SnapShots

**September 24, 2021**



**Team 28**

Alex Goswick

Rachel Hawley

Samara Saquib

Evan Shaw

**Index**

[**Purpose**](#_e87syq8oiitr) **3**

[Functional Requirements](#_w1isugjvm5g2) 3

[Non-Functional Requirements](#_a6uuqotyscbm) 5

[**Design Outline**](#_ogp52fyi6djo) **9**

[High Level Overview](#_m85amwc9d31m) 9

Sequence of Events Overview 10

[**Design Issues**](#_ksmxnrgo4jdp) **11**

[Functional Issues](#_wsgvlwo6y3on) 11

[Non-Functional Issues](#_73vat9zevn5r) 12

[**Design Details**](#_am6vde4xv49p) **13**

[Class-Level Design Diagram](#_eznadpfrjqrm) 13

[Description of Data Classes and Their Interactions](#_ajvtuhiheohc) 14

[Sequence of Events Diagrams](#_ar4l4ljtkggq) 16

# 

# 

# 

# 

# 

# 

# Purpose

With so many different types of devices (Mac, PCs, iPhones, Androids) out today, sharing photos outside of texting people is difficult. Texting or sending photos over messaging apps compresses them and does not maintain the original quality. While applications such as Google Photos exist, they do not allow users to upload pictures in the original quality without paying a fee for storage space. In order to effectively transfer pictures to other people, they may have to resort to using old-fashioned devices such as USB flash drives which have become increasingly obsolete in the Cloud age. Most new laptop devices no longer contain ports for USB flash drives because of their declining relevancy.

In addition, USB flash drives often have to be formatted when used across different operating systems. Our application, Snapshots, seeks to solve this issue of sharing photos across different platforms. Snapshots will provide a secure, storage space for users to upload their photos in its original quality. It will also allow users to share their photos with friends and family members who have different devices. Snapshots will also support photo editing and social media sharing services to convenience users in having several tools within one application.

## Functional Requirements

* 1. User can store photos in a central, secure location that is compatible across platforms

**As a user,**

* + 1. I want to store my pictures in an application that is easy to access
    2. I want to secure my photos with a password
    3. I would like to enable multi-factor authentication
    4. I want to upload and view pictures
    5. I would like to easily backup my photos, without connecting it to any Cloud services
    6. I would like to store an email and phone number with my account
    7. I want to download photos in multiple formats
    8. I want to upload Apple Live Photos
  1. User can enhance their photos with editing tools

**As a user,**

* + 1. I want to fix red-eye effects using the application
    2. I want to edit the colors and adjust color curves
    3. I want to retouch and blur photos
    4. I want to edit photos without losing the original, unretouched photograph
    5. I want to create duplicates of photos to edit them
    6. I want to crop photos
  1. User can share photos with other people

**As a user,**

* + 1. I want to easily share my photos with other people
    2. I would like to directly share my photos to a social media app
    3. I would like to receive photos from other people’s Snapshots accounts
    4. I would like the ability to save photos that are sent to me from other people, both to my Snapshots account and to my device
    5. I would like to select multiple photos at a time to send to others
  1. User can use mobile devices to access their photos

**As a user,**

* + 1. I want to view photos from my mobile device
    2. I would like to upload my photos from my mobile device
    3. I would like my photos to upload automatically in the background of my mobile device
    4. If I have an Apple device, I would like my Live photos to upload
    5. I would like my screenshots to upload in a separate folder

## Non-Functional Requirements

* 1. Client Requirements

**As a developer,**

* + 1. I want the application to be used across platforms on desktops
    2. I want the application to be compatible with mobile devices
    3. I want the application to ask for the client’s permission before accessing their device’s photo library
    4. I want the application to ask for the client’s permission before accessing their location tags in photos
    5. I would like it to be the client’s responsibility to input the correct password
  1. Server Requirements

**As a developer,**

* + 1. I want the server to store photo metadata and user data in a relational database
    2. I want the server to respond to requests quickly to facilitate a smooth client user experience
    3. I want the server to keep track of metadata so that photos can be sorted in real time without reading in every file
    4. I want the server to be available without downtime
    5. I want the server to be able to handle multiple users uploading and editing files without any conflicts or data corruption
  1. Design Requirements

**As a developer,**

* + 1. I want the client applications and server to share a common set of classes to facilitate API access to the server
    2. I want a continuous integration setup where the server is automatically built as a Docker container and deployed to the development server
    3. I would like the server to handle at least 10 per library
    4. I would like the application to be compatible across different operating systems
    5. I would like the application to be able to upload and transfer photos without the user worrying about the type of device it is being sent to
  1. Performance Requirements

**As a developer,**

* + 1. I want the server to handle photo upload as fast as network speeds will allow
    2. I want the server to handle errors without interrupting upload or download progress of photos
    3. I want the server to handle at least 30 requests per each 100 milliseconds
    4. I would like the home page to display the top of the photo library, with the photos loading without delay as the user scrolls through the library
    5. I would like the server to send accurate error messages to the client when incorrect passwords are inputted to ensure security
  1. Appearance Requirements

**As a developer,**

* + 1. I want the application to be user friendly for people of all ages
    2. I want the application to be aesthetically simple and not cluttered with too many features
    3. I want the application to have subsections for different features to keep the Home page clean
    4. I want the Home page to display the photographs uploaded, with the option to scroll through them with a scroll bar
    5. I would like a color scheme that is professional but not neutral so that the application appeals to younger users
    6. I would like there to be a Sort button for users to choose how to sort their photos in their photo library display
  1. Security Requirements

**As a developer,**

* + 1. I want the app to ask for the user’s permission before accessing their photos, data, or location
    2. I want the application to be secured with the user’s password, email, and phone number
    3. I want the storage system to not necessarily rely on Cloud services that are susceptible to security breaches
    4. I want the application to allow setting up multi-factor authentication
    5. I want the application to be safe from third party apps or social media sites from posting without permission

# 

# 

# 

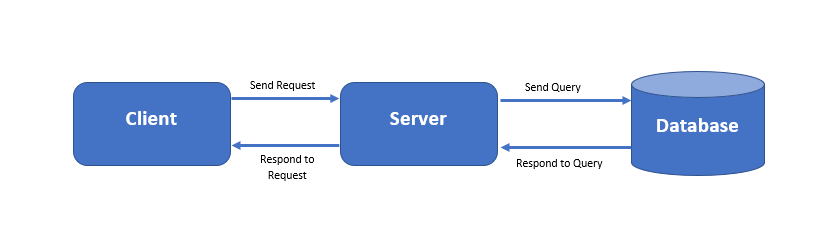
# 

# 

# Design Outline

## High Level Overview

This project will be an application that allows users to upload and share photos as well as functionality for creating albums and photo editing. The application will be utilizing the client-server model. The server will handle requests from any number of clients that want to access the database. The database stores all user and photo data and will receive and respond to queries from the server.



1. Client
   1. Client is the user interface for our service
   2. Client sends requests to the server
   3. Client receives responses from the server and parses and formats them to be displayed
2. Server
   1. Server will handle requests from the client
   2. Server will send a query to the database for information
   3. Server will send data received from the database to the client
3. Database
   1. Database stores all user data. This includes account information, photos, etc.
   2. Database responds to queries from the server and sends the appropriate data

## Sequence of Events Overview

The main components of the software are the client, server and the database. When a user opens the application, the login sequence is started. The client sends a request to the server. The server then sends a query to the database for password information. The database will respond with password information for the server to check, or an error. After the login process, the client sends requests to the server for actions like uploading photos and sharing photos with other users. For uploading photos, the server will send a request to add data to the database, and the database will respond with a success or error message. The server then sends a success or error message to the client, and the Photo Library page will be reloaded. For sharing photos with other users, the client sends a request to the server with the username of the user that the photos will be shared with. The server queries the database for stored photo data. The database will respond with the stored photo data or an error message. The server will then make a request to the database to add the shared photos to the target user’s stored photo data. The database will respond with a success or error message. The server will respond to the client with a success or error message, and the client will reload the Photo Library page. Sequence diagrams of these actions can be found on page 15.

# 

# 

# 

# Design Issues

## Functional Issues

* 1. Do users need to sign in each time they open the application?

**Option 1:** Users may save their password on their device

**Option 2:** Users may select to stay signed in

**Option 3:** Users will have to sign in with their password each time

**Decision:** We will allow users to select the option to stay signed in.

* 1. How will users share photos with their friends?

**Option 1:** They can share it within the app to other friends who also have the app

**Option 2:** They can send it via email

**Decision:** Users can share it within the app to other friends who also have the app. This way, photos won’t be compressed within an email.

* 1. How will the photos be displayed?

**Option 1:** They will be sorted by date

**Option 2:** They can be sorted by albums

**Option 3:** They will be sorted by location tags

**Decision:** They will be sorted by date by default, but users will have the ability to customize the sorting with locations, albums, etc.

## Non-Functional Issues

1. What language will be used for Snapshots??

**Option 1:** C

**Option 2:** Java

**Option 3:** C++

**Option 4:** HTML

**Option 5:** JavaScript

**Option 6:** REACT

**Decision:** We will use Java to program the backend application so that it can be run on any server. We will use REACT to program the frontend.

1. How will our application be hosted?

**Option 1:** AWS Server

**Option 2:** Linux

**Decision:** We will use Linux to host our application.

1. What database will be used?

**Option 1:** my SQL database

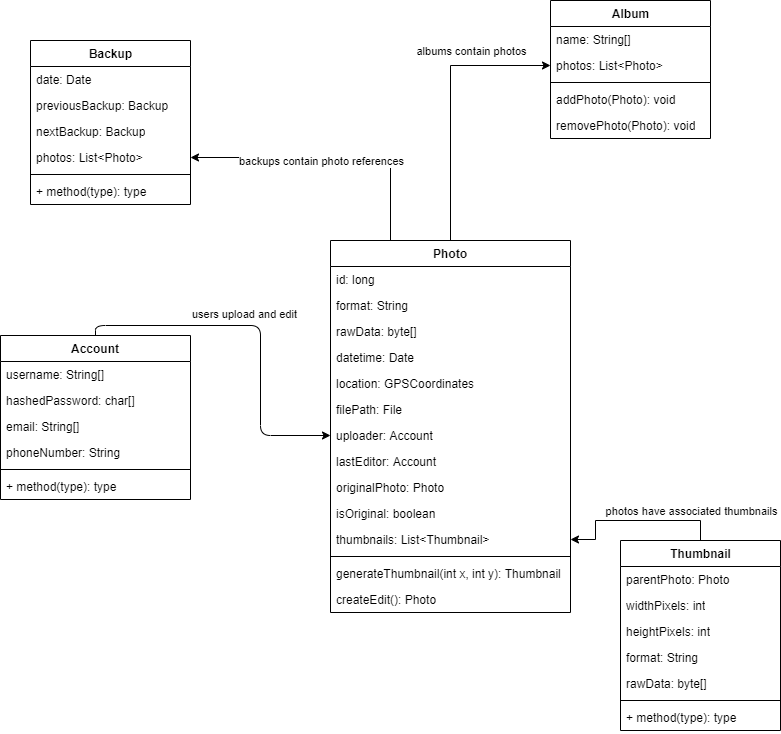
**Option 2:** MongoDB database

**Decision:** We will use mySQL since it stored data in tables and rows, which seems more appropriate for a photo storage application.

# 

# Design Details

## Class-Level Design Diagram



## 

## Description of Data Classes and Their Interactions

Description of Data Classes and Their Interactions

Our data class design is primarily based on the structure of data that will be stored in the MySQL relational database. The classes are the core data components.

Photo

* Represents a single photo in the library
* Encapsulates relevant metadata about the photo such as date taken and gps location
* Can represent different photo formats
* Created when photos are uploaded or when a new edit is created
* When an edit is made, a photo is copied to facilitate the non-destructive editing, then linked to the original

Thumbnail

* Represents a Thumbnail of a photo, and is always associate with Photo
* Thumbnails are generated when images are viewed in browsing mode
* Multiple Thumbnails may be associated with an image
* Can be variable in size

Album

* Is a collection of pictures from the library
* Can be named
* Does not contain duplicates
* Shared between users

Account

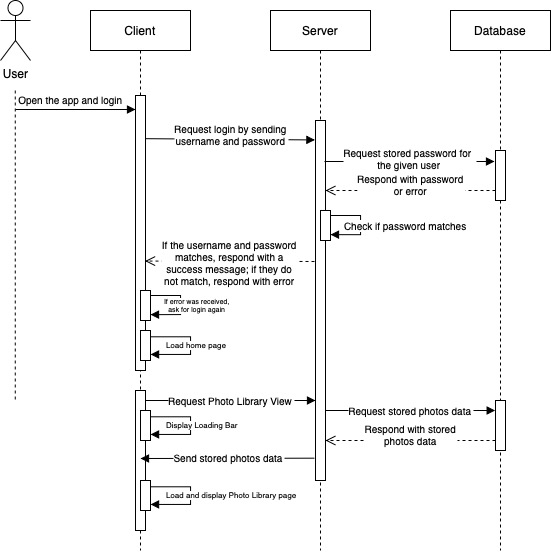
* Each account is for a single user
* Contains account username, hashed password in the database
* User token must be obtained to log in

Backup

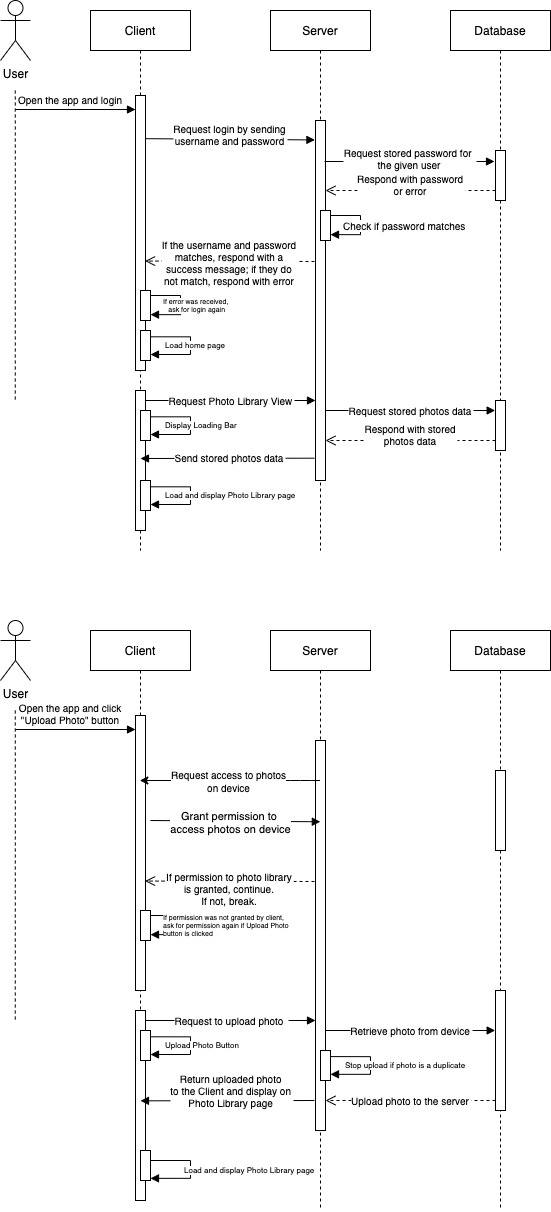
* Each backup object is specific to a date, enabling roll back to any date a backup was recorded on
* A backup keeps track of the next and previous backup, and the changes between backups
* Backups can be created automatically on a set time period, or created manually

## Sequence of Events Diagrams

**Sequence of events when user starts the application:**



**Sequence of events when user uploads a photo:**



**Sequence of events when user shares a photo:**

