# 3.2 Data Dictionary

Collection: user\_profile

|  |  |  |
| --- | --- | --- |
| **Field** | **Data Type** | **Description** |
| login\_id | String | Unique identity used to enter into the system |
| display\_name | String | Name of the user used to show in system |
| password | String | Hash value of secret key used by the user to log-in to system |
| profile\_photo | String | Filename of an image file used by the user as his/her profile picture |

Collection: photo\_collection

|  |  |  |
| --- | --- | --- |
| **Field** | **Data Type** | **Description** |
| login\_id | String | Used as identity of album folder owned by the user |
| filename | String | Filename of a photo uploaded by the user |
| ISO | String | Technical aspects of a photo |
| aperture | String |
| shutter | String |
| EV | String |
| likeCount | Integer | Total number of like response on a photo |
| likeBy | Array of string | Login ID of user(s) who response like a photo |
| raw1 | Raw photo object | Raw photos used to compose the photo in this record |
| raw2 | Raw photo object |

Object: raw photo

|  |  |  |
| --- | --- | --- |
| **Field** | **Data Type** | **Description** |
| filename | String | Filename of a raw photo uploaded by the user |
| ISO | String | Technical aspects of a photo |
| aperture | String |
| shutter | String |
| EV | String |

# 3.3 Assumption & Limitation

As a pilot project, few assumptions have been made to simplify system implementation process. Those assumptions are listed as below: -

* Each user is assumed to have one and only one album
* Each multi-exposure photo has maximum of 2 raw photos
* This project is assumed as low usage volume application

Due to above assumptions, system inherits with below limitation: -

* Each photo belongs to one and only one album. It cannot be shared among different album.
* System will create an album for each new user, and cannot be changed. User do not have rights to create any additional album.
* System only provides two slots for each photo to upload raw photos in photo upload module
* No performance issue is considered when design and implement this application

# 7. Security Control

To prevent application from unauthorized or unauthenticated use, certain security measurements are enforced.

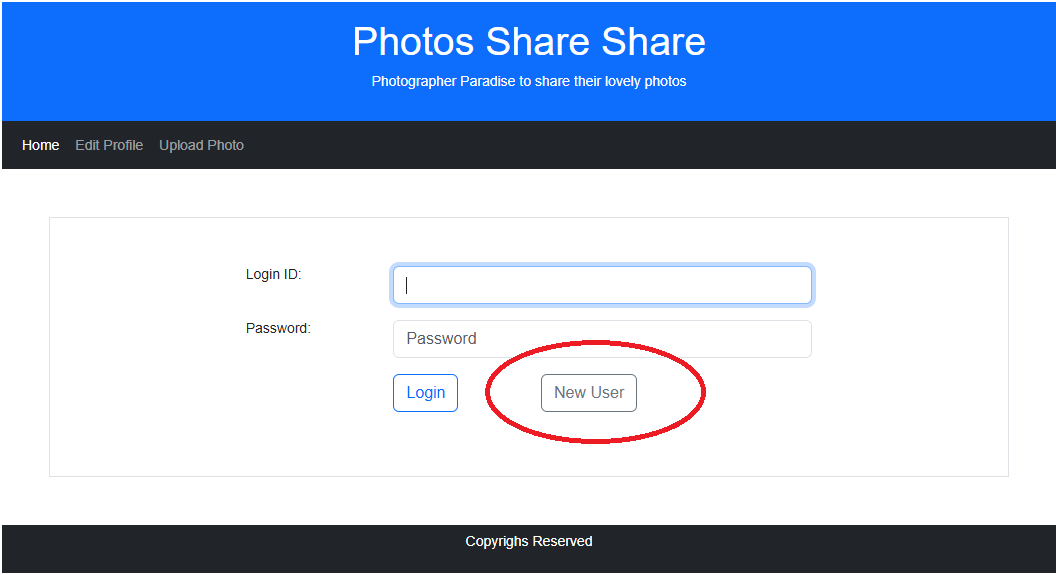
* User password is hashed before stored into database. It prevents anyone to spy on someone password by directing retrieving data from database.
* As /public folder is opened for anyone to access. If image files are kept under this folder, anyone can by-pass the application to access those files directly. To prevent this, A /data folder separated from /public folder is used for image files storage. Outsiders has no access rights to this folder. When application accesses those files, API is used to re-route access path to /data folder.

# 3.3 Brief Description of System Functionality

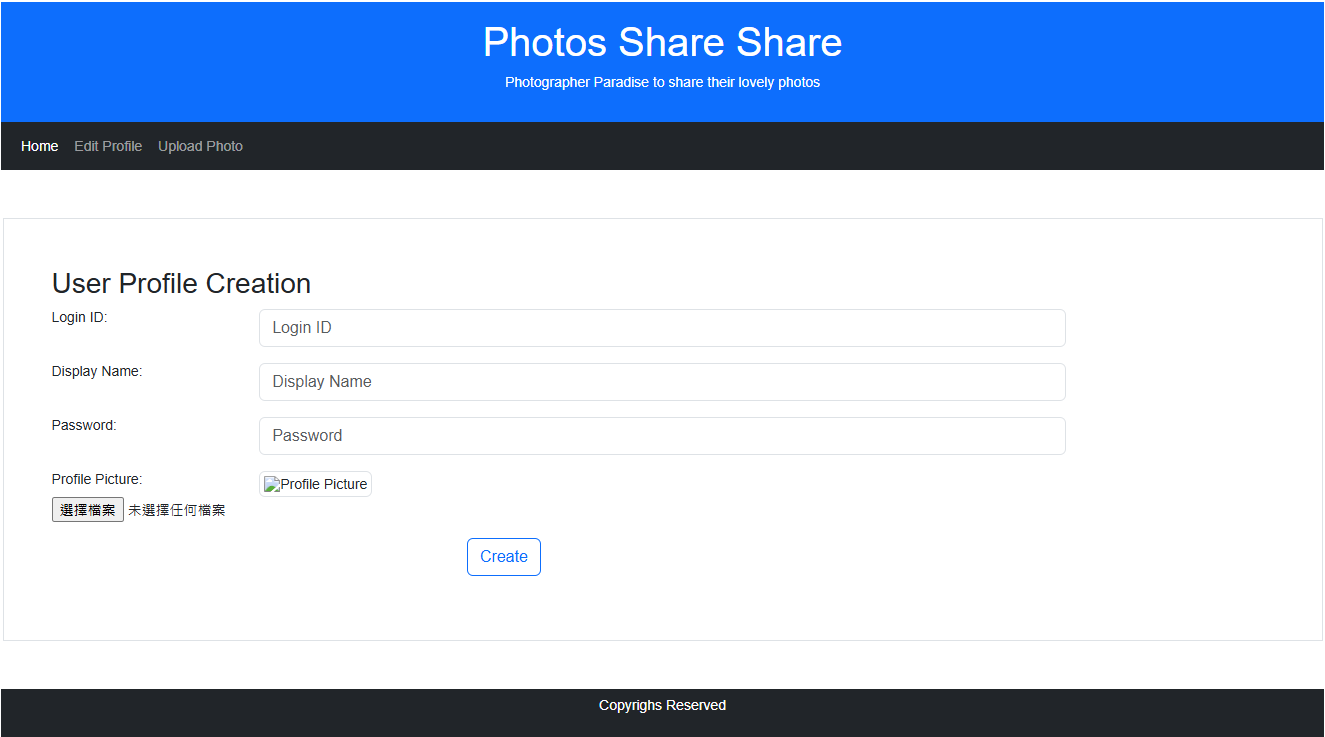
* User Account Management
  + System allows user to create/modify/remove its own user record. Besides of user ID, system will also record his/her chosen name showed in system and profile picture (optional).
* Album Management
  + Each user has its own album. User can upload its personal photo into its own album. Photo in the album can also be removed.
* Photo Browsing
  + User can browse all albums recorded in the system. At photo detail level, user can also view raw photos of those multi-exposure photo. Like icon is provided for user to show appreciation on any photo.

# 4. User Management

### Sign Up



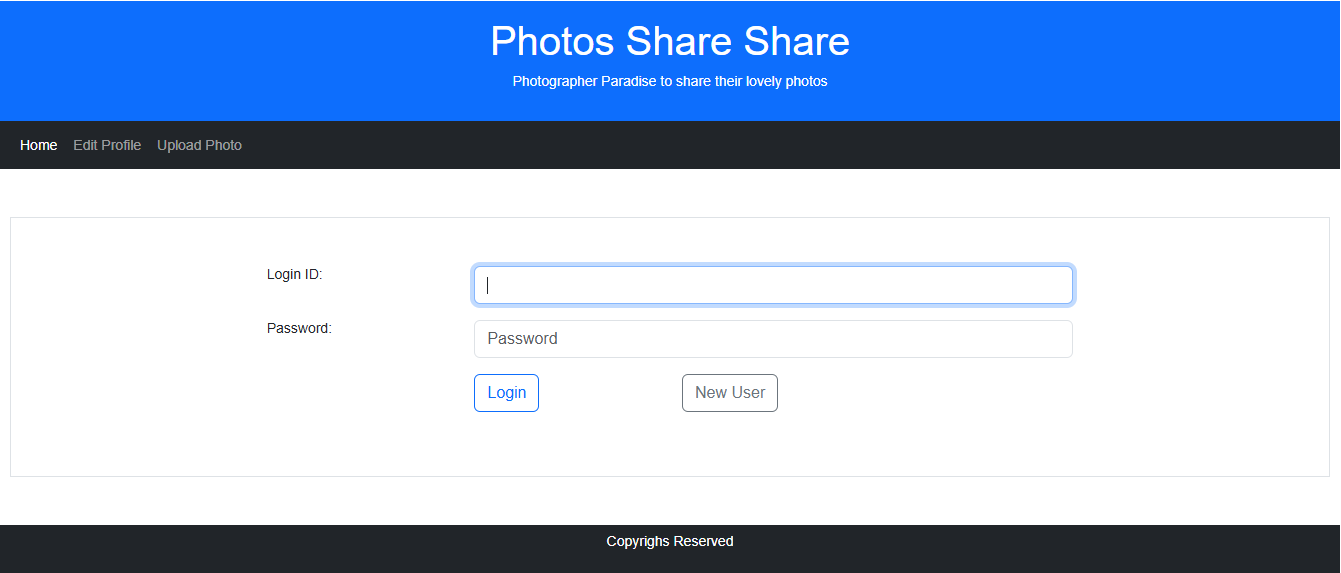
For new user who do not have account in the system, click <New User> button in log-in screen. It will lead to below new user creation screen.



Except for profile picture, all fields are mandatory. After user input all required information, press <Create> button to generate a user record. If success, system will direct to login screen then.

### Login and Logout

To log-in into system, enter log-in and password in log-in screen. Then press <Log-in> to proceed

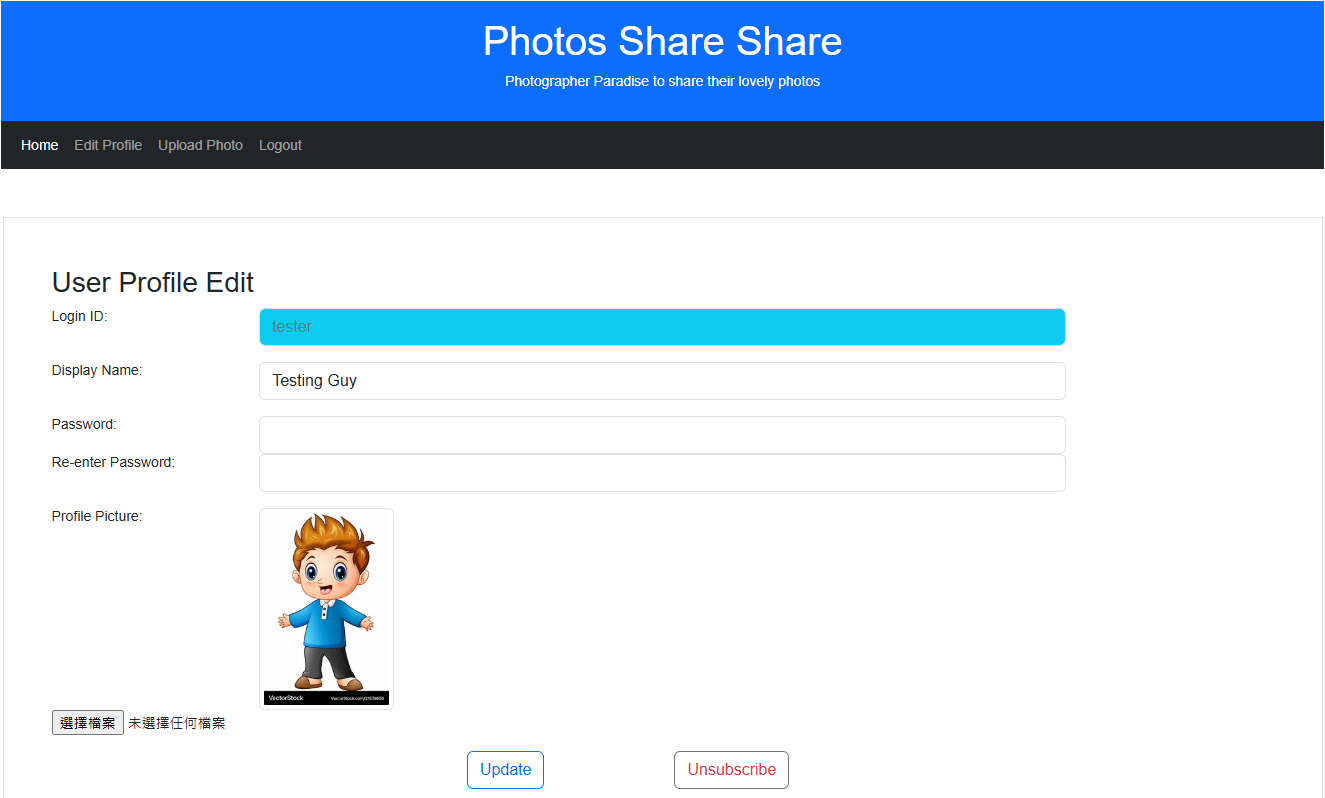


In main menu, there is a <Logout> menu item. Press it to log-out from system. System will then direct to log-in screen.

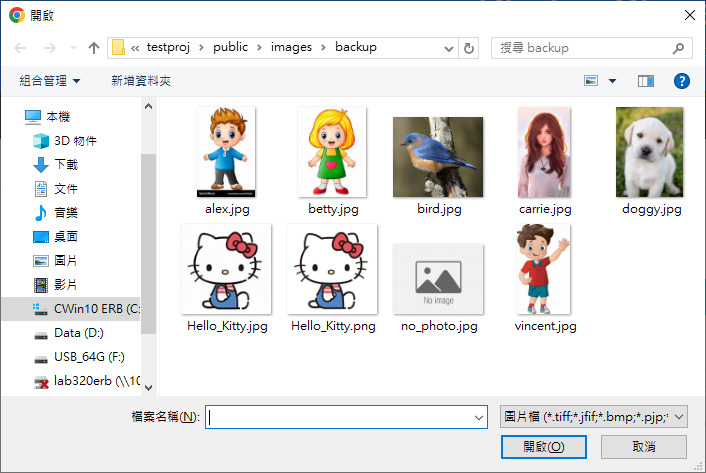


### User Profile Edit

Created user profile can be edited by its owner. After log-in to system, press <Edit Profile> menu option will lead to user profile edit screen, similar as below: -



To change profile picture, press <Choose File> button. A file selection dialog box will pop-up.

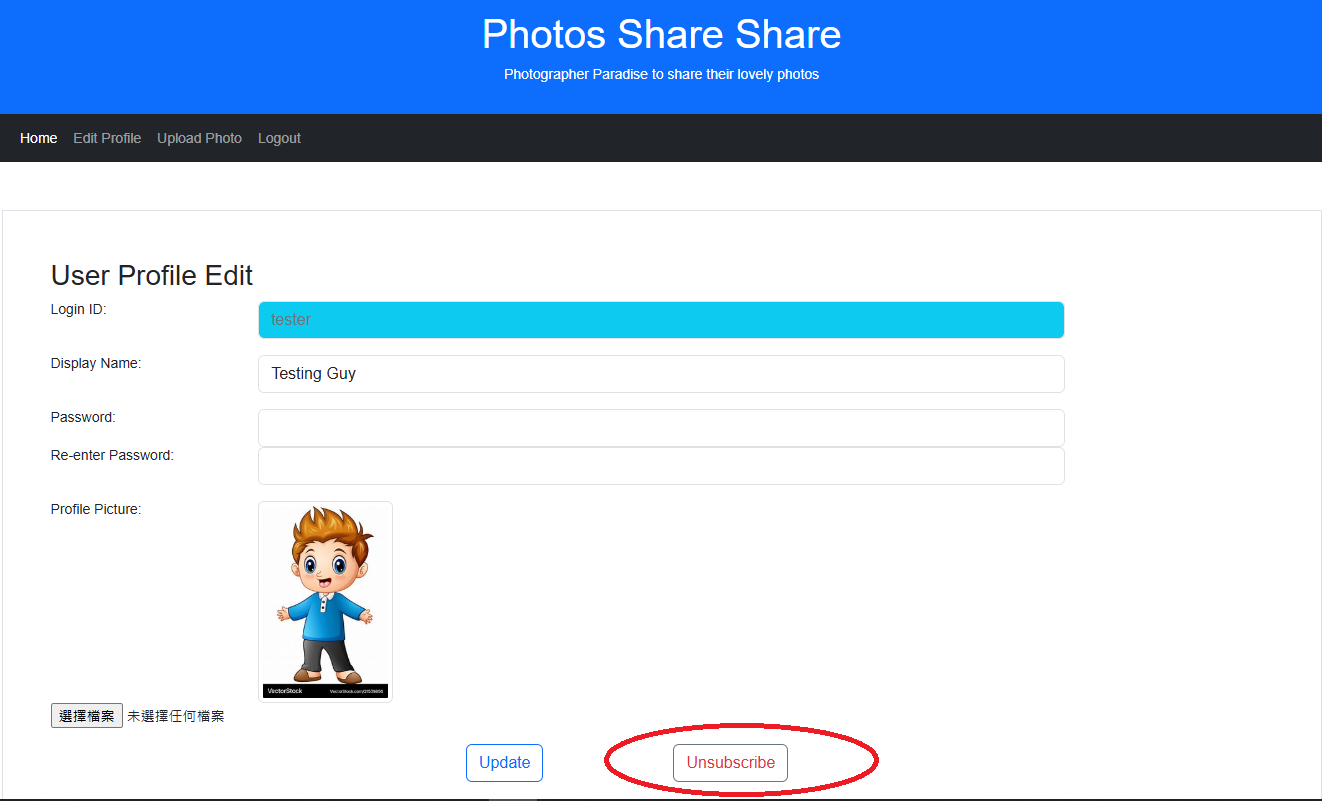


choose an image file, then press <Open> button

After necessary changes has been made, press <Update> to save changes.

### Un-subscription

In user profile edit screen, there is a <Unsubscribe> button. Click to remove user’s own user record from system.



Please note that all uploaded photos will be removed also. This process is IRREVERSIBLE!

After process done, system will direct to log-in page.

### 6. API Endpoints

#### User Management

* GET /login
  + Present log-in page to front-end.
  + If URL carries unsuccessful log-in status of last log-in, corresponding error message will be embedded in the page.
* POST /login
  + Retrieve corresponding record from database with login-ID as parameter
  + If found, check if input password matches with database record
  + If password match, direct to system main page. Also, generate a session variable to carry login-ID
  + Otherwise, show log-in page again. URL is formulated to carry unsuccessful log-in reason.
* GET /login/logout
  + Clean-up login-ID session variable
  + Direct to log-in page
* GET /user/addUser
  + Present add new user page to front-end
* POST /user/addUser
  + Generate hash value for input password
  + Create a user record in database, per user inputs
  + Create an album folder under /data
  + If profile picture is provided, store into the user’s album folder
* GET /user/editUser
  + Retrieve user record from database
  + Present edit user page to front-end, with retrieved data as default value of form fields (except password field)
* POST /user/editUser
  + Generate hash value for input password
  + Update user record per user inputs
  + If new profile picture provided, remove old image file (if any) and store new image file into the user’s album folder.
  + Direct to main page
* GET /user/delUser
  + Remove all photo collection record(s) from database
  + Remove user profile record from database
  + Remove user’s album folder