an album at the same time, first the album needs to be created and then photos can be uploaded.
Tags are assumed to be separated by photos
Another assumption that is made is that if someone is trying to add a friend, the friend already has an account. This also has the limitation that you run into an error message if you were to mistype an email address or something like that. Additionally, friends can only be searched by exact email addresses, and not names etc.
Assumption/limitation: assuming all comments are from users/ comments can only be made by users

Additionally, if users delete albums or photos, we have no way of freeing up that album ID number or the photo number. This affects the user contributions as well (i.e user uploads 2 photos but deletes one, we still count 2 photos)

Another limitation that could possibly be faced is when we have an overload of users/photos/tags to index. This number is currently an unknown amount but I figured it was worth mentioning as a limitation of the project implementation
Limitation- comments cannot be edited or deleted
Another limitation is navigation out of certain pages i.e album and photo upload, requires clicking thought the navi or just changing the html template youre operating on

The final Schema of our project:

```
CREATE DATABASE IF NOT EXISTS photoshare;
USE photoshare;
DROP TABLE IF EXISTS Pictures CASCADE;
DROP TABLE IF EXISTS Users CASCADE;

CREATE TABLE Users (
    user_id int4  AUTO_INCREMENT,
    email varchar(255) UNIQUE,
    firstName VARCHAR(100),
    lastName VARCHAR(100),
    birthDate DATE,
    homeTown VARCHAR(100),
    homeTownState VARCHAR(100),
    hometownCountry   VARCHAR(100),
    currentCity VARCHAR(100),
    currentState VARCHAR(100),
    currentCountry VARCHAR(100),

    password varchar(255) NOT NULL,
  CONSTRAINT users_pk PRIMARY KEY (user_id)
);

CREATE TABLE Friends(
  user_id1 INTEGER,
  user_id2 INTEGER,
  PRIMARY KEY (user_id1, user_id2),
  FOREIGN KEY (user_id1)
  REFERENCES Users(user_id),
  FOREIGN KEY (user_id2)
  REFERENCES Users(user_id)
);

CREATE TABLE Pictures
(
  picture_id int4  AUTO_INCREMENT,
  user_id int4,
  imgdata longblob ,
  caption VARCHAR(255),
  INDEX upid_idx (user_id),
  CONSTRAINT pictures_pk PRIMARY KEY (picture_id)
);
CREATE TABLE Albums(
  albums_id INTEGER AUTO_INCREMENT,
  name VARCHAR(100),
  date DATE,
  user_id INTEGER NOT NULL,
  PRIMARY KEY (albums_id),
  FOREIGN KEY (user_id)
  REFERENCES Users(user_id)
);
CREATE TABLE Tags(
  tag_id INTEGER AUTO_INCREMENT,
  name VARCHAR(100),
  PRIMARY KEY (tag_id)
);
CREATE TABLE Tagged(
  photo_id INTEGER,
  tag_id INTEGER,
  PRIMARY KEY (photo_id, tag_id),
  FOREIGN KEY(photo_id)
  REFERENCES Photos (photo_id) ON DELETE CASCADE,
  FOREIGN KEY(tag_id)
  REFERENCES Tags (tag_id)
);
CREATE TABLE Comments(
  comment_id INTEGER AUTO_INCREMENT,
  comment VARCHAR(100),
  photo_id INTEGER NOT NULL,
  user_id INTEGER NOT NULL,
  PRIMARY KEY (comment_id),
  FOREIGN KEY (photo_id) REFERENCES Photos (photo_id) ON DELETE CASCADE,
  FOREIGN KEY (user_id) REFERENCES Users (user_id)
);

CREATE TABLE Likes(
  like_id INTEGER AUTO_INCREMENT,
  photo_id INTEGER NOT NULL,
  user_id INTEGER NOT NULL,
  PRIMARY KEY (like_id),
  FOREIGN KEY (photo_id) REFERENCES Photos (photo_id) ON DELETE CASCADE,
  FOREIGN KEY (user_id) REFERENCES Users (user_id)
);

INSERT INTO Users (email, password) VALUES ('test@bu.edu', 'test');
INSERT INTO Users (email, password) VALUES ('test1@bu.edu', 'test');
```