School of Computer Science and Engineering
Faculty of Engineering
UNSW Australia

PhotoPro
Project Report

By
COMP3900-F11A-Tofu

Name	zid	Email	Role
Airi Simonsen	z5127033	z5127033@unsw.edu.au	Developer
Peter Nguyen	z5061984	z5061984@unsw.edu.au	Developer + Scrum Master
Timothy Ryan	z5080316	z5080316@unsw.edu.au	Developer
Wendy Huynh	z5114100	z5114100@unsw.edu.au	Developer
Zachary Sanchez	z5194994	z5194994@unsw.edu.au	Developer

Submitted as a requirement for

COMP3900: Computer Science Capstone Project

Project Submission Date: Monday, 16th November 2020.

Trimester 3 2020

Lina Zhang

Overview

Architecture and Design of the System

The system's business logic is centred around the sales of photographic works uploaded by users (contributors). When a user (explorer) is looking at an image, they can see a preview of the original image, with a watermark over it. Upon payment by the explorer, they are then able to download the original unwatermarked image. It is this functionality that allows the site to generate revenue. In addition to this business logic is the ability of explorers to like, comment and search for contributors, with the intention that these actions allow users to feel as part of a community.

The user will have the ability to be both an explorer and contributor of the PhotoPro App. The user will have the opportunity to login or sign up, filling in their personal and billing details, to become a user of the PhotoPro App. Once the user has successfully logged in. If the user decides to pursue a role as an explorer, the user will be able to search photos that they like, select the respective photos and like and/or comment on the watermarked photo.

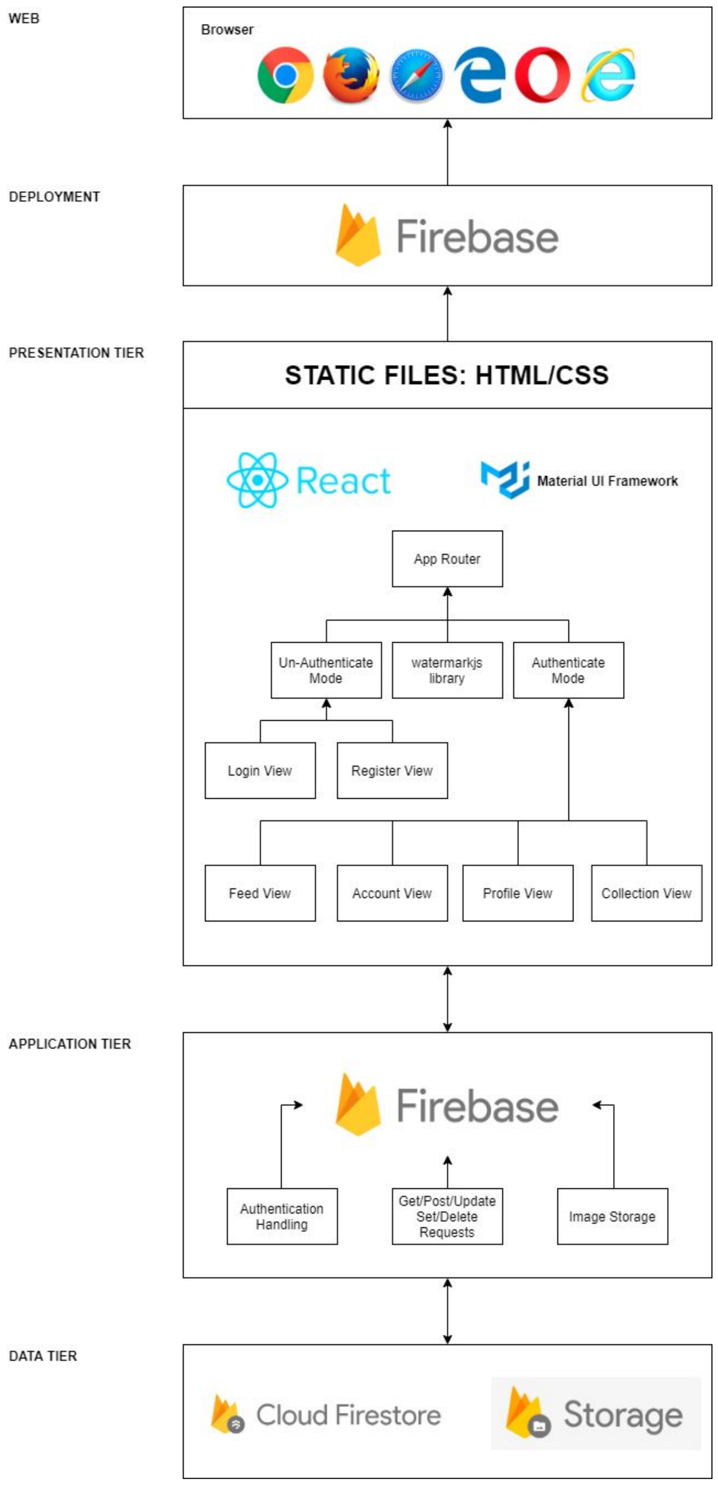
The user will be able to have an option to buy the photo where they will be taken to the payment process where if the transaction is successful, the user will be able see and download the non-watermarked version of the photo The explorer will be able to look at their photo feed where they can repeat the previous process whilst seeing similar photos based on photos that they previously liked and/or bought.

If the user wants to take the contributor path, they will be able upload their own photo from their profile or from the news feed. The contributor will be able to add the details to their photo as well as the price that they will charge before submitting their photo for other explorers to see. The user will be able to view their own portfolio of pictures they have uploaded already with the number of likes and purchases, as well as the comments present on their photos.

The frontend will be written predominantly in the React JavaScript framework (*React 2020*) with extensive usage of CSS in order to fix the styling and formatting where the Material-UI (*Material-UI 2020*) will be used as the React-based User Interface Toolkit where most of the designs and appearance of our website will inherently derive from.

The backend that will be used will predominantly be in Google Firebase (*Firebase 2020*) where posts, comments, photos and other relevant information will be obtained from and posted into the Firebase Cloud Database service where Firebase will support and store the information and media content that the user gets and posts in real time which will be connected with the React frontend from the Firebase backend and database. User authentication for the user registration and login will also be handled by the Firebase services which will be linked to the login and registration user interface handled in the frontend.

The original photo supplied by the user along with its associated information will be obtained from the user making inputs in the frontend. The watermarked version of the image will be generated by the front end invoking the watermark.js library (*watermark.js Library 2020*) and return it to the user which will then subsequently be stored in the Firebase backend database. This will be then obtained by a user get request and stored in their portfolio by a post request which the users will be able to do from the user interface in the frontend.



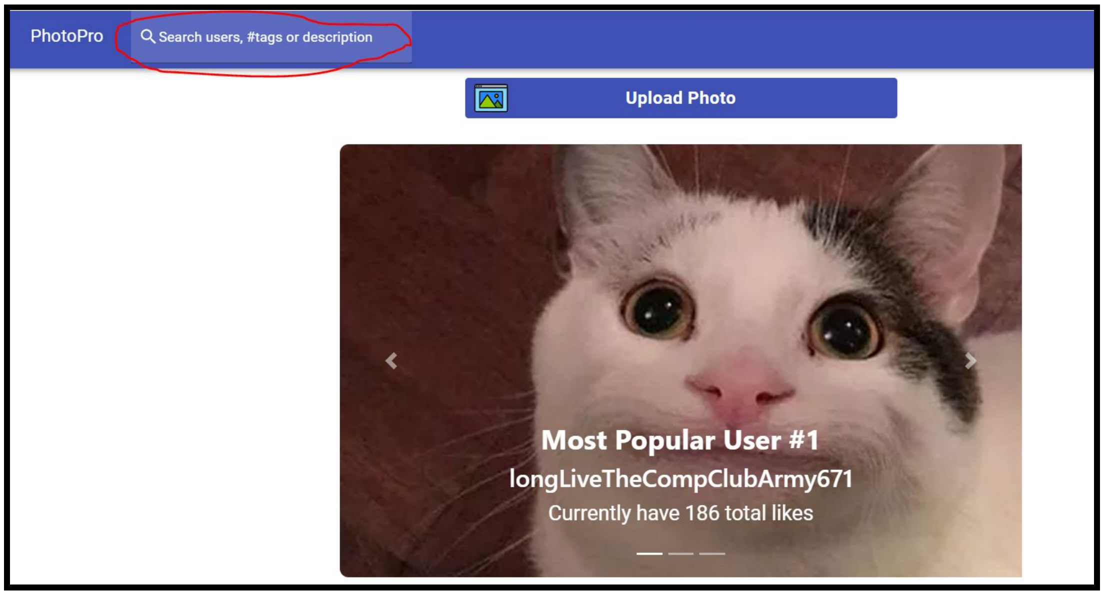
Functionalities and Implementation Challenges

Functionalities

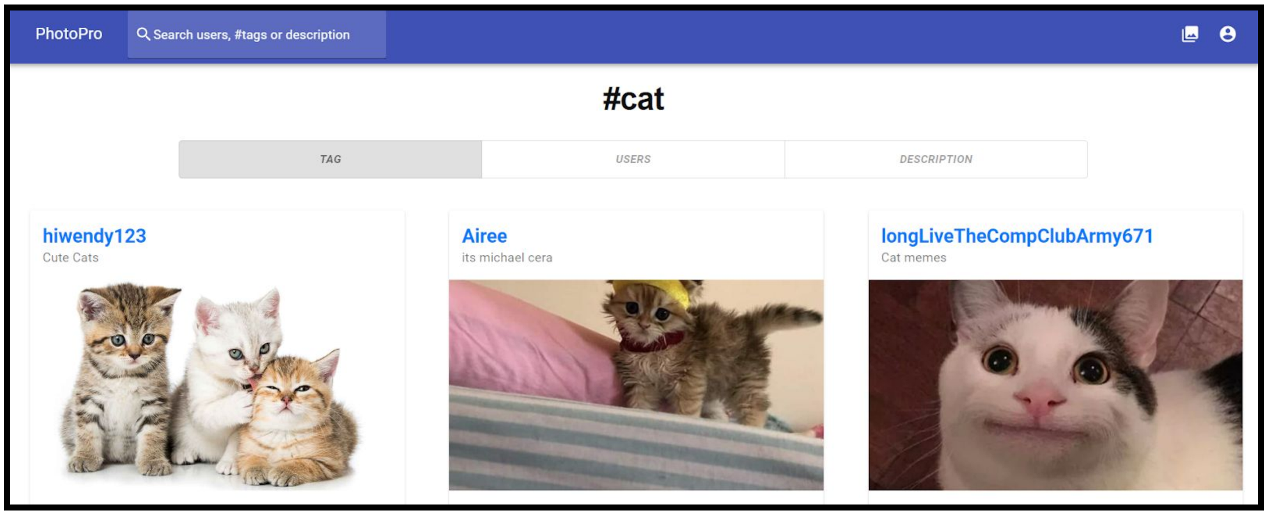
This section contains a description of the system's functionalities inline with the project objectives.

Project Objective 1	Have an account where users can find quality photos based on their interest.

Users can search for photos by typing keywords in the search bar.

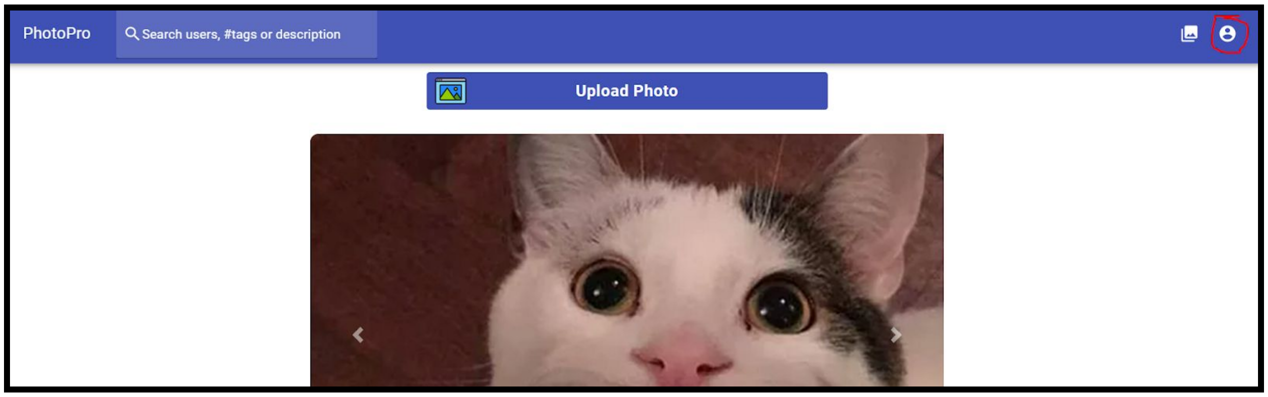

PhotoPro
Search users, #tags or description
Upload Photo
Most Popular User #1
longLiveTheCompClubArmy671
Currently have 186 total likes

Matching photos will appear.

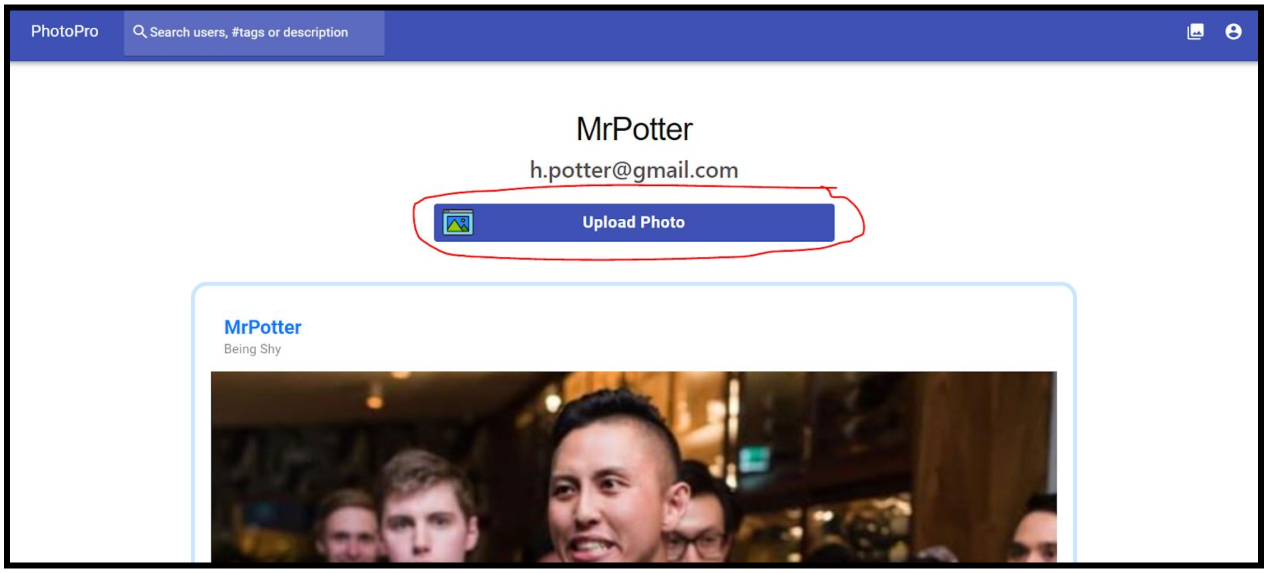


Project Objective 2	Create their own portfolio where users can upload photos to their own profile.

A user views their own profile by clicking the profile icon in the top right corner > “Profile”.

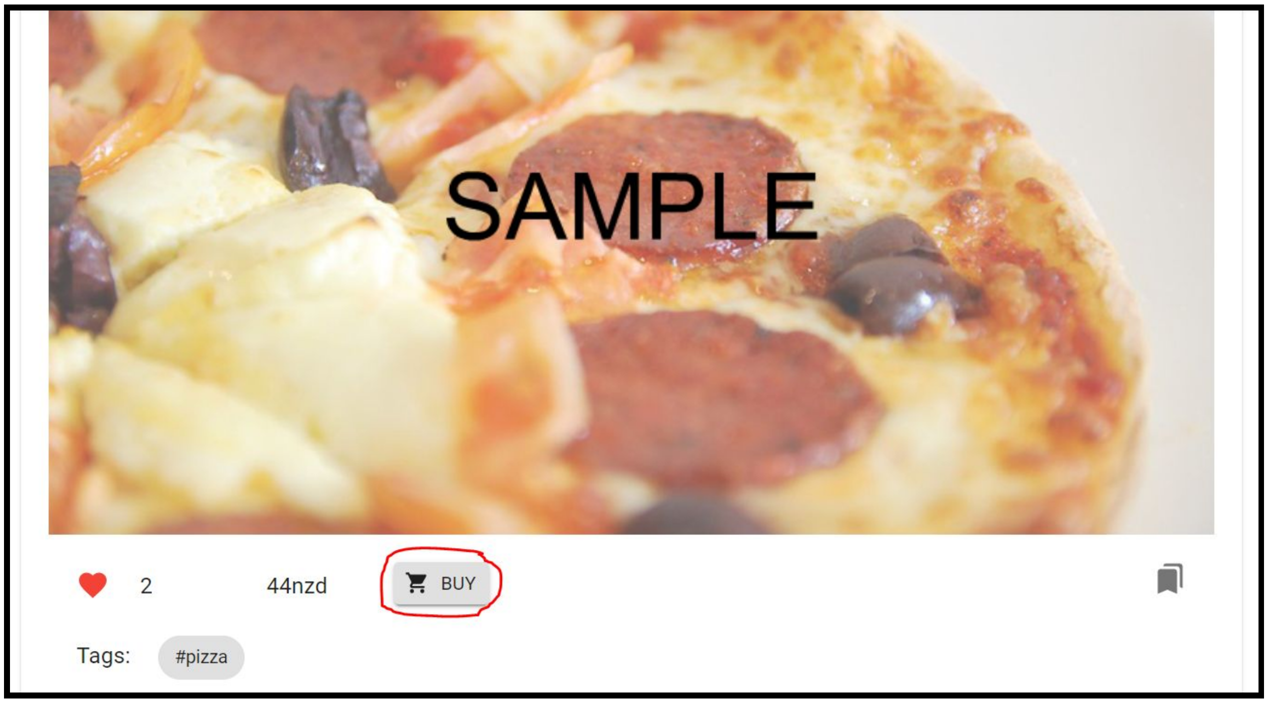


Then they are able to view their profile, where they can upload their own photos.

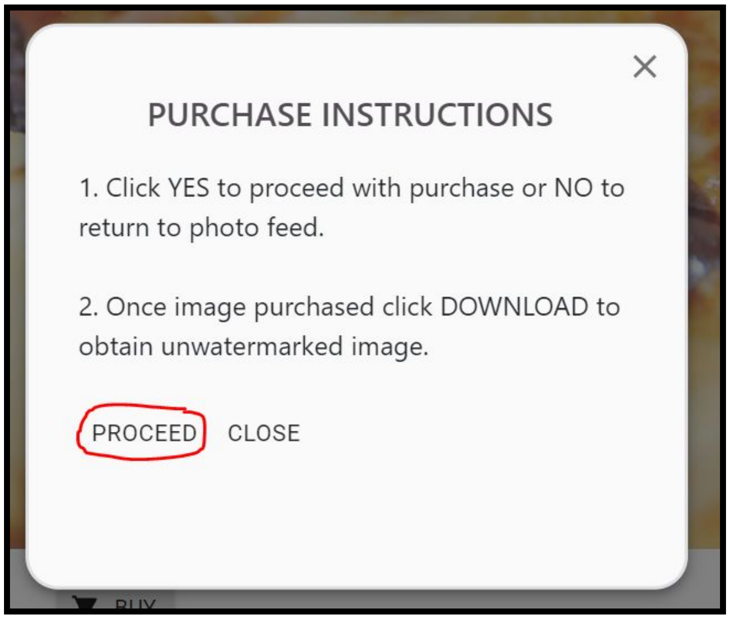


Project Objective 3	Be able to sell their photos for other users so that they can earn a living from it.

In order for a contributor's photo to be sold the user can click "BUY".



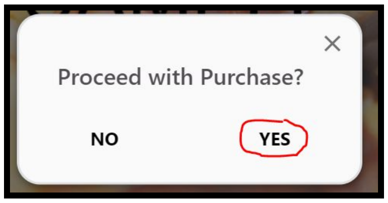
Then purchase instructions will be available and the user clicks "PROCEED".


PURCHASE INSTRUCTIONS

1. Click YES to proceed with purchase or NO to return to photo feed.

2. Once image purchased click DOWNLOAD to obtain unwatermarked image.

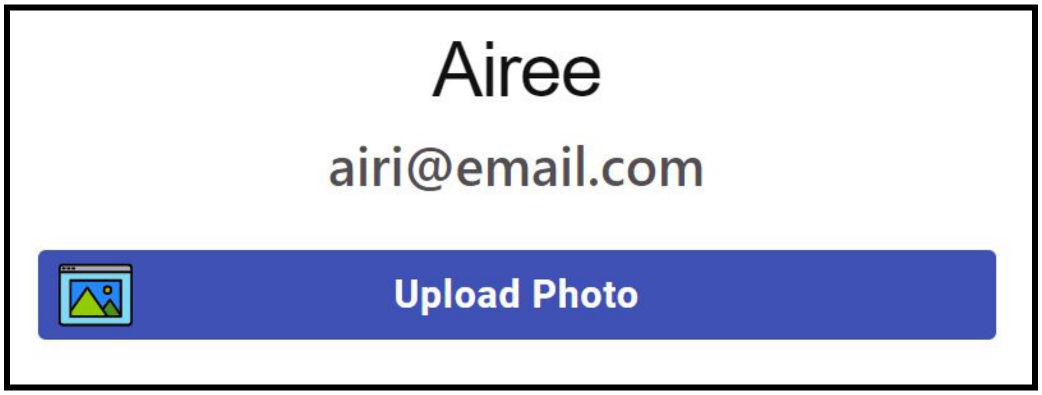
Upon clicking "YES", the photo is sold.



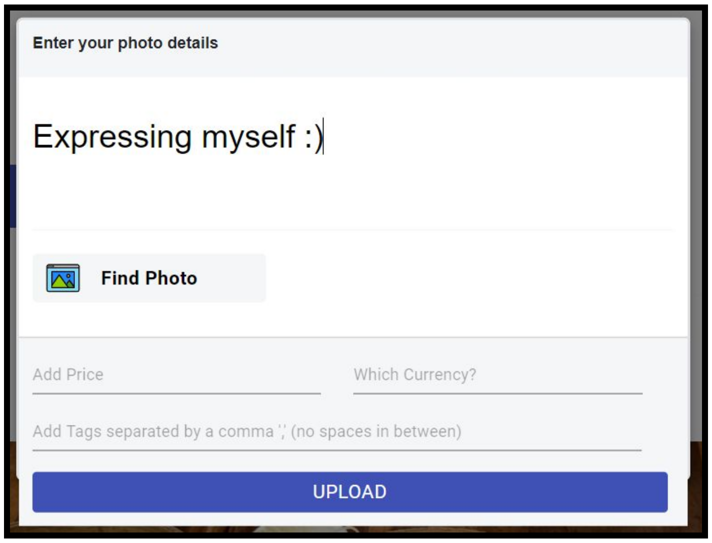
Project Objective 4	Express themselves by posting their own thoughts with their photos on an online feed.

Users can express themselves by adding comments. They can also express themselves by uploading photos with a caption included.

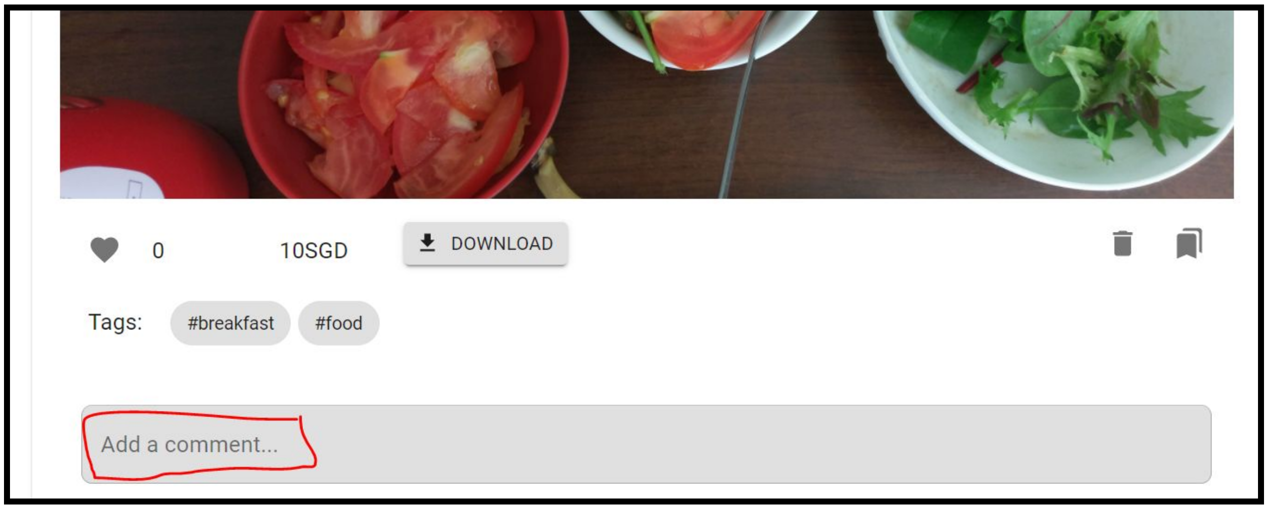
To upload a photo the user must navigate to their profile page and click "Upload Photo".



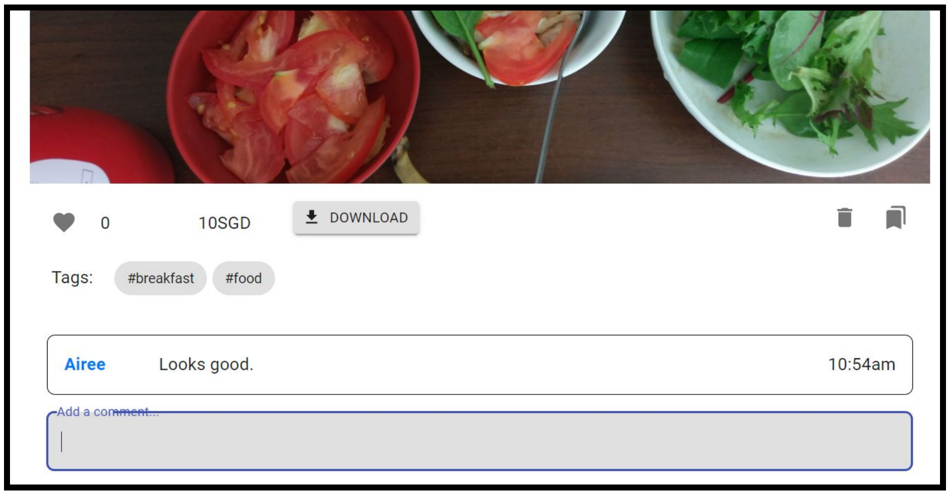
From here, they can add a caption to express themself / share their thoughts. Upon clicking "UPLOAD" it will be posted on their profile and online feeds.



Adding a comment.
To add a comment the user types in the “Add a comment..” section below a photo.



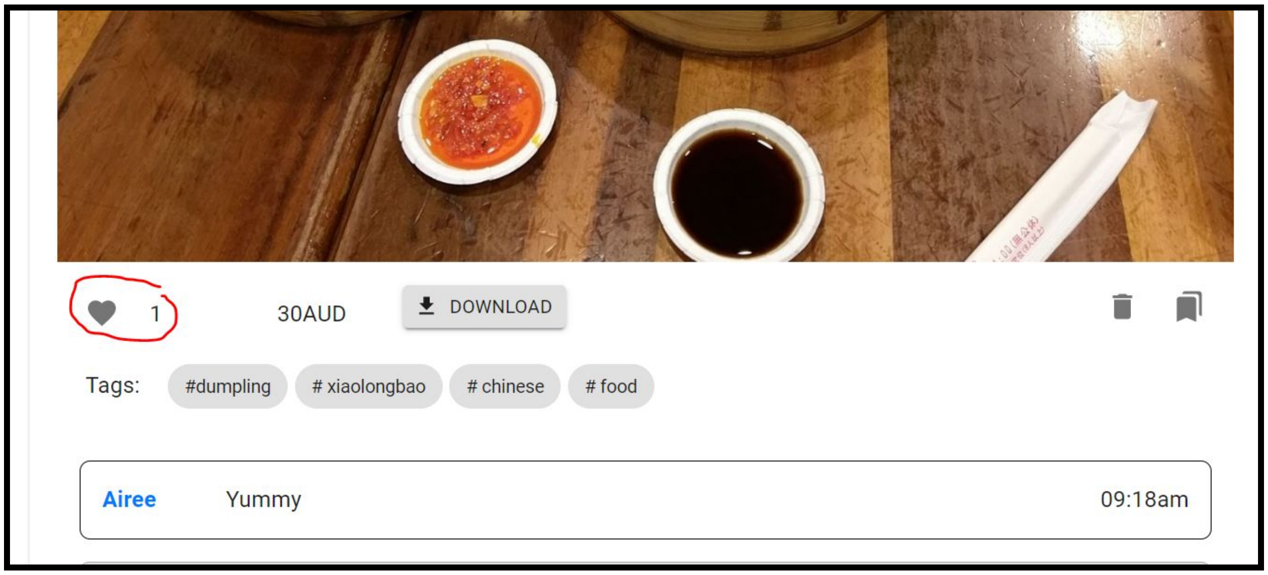
The comment will appear above.



Airee Looks good. 10:54am

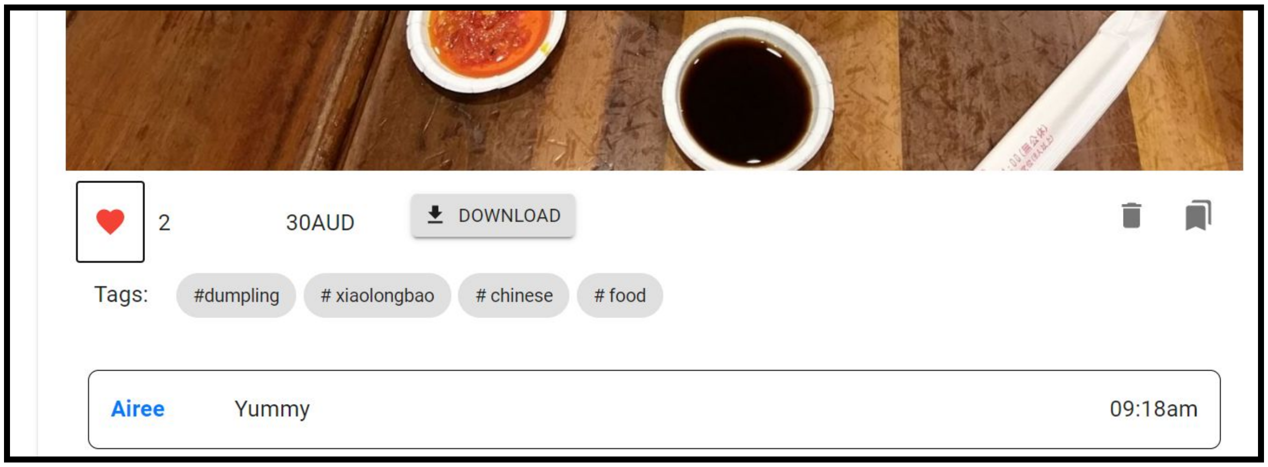
Project Objective 5	Give them the option to be able to like and add photos to the user's own collection.

A user can like a photo and view the number of likes.



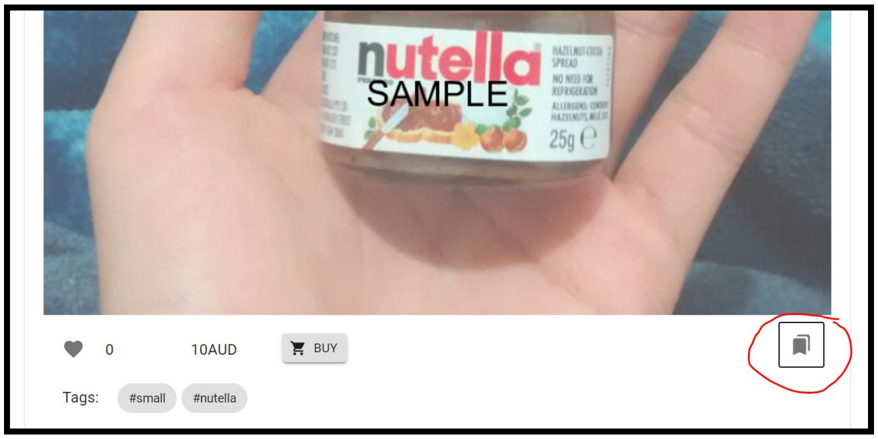
Airee Yummy 09:18am

After clicking on the heart.

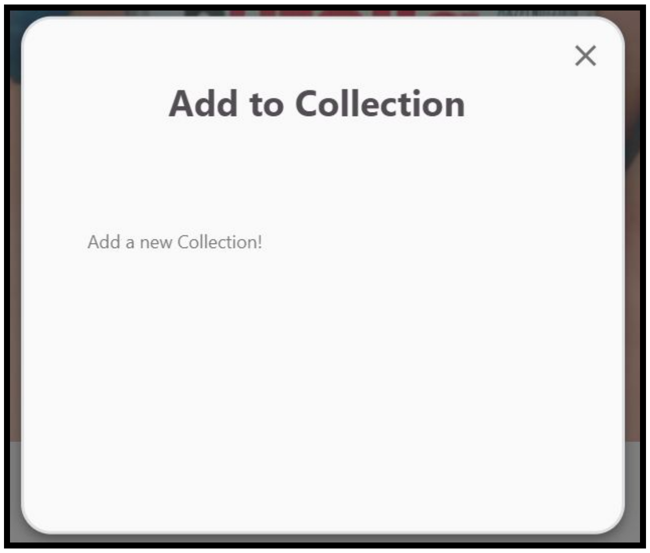


Airee Yummy 09:18am

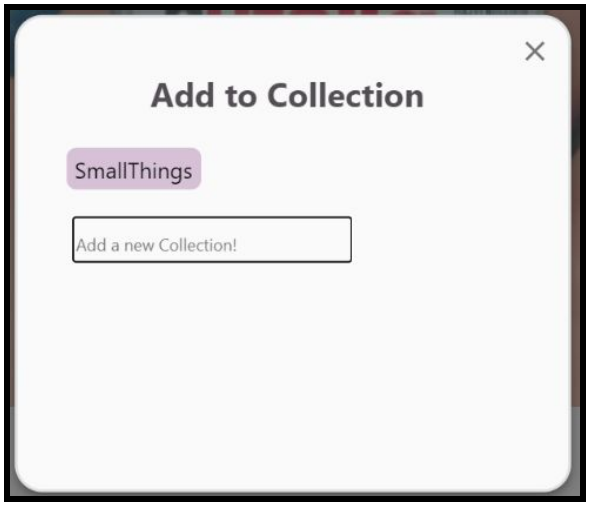
A user can also create collections of photos and add photos to a collection by clicking the bookmark icon.



A new collection can be created by typing in the collection name and pressing “Enter”.

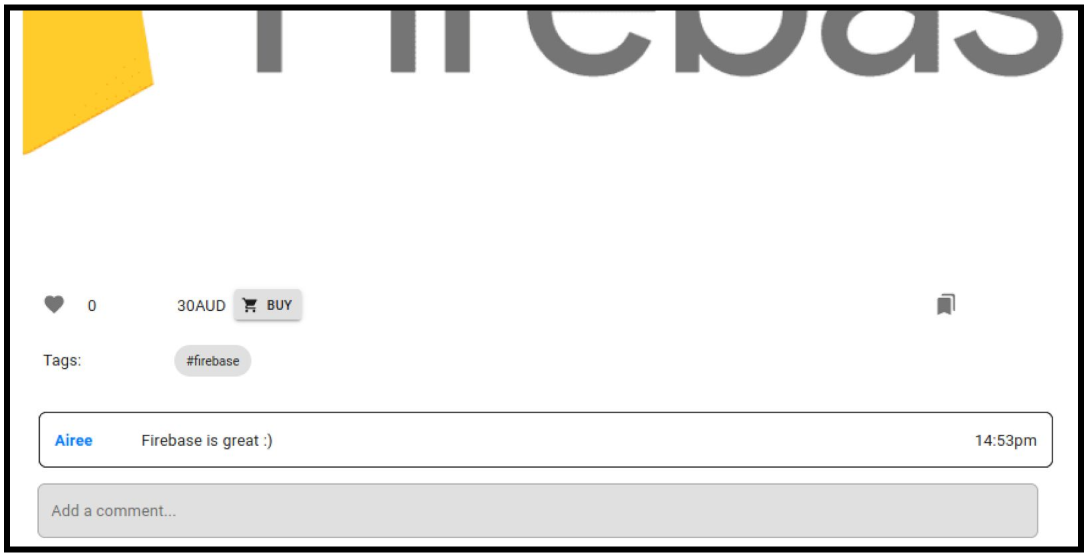


Once a collection is created, the photo can be added to the collection (e.g. SmallThings) by clicking on the collection name.



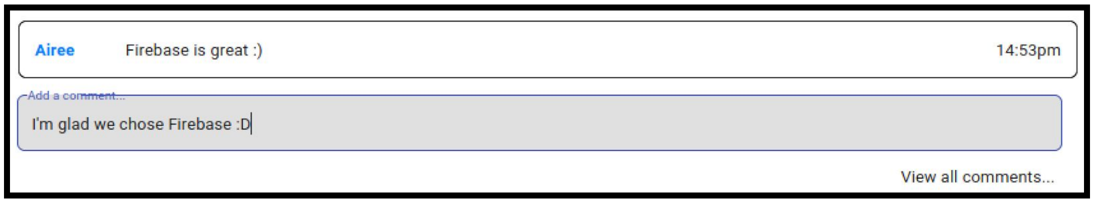
Project Objective 6	Be able to comment and express their opinion over other people's photos

A user can add a comment by typing in inside the 'Add a comment' text-box.



Airee Firebase is great :) 14:53pm

Add a comment...



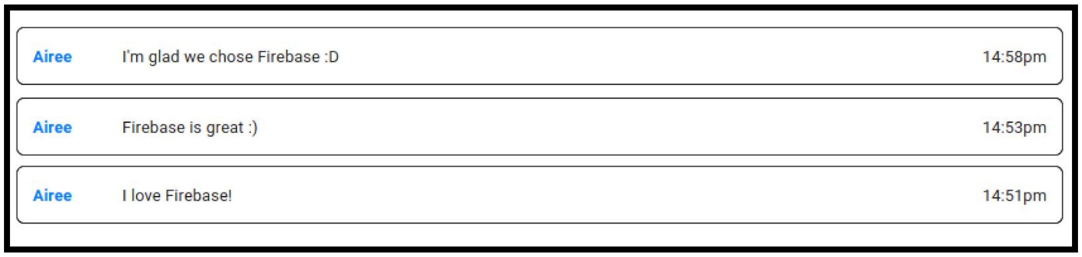
Airee Firebase is great :) 14:53pm

Add a comment...

I'm glad we chose Firebase :D

View all comments...

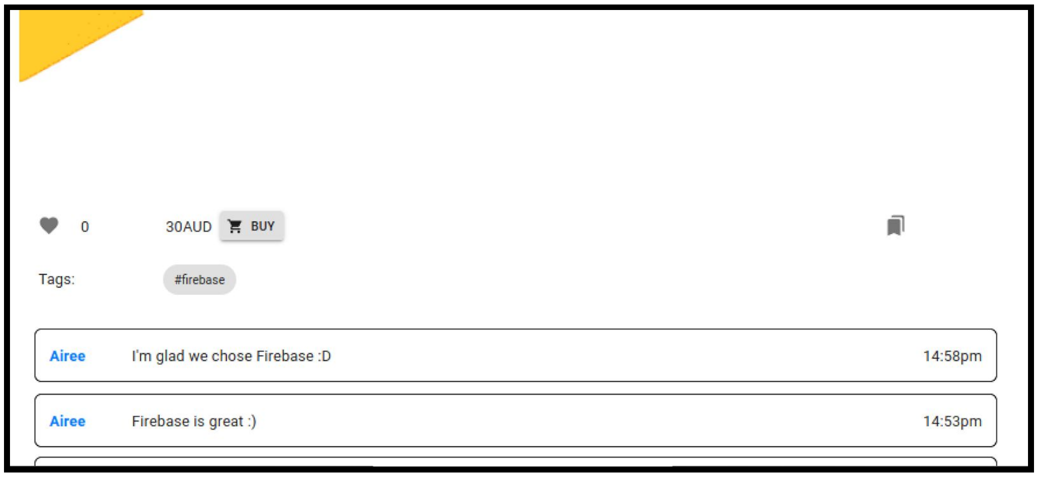
A user can access all comments by pressing "View all comments…"



Airee	I'm glad we chose Firebase :D	14:58pm
Airee	Firebase is great :)	14:53pm
Airee	I love Firebase!	14:51pm

Project Objective 7	Allow users to purchase, download and keep photos that they like from the application

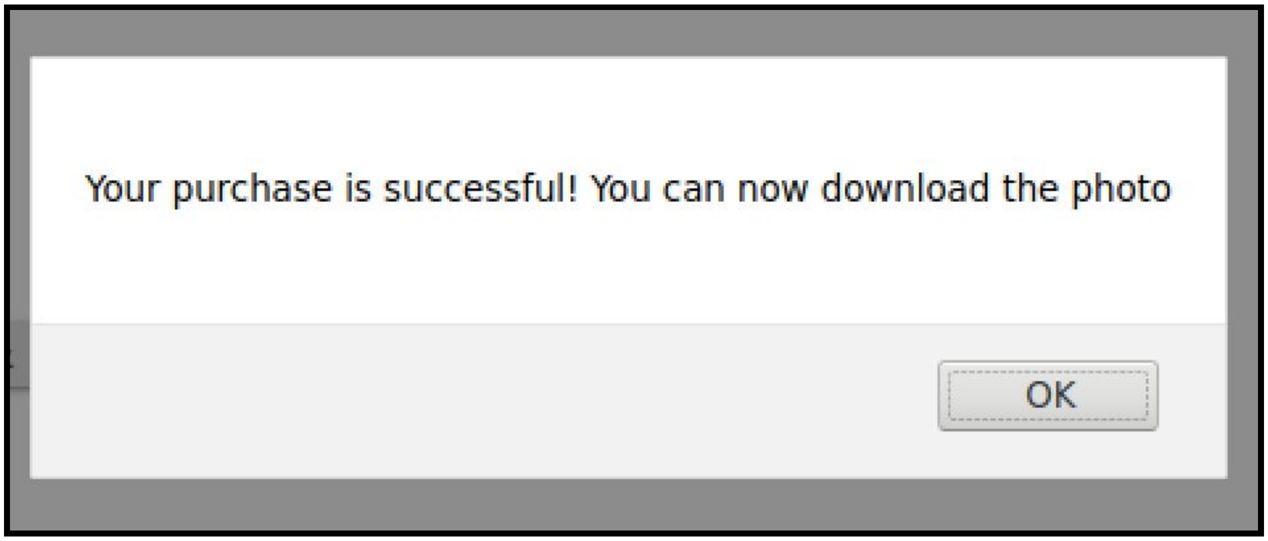
A user can purchase a photo by selecting the “BUY” icon.



Airee I'm glad we chose Firebase :D 14:58pm

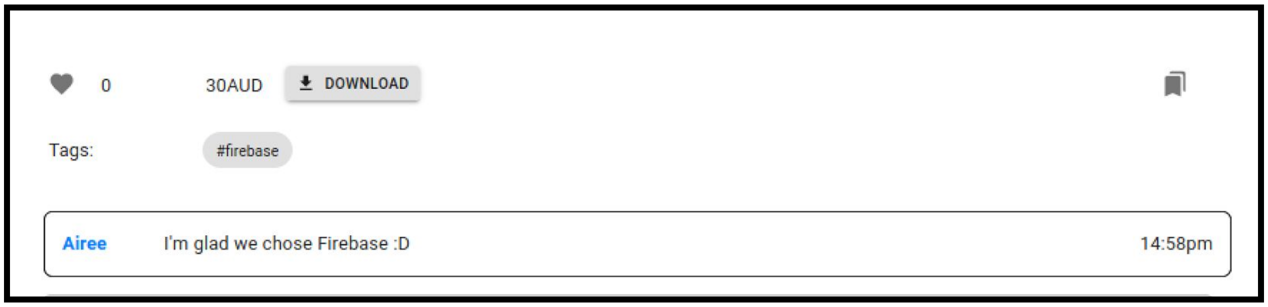
Airee Firebase is great :) 14:53pm

A confirmation message is displayed upon a successful purchase.



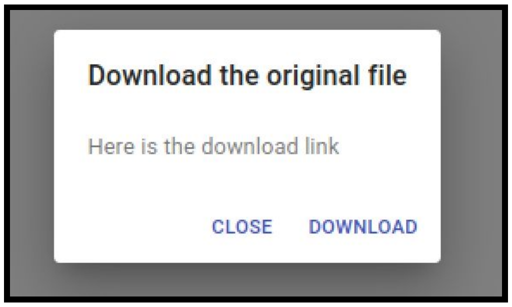
Your purchase is successful! You can now download the photo

“BUY” icon is replaced by “Download” after purchase.



Airee I'm glad we chose Firebase :D 14:58pm

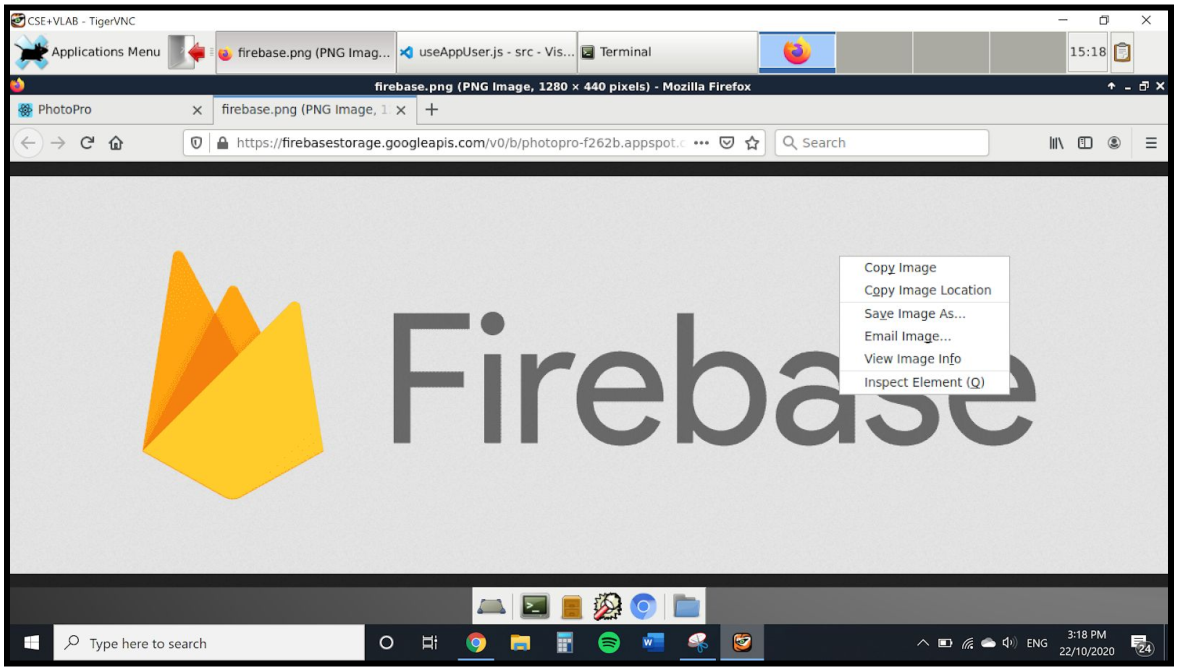
Download link modal displays upon “Download” selection.


Download the original file

Here is the download link

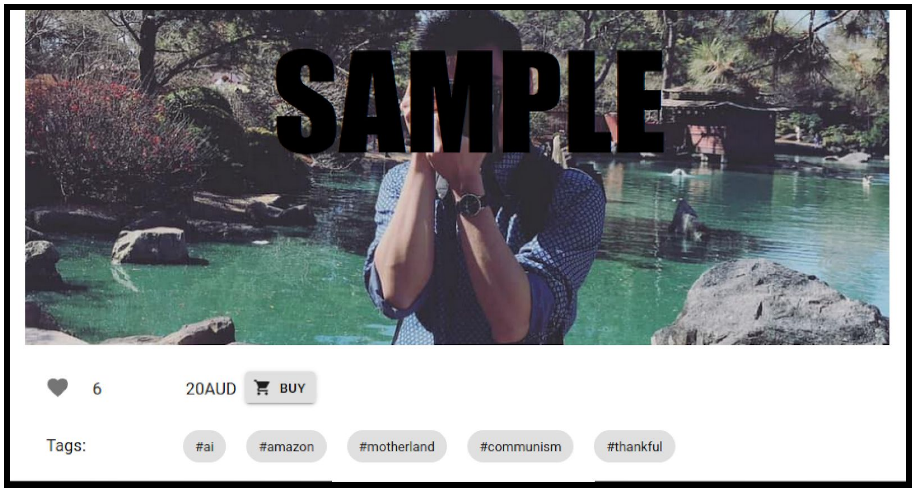
CLOSE DOWNLOAD

User is redirected to the image, where the image can then be stored by clicking “Save image As…”

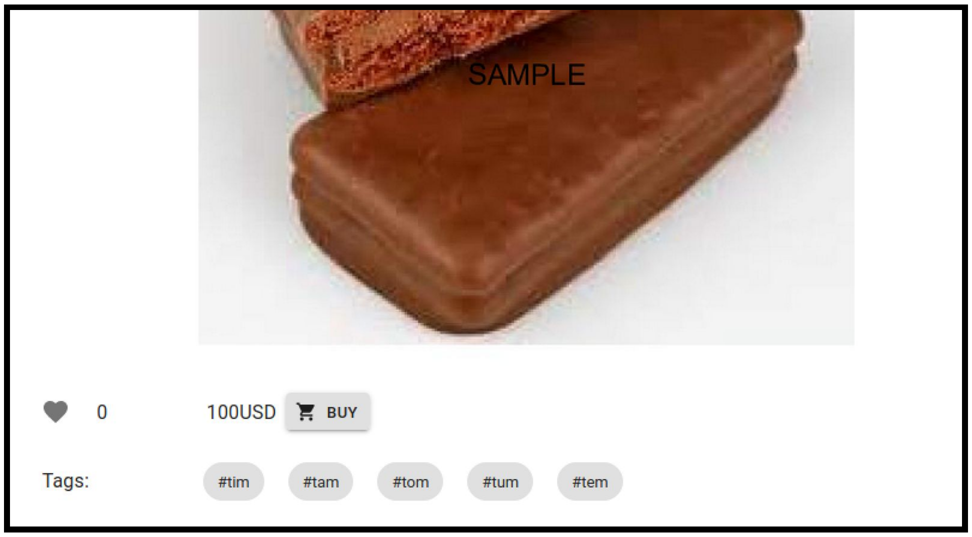

firebase.png (PNG Image, 1280 × 440 pixels) - Mozilla Firefox
https://firebasestorage.googleapis.com/v0/b/photopro-f262b.appspot.c
Copy Image
Copy Image Location
Save Image As…
Email Image…
View Image Info
Inspect Element (Q)

Project Objective 8	Be able to see similar photos on their feed based on what the users previously liked.

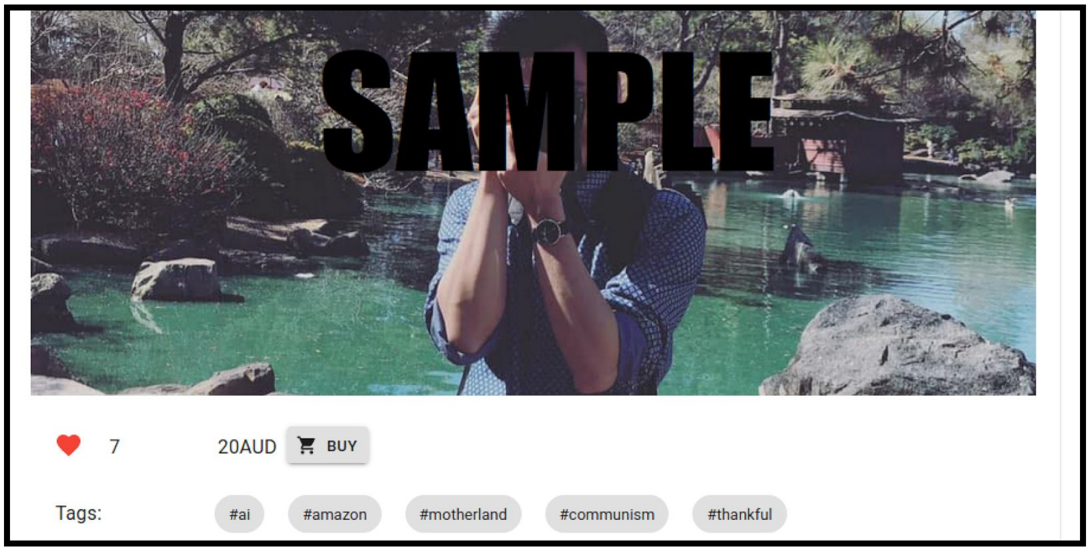
The two images below display the first two posts on a users' feed.

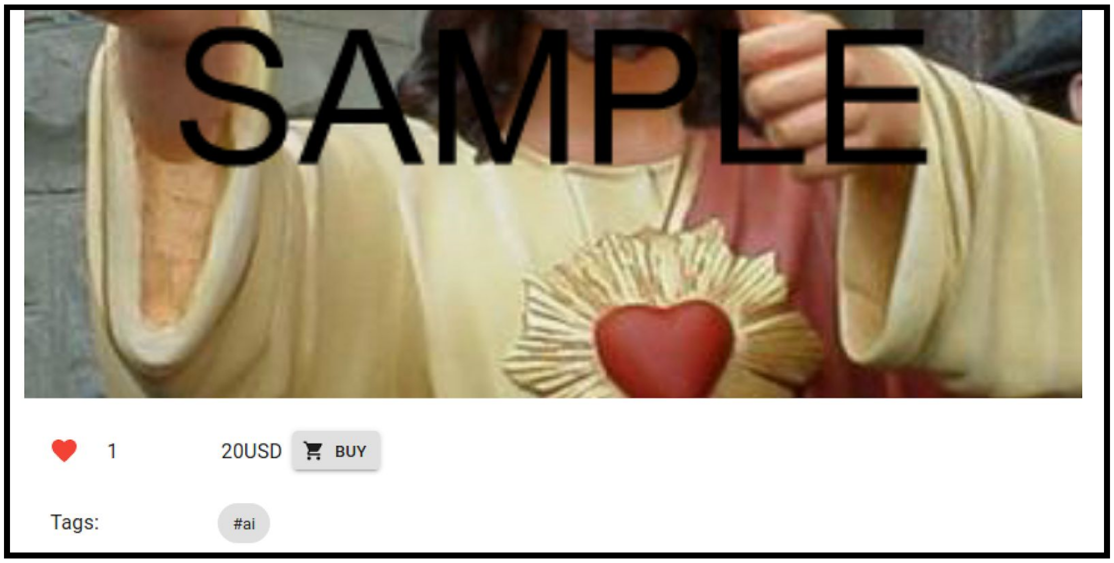


Take note of the absence of the above tags in the post below.



Similar photos are rendered based on the images which a user likes. Other posts which contain the same tag(s) as of the tags of a liked post are rendered at the top of the page. This change is demonstrated in the next set of images below.
After an image is liked as shown below, the next image rendered changes, to a post with the tag "#ai".
This allows for users to view posts which they may find appealing.





Project Objective 9	Have a guarantee that whatever photos they buy is of quality and ready to use.

Explorers can access the contributor's Name, email, Post Description, Image Cost, and Tags. In the case something goes wrong they can contact the seller by emailing them.
They are also able to view a watermarked version (with the 'SAMPLE' text) before making the decision to purchase.

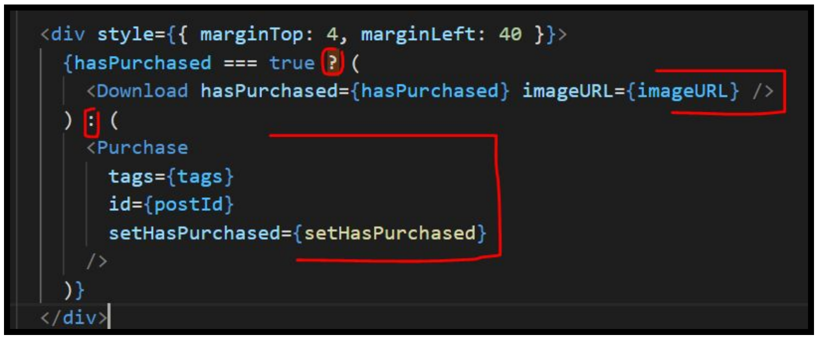
Project Objective 10	Display a preview of their photos where other users can buy an un-watermarked photo.

Each unbought post is a watermarked image, displaying a preview of the contributor’s photo. Upon purchase, the watermark (“SAMPLE” text) is removed.

Challenges

Challenge 1

Switching the DOWNLOAD and BUY button in user story 1.17 - Purchase Instructions. A post which has not been purchased should display a BUY button, however when purchased, it is to be replaced by a DOWNLOAD button. Implementing this relationship was met with many challenges as the team had originally had these buttons displaying at all times, regardless of the purchase status of the post. The challenge lay in interchanging one button for the other which was overcome by looking for a method that had the functionality of an “if-else statement” but worked with JSX. The team member working on this user story had very little experience in React JS which made it difficult. With a bit of guidance and online resources the team member was able to learn to use ternary statements to implement the desired functionality.
Implementation of the ternary statement can be seen in the screenshot below (in the file Post.js).



```
<div style={{ marginTop: 4, marginLeft: 40 }}>
  {hasPurchased === true ? (
    <Download hasPurchased={hasPurchased} imageURL={imageURL} />
  ) : (
    <Purchase
      tags={tags}
      id={postId}
      setHasPurchased={setHasPurchased}
    />
  )}
</div>
```

Challenge 2

Initially, the codebase was majorly decentralised, with UseEffect hooks, a React function, utilised multiple times, across several javascript files. UseEffect hooks are asynchronous in nature. Hence, features such as the bookmark collections did not render immediately, despite elements such as the bookmark icon having been selected on the application.
To resolve this issue, considerable refactoring was required. “useAppUser” was implemented in order to centralise information, allowing other files to obtain user data, collection data as well as bookmark data. UseEffect hooks, for obtaining collections and user information that were originally in post.js were transferred into useAppUser.js file. With this implementation, the drawbacks of asynchronous programming did not come into play, allowing bookmarks and collections to render immediately upon selection. Furthermore, needless duplication was removed, providing a cleaner, more stylish code.

Challenge 3

Searching was originally limited to direct exact queries against the firebase database. So when searching for posts, users would only get useful results if the exact case sensitive terms were used. Additionally, the search feature only allowed queries against a single collection, so it wasn’t clear how to specify a search for a tag, search for a profile or search for a description. Lastly, due to the asynchronous nature of the database querying, originally the searching was happening asynchronously but the web app was not waiting to render, meaning there were memory leaks and bugs in the search functionality.

The first issue was resolved by using the Autocomplete feature from Material-UI. This allowed us to do a single query against the database and get autocomplete results that are user friendly. To allow for searching using either tags, profiles or description we used “#” to delineate tag searches and then we used different firebase queries for tags, users and descriptions that change on user input. To fix the asynchronous issue, we used the async functionality for firebase queries and then added booleans in UseEffect to wait to display data before rendering.

Challenge 4

Implementing the tags system in order to prioritise posts with certain tags the users were following was challenging to solve which was determined by if they liked, bookmarked, searched and bought the photos with these respective tags. At first whenever a user adds a tags to the Tags Followed collection using the aforementioned methods above, it filters out and rerenders the feed of photo posts with only photos that the user follows in the feed but the issue with that is if the users initially have no tags that they followed and they follow only a single tag, they will only see the single tag effectively preventing the users from seeing the rest of the posts in the feed that was stored in the database.

So the temporary solution for our team members to be able to view all the posts is to delete the tags followed collection so that we can all see all the posts and repeat the same process every a tag was followed whilst experimenting with the other features of the code such as the like, bookmark, buy and search feature which became a tedious process. The next step that was done to improve this process more efficiently was to map all the posts into two keys, one key that contains all of the posts that was followed, and the other key that contains only the posts that were followed. Afterwards, a query was performed that subtracts the tags that are followed from all of the posts to return a map with the key of all of the remaining posts that complements all of the posts that contains all the tags the user is following. We then stack all of the posts from the post map with the key of all the posts that consist of tags that the user follows to be displayed first, then display all of the remaining posts that consist of the tags that the user doesn't follow to address the previous challenge.

The other issue is that Firebase can only query up to 10 elements of the tags followed array so we have to order the tags followed by the timestamp of when the particular tag was followed starting from the most recent tag that was followed. To query all the posts with the tags followed, only the top 10 most recent tags followed was sliced from the extensive tags followed array from the collection sorted by the timestamps. Since we fixed these issues, after re-rendering the posts when the tags followed collection from the database has been updated, the feature to display all the posts in the feed that prioritise and displays the tags that the user followed has then been successfully achieved addressing our project requirements.

Challenge 5

Finding a suitable API to use to generate a watermarked image was challenging. Many of the free APIs found online were limited in some way, usually by the volume of requests that can be made in a particular time period.
One such API that was investigated was Neutrino API (*Image Watermark API - Neutrino API Docs*), which offered a free tier with up to 10 conversions per day, but was deemed to be not enough to allow for demonstration and testing on the same day. Another free API that was considered was Filestack (*Filestack Transform*), which offered 1000 monthly transformations, but only 100 monthly file uploads. This API required that a file be uploaded to their platform prior to being converted, so it effectively had a limit of 100 image transformations per month, which was also deemed too few for our purposes.
Finally, Moesif text over image (*Text Over Image API*) was investigated, and deemed suitable for our purpose as it had unlimited requests, but was significantly slower than the other APIs. Since we are expecting that new photos are to be uploaded to the website fairly infrequently, this API was chosen for use in the project.

User Document/Manual

How to build, set up and configure the system

Part 1 - Cloning and npm

Step 1
Clone the repo:
https://github.com/unsw-cse-capstone-project/capstone-project-comp3900-f11a-tofu.

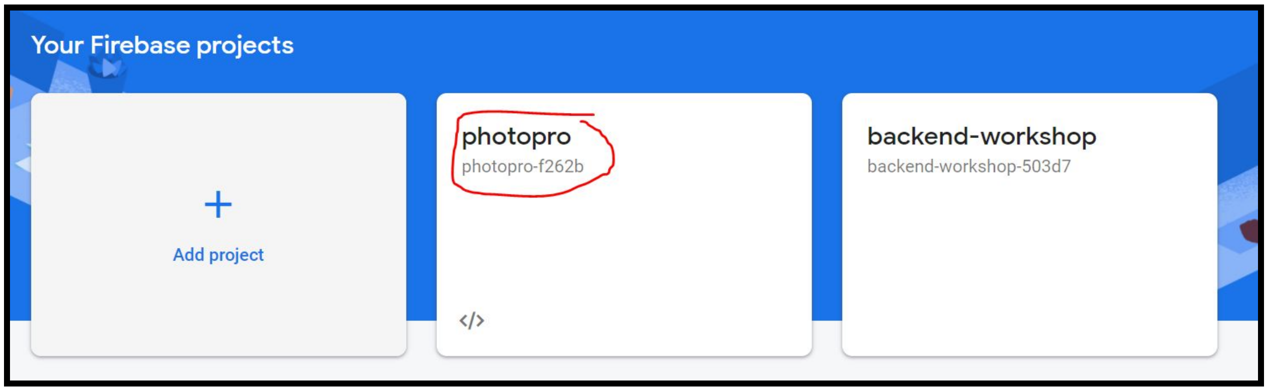
Step 2
Open a terminal and change directory to the repo and run the command "**npm install**". This will take some time (possibly a few minutes)

Step 3
In the terminal, do "**npm start"**. This may take a moment to compile. Once ready, you will be able to access the website via the url http://localhost:3000/ (this will be printed in the console, and may differ depending on the machine e.g. http://localhost:3001/)

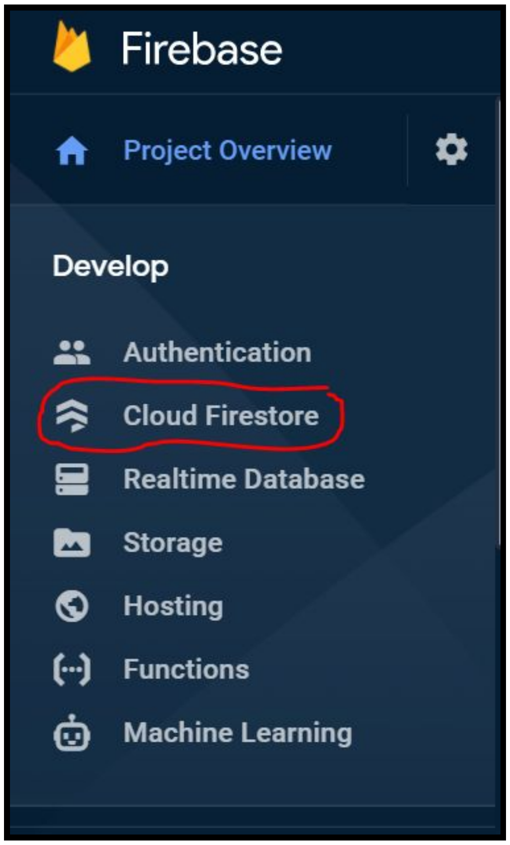
Part 2 - How to use Firebase

In order to use firebase and firestore (the backend and database of this project), you must be added as an administrator of the firebase project. You can gain access by asking the current project owner to add your gmail to the project.

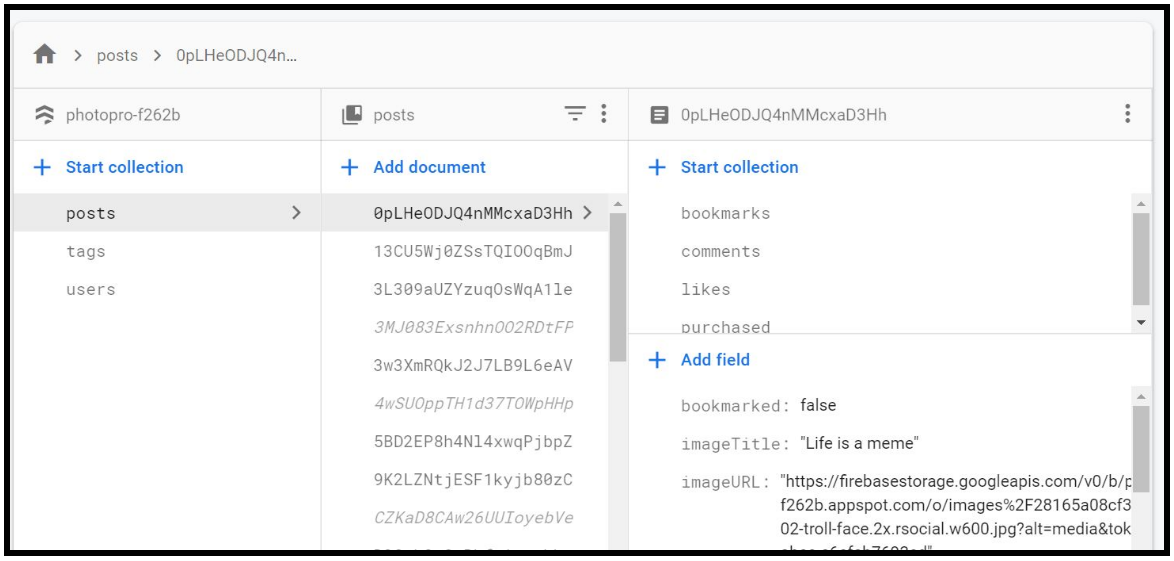
Step 1
Once you are added to the project go to firebase.google.com and click "Go to console" on the top right hand corner. From here click "photopro".



Step 2
Then click "Cloud Firestore" on the left hand side. This is how you access the project's database.



Step 3
You should see a screen like the one below. From here you can *add, update, and delete* all types of data stored for the project. This includes photo tags, users of the platform, the number of likes a post has, a post of an image and the corresponding information. Any changes to the front end will be reflected in the database, and vice versa.



posts
tags
users

0pLHeODJQ4nMMcxaD3Hh
13CU5Wj0ZSsTQIOOqBmJ
3L309aUZYzuqOsWqA1le
3MJ083Exsnhn0O2RDtFP
3w3XmRQkJ2J7LB9L6eAV
4wSUOppTH1d37TOWpHHp
5BD2EP8h4N14xwqPjbpZ
9K2LZNtjESF1kyjb80zC
CZKaD8CAw26UUIoyebVe

bookmarks
comments
likes
purchased

bookmarked: false
imageTitle: "Life is a meme"
imageURL: "https://firebasestorage.googleapis.com/v0/b/p
f262b.appspot.com/o/images%2F28165a08cf3
02-troll-face.2x.rsocial.w600.jpg?alt=media&tok

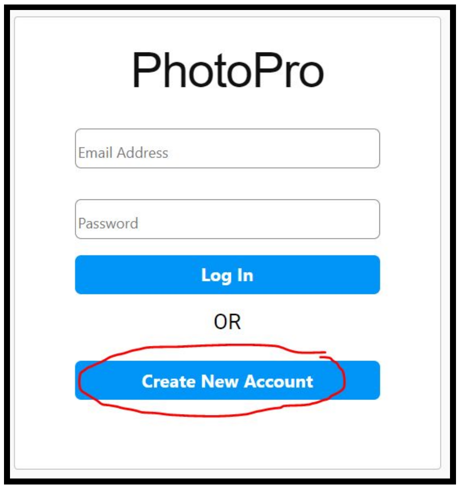
<u>Note</u>: do not like and unlike a post too fast as firebase requires time to rerender, and excessive clicking will confuse the database which may skew the number of total likes.

How to use the system and functionalities

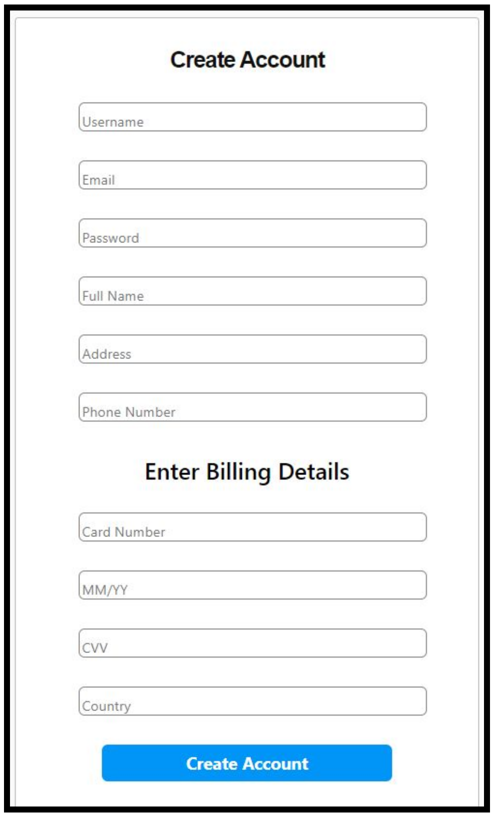
Below you will find a complete list of all the functionalities available as a user of PhotoPro and how to use them.

1. **Sign up/register, create a profile/account:**

   To create an account click the “Create New Account” button on the login page.

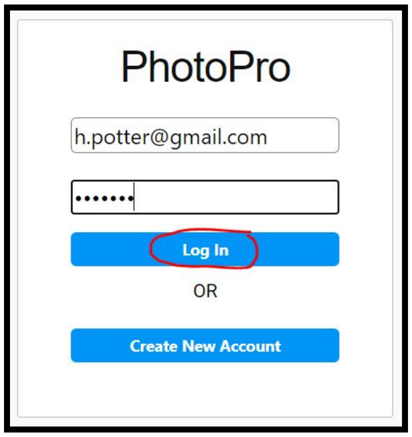


   You will then be prompted to fill in your personal details and save your payment details. Fill in all the fields and click “Create Account”.

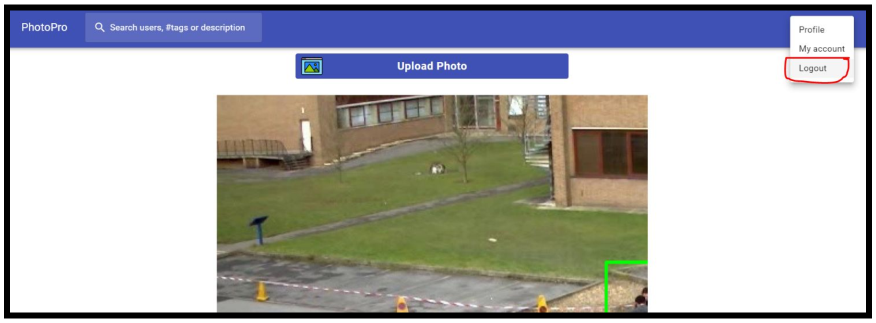


Your account should now be successfully created.

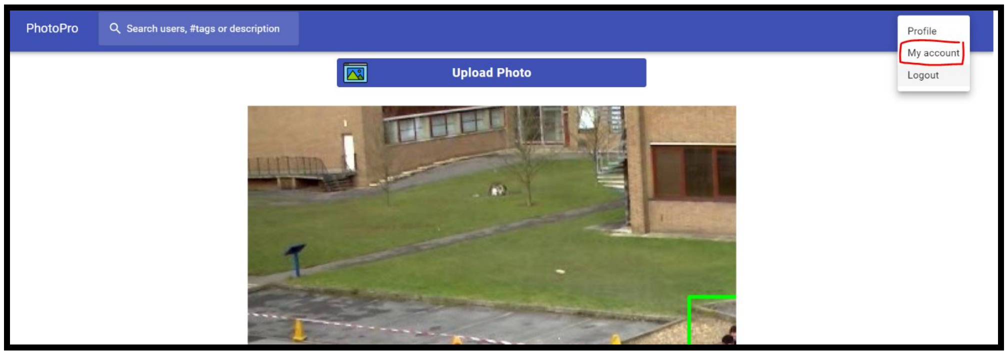
2. **Login:**
   To login type a valid email and password and click “Log In”.



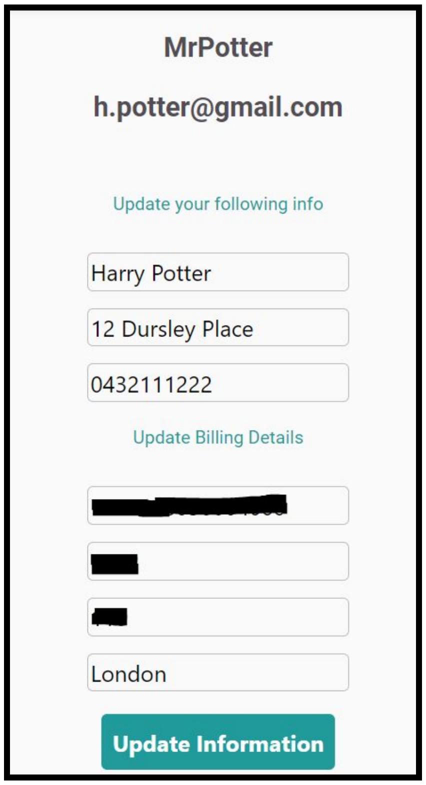
3. **Logout:**
   To logout, click the profile icon in the top right hand corner, and then click “Logout”.



4. **Update personal information:**
   To update personal information click “My account” from the dropdown on the top right hand corner.



You will then be redirected to the account page, where you can update your information.


MrPotter

h.potter@gmail.com

Update your following info

Harry Potter

12 Dursley Place

0432111222

Update Billing Details

[redacted]

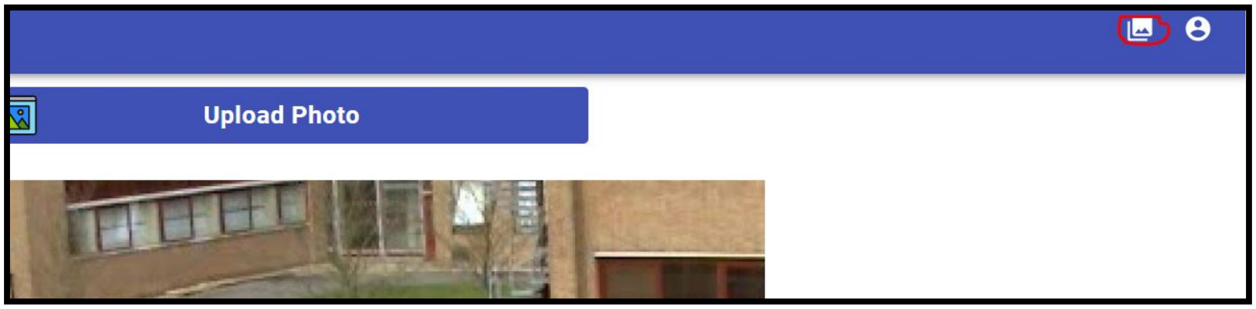
[redacted]

[redacted]

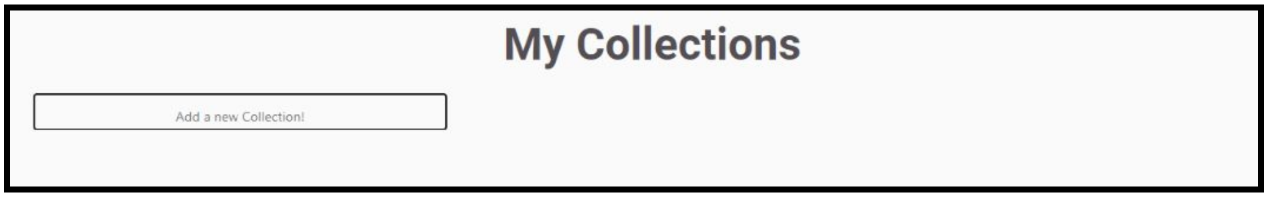
London

Update Information

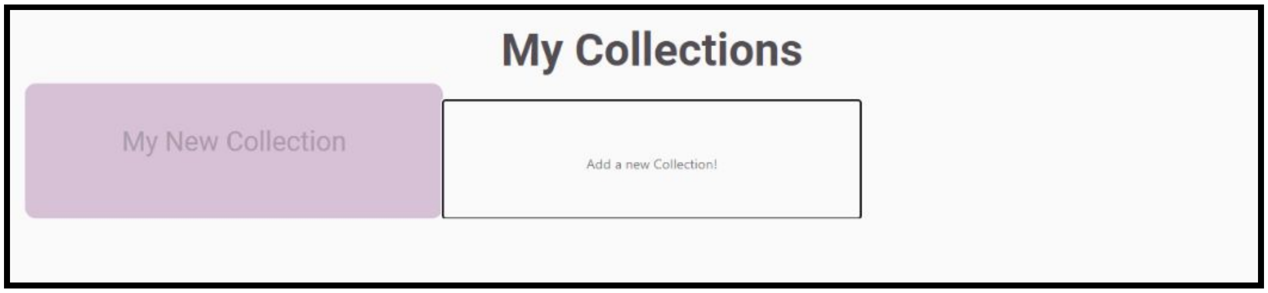
5. **Creating named collections of photos:**
   There are two ways to create named collections.
   The first method is by clicking the collections icon in the top right.



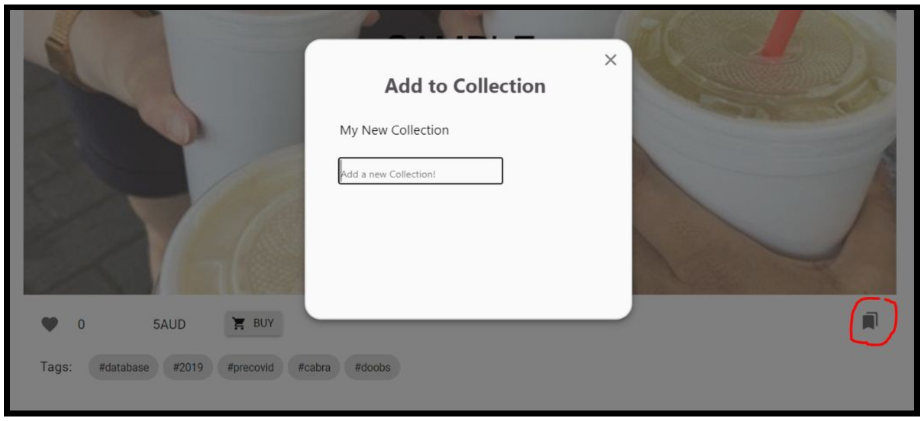
From here you can add a collection by typing the name of the new collection in the textbox and pressing the “Enter” key.


My Collections

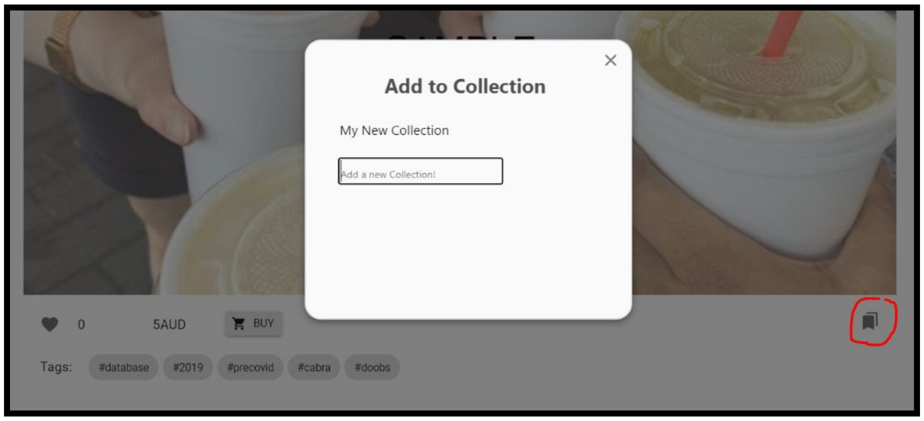
Add a new Collection!



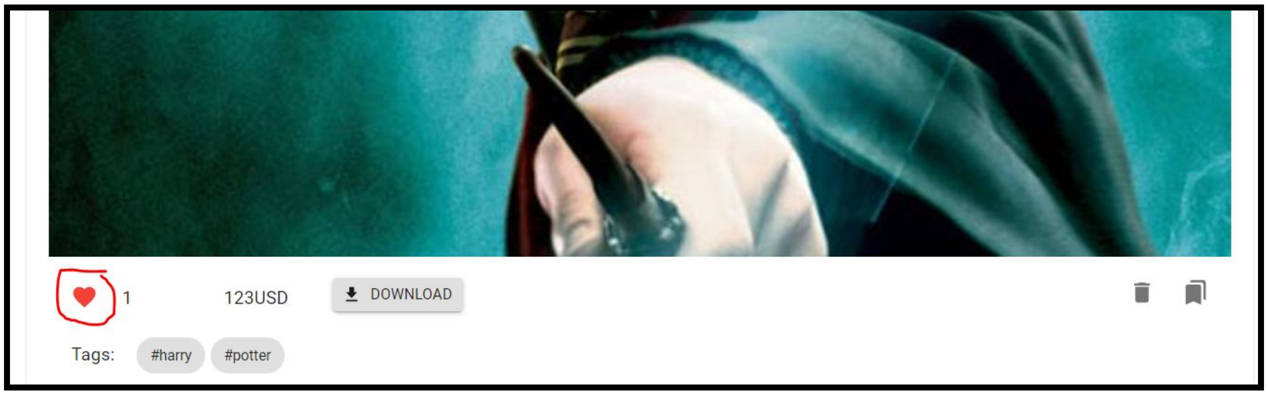
The second way to create a new collection is through the bookmark icon at the bottom right of each image on the photo feed. Once the bookmark icon is clicked, a pop-up will appear. By typing the name of the new collection followed by the "Enter" key, a new collection can be created.



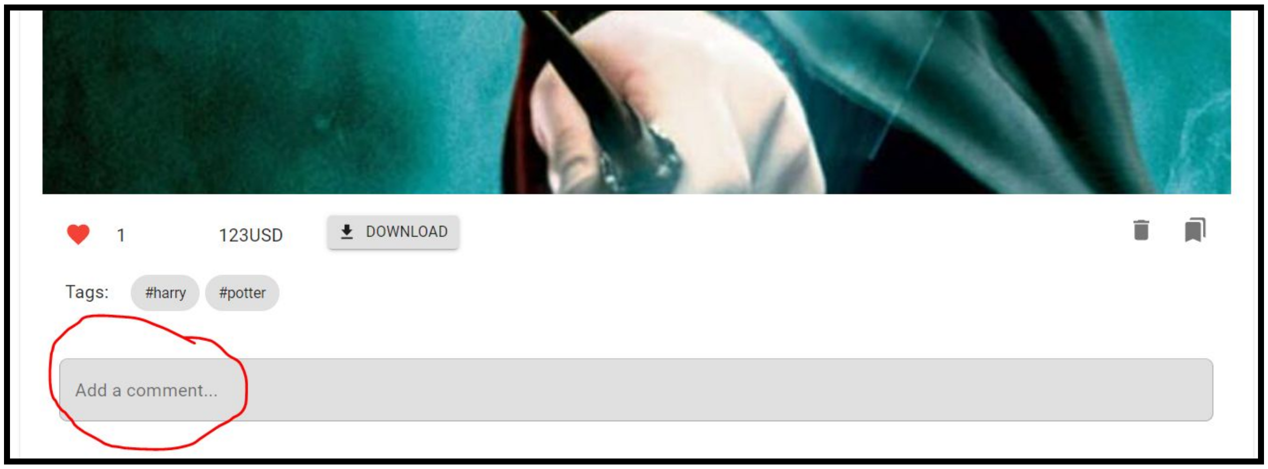
6. **Adding a photo to a collection through the bookmarking functionality:**
   Click the bookmark icon and then click the name of whichever collection you want to add it to. E.g. "My New Collection".



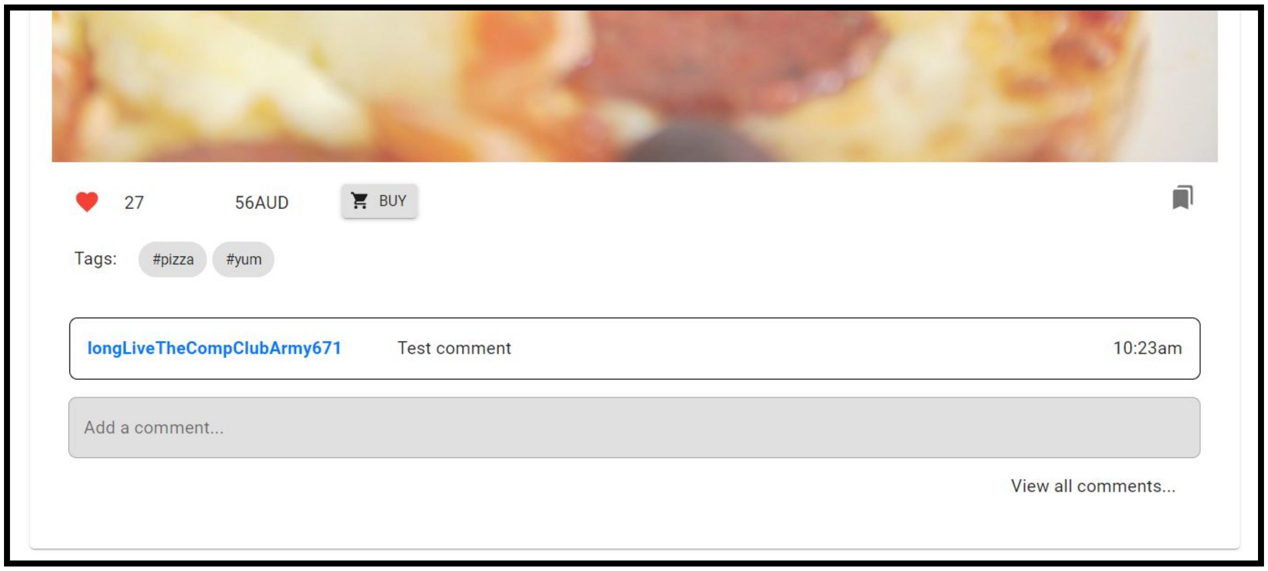
7. **Liking a photo and seeing the number of likes a photo has:**
   To like a photo click the heart icon at the bottom left of the photo. Next the heart there is a number which indicates the number of likes it has.



8. **Commenting on a photo:**
To comment on a photo type your comment in the “Add a comment…” section and press “Enter”.

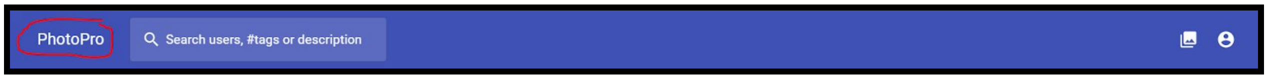


9. **Viewing all comments on a photo:**
Click “View all comments…”. Note that photos with 1 or less comments will not have this option.



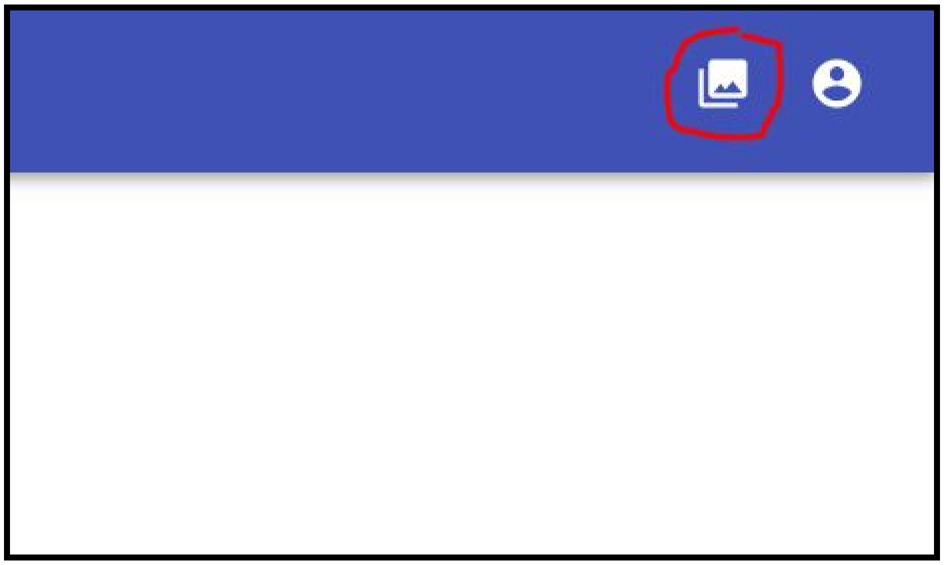
10. Accessing a photo feed (a feed of photos on the site):

Once you are logged in you will automatically be shown a photo feed. However, if you require to navigate to it, click the “PhotoPro” text in the top left hand corner.

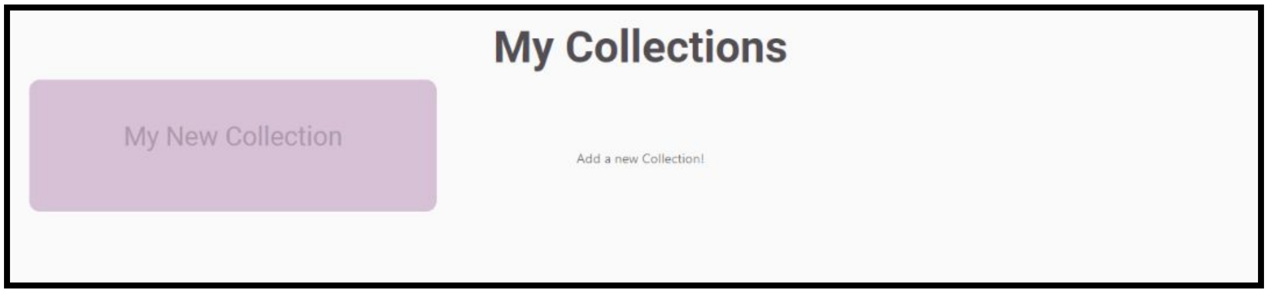


11. Viewing collections of photos you have bookmarked:

Click the collections icon in the top right-hand corner.



Then click on the collection you wish to view.

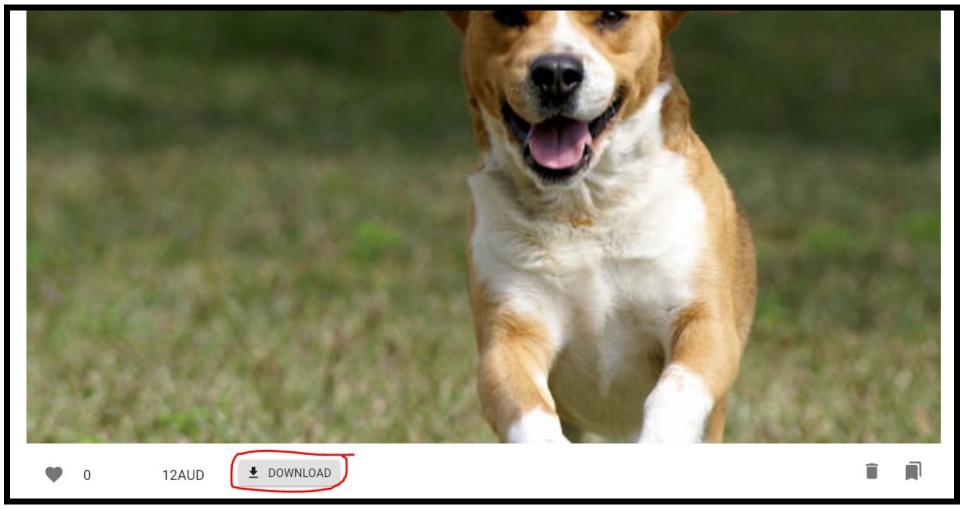


12. Viewing watermarked versions of photos as a preview before purchasing:

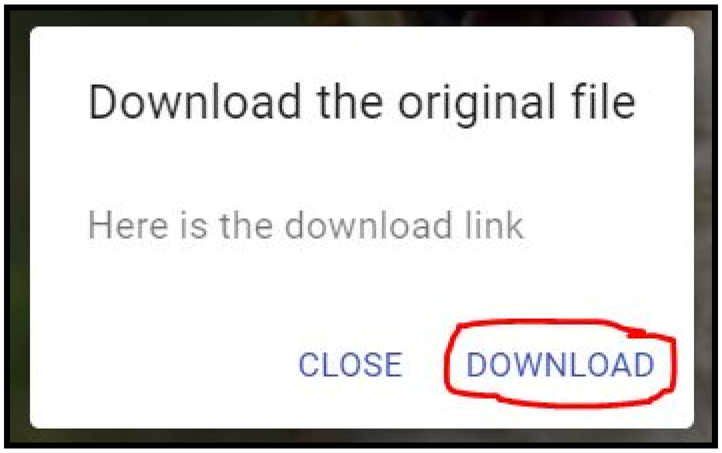
Whenever you view an unpurchased photo, it will be watermarked with the text “SAMPLE” as seen below.

13. Viewing unwatermarked versions of photos once purchased:
Once a photo is purchased, it will no longer be watermarked, as seen below.

14. Downloading an image once you have purchased it:
Click the “DOWNLOAD” button to download.

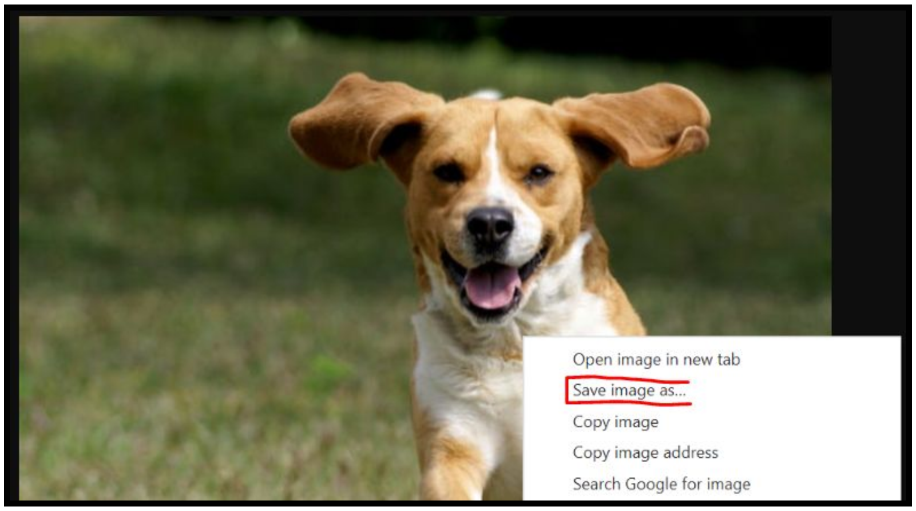


Then click “DOWNLOAD” again.


Download the original file

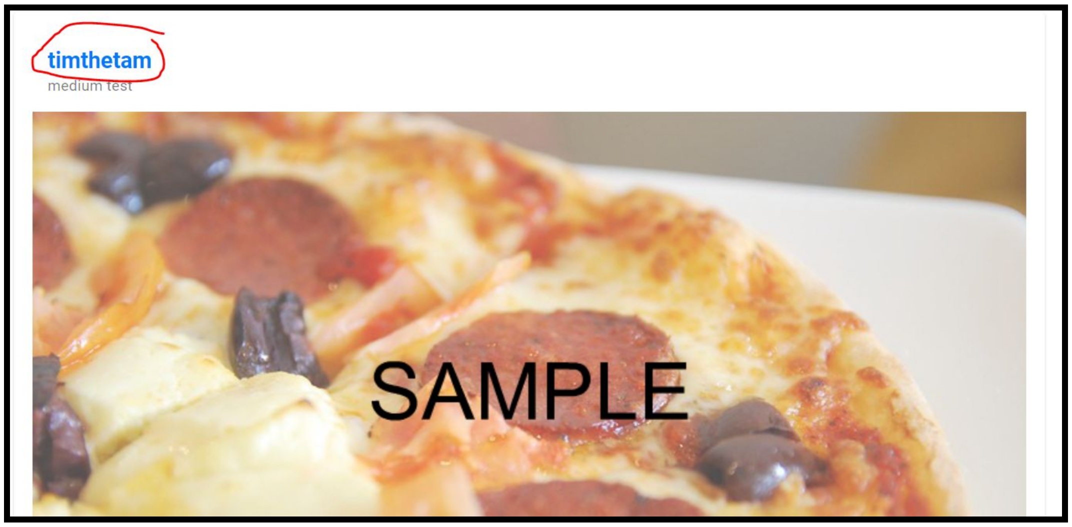
Here is the download link

You will be prompted to another page, where you can right click and save the image.

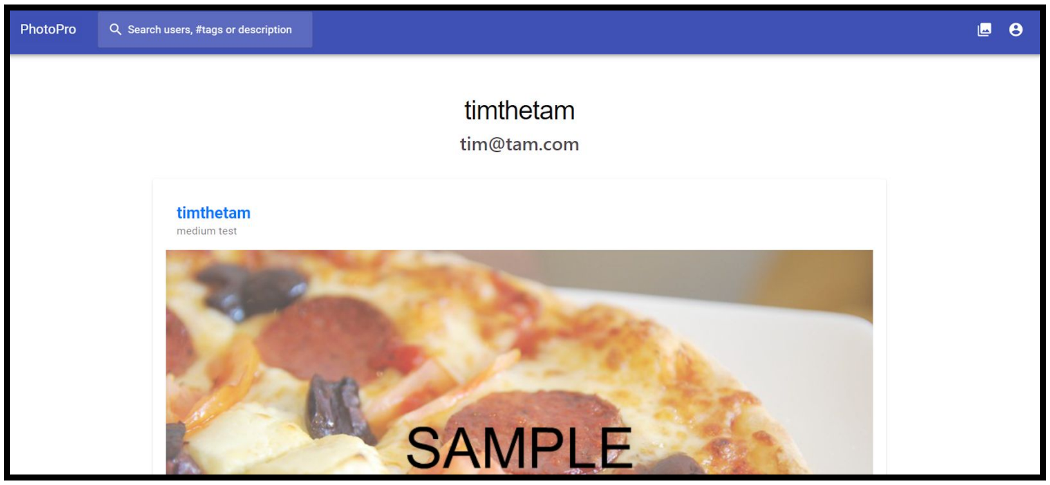


15. View contributor’s contact details and all public photos by that contributor:

By clicking on a contributor's nickname you should be able to see their profile which includes their nickname, email, and photos they have uploaded. To view a contributor's details click on their nickname which is located above a posted image. E.g. "timthetam".

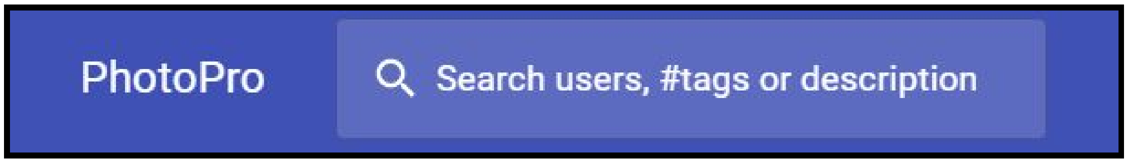


From here you'll be able to see their nickname, email and photos (as you scroll down).



16. Search for photography:

Use the search bar in the top left corner to search. For users and descriptions simply type the keyword, for tags you must include a "#" before the word. E.g. to search for tag "small", you must type "#small". Press "Enter" to search.

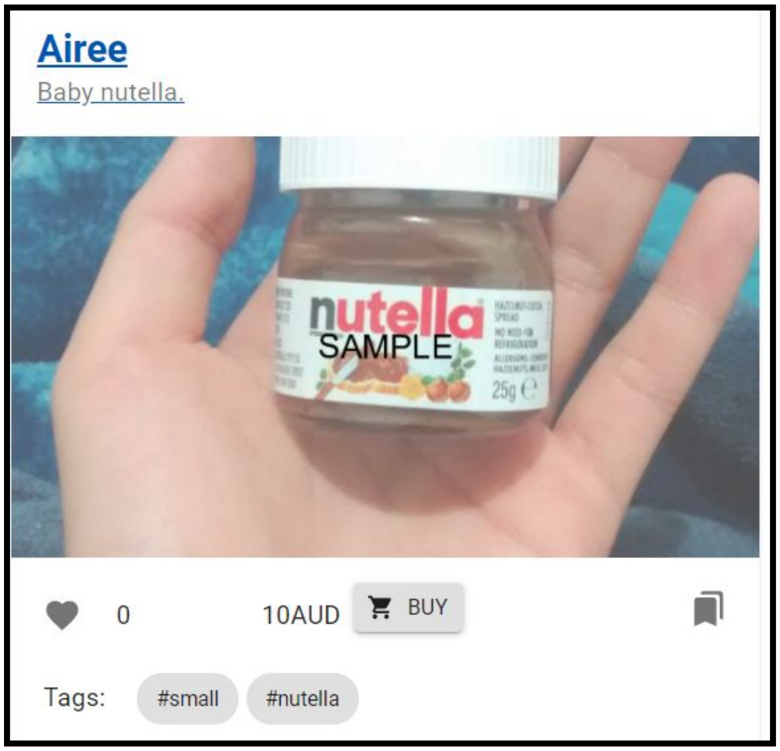


17. Search for contributors:

Searching for contributors is the same as the step above.

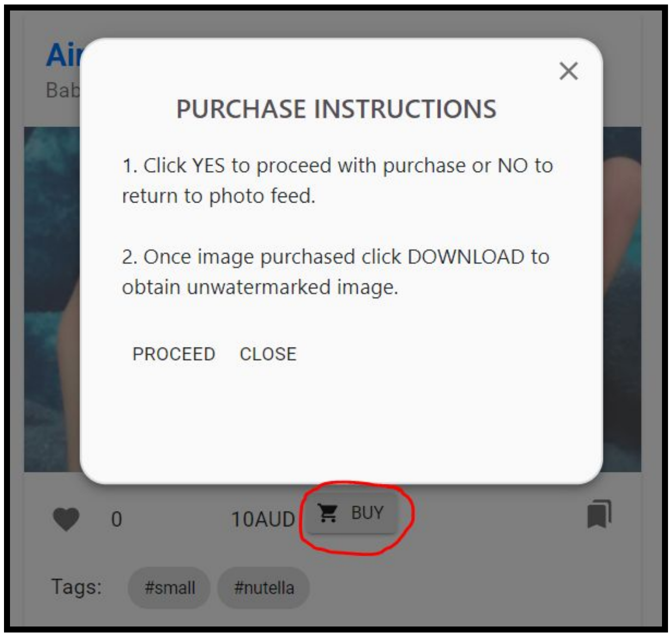
18. View photo details:

All photo details (such as price, likes, tags) are displayed with each post.



19. View purchase instructions:

Click “BUY” and the purchase instructions will pop up.



PURCHASE INSTRUCTIONS

1. Click YES to proceed with purchase or NO to return to photo feed.

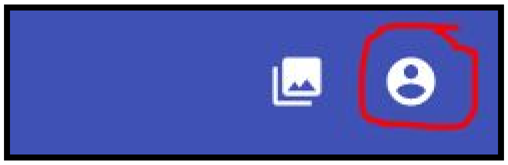
2. Once image purchased click DOWNLOAD to obtain unwatermarked image.

20. Upload photos to sell and include photo details (price, tags, caption):

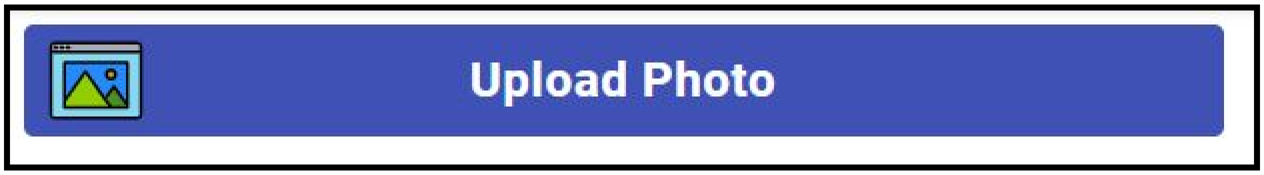
There are 2 places where a user can upload a photo.

- The first is from the homepage/photo feed page.
- The second is from their own profile page.

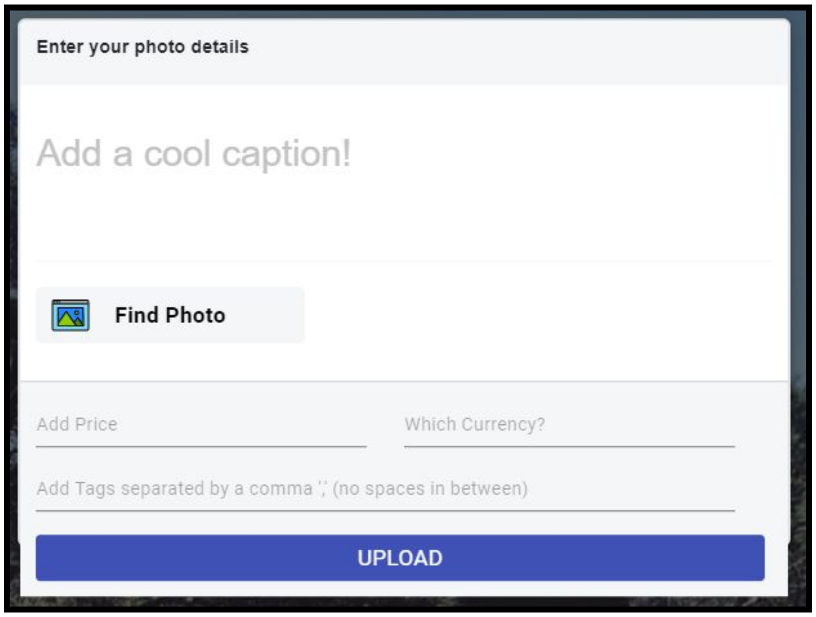
To navigate to the profile page, click the profile icon in the top right-hand corner.



Then click “Upload Photo”

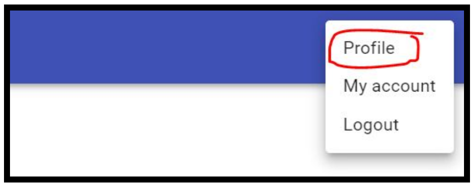


The photo will be uploaded once you fill in your details and click “UPLOAD”.

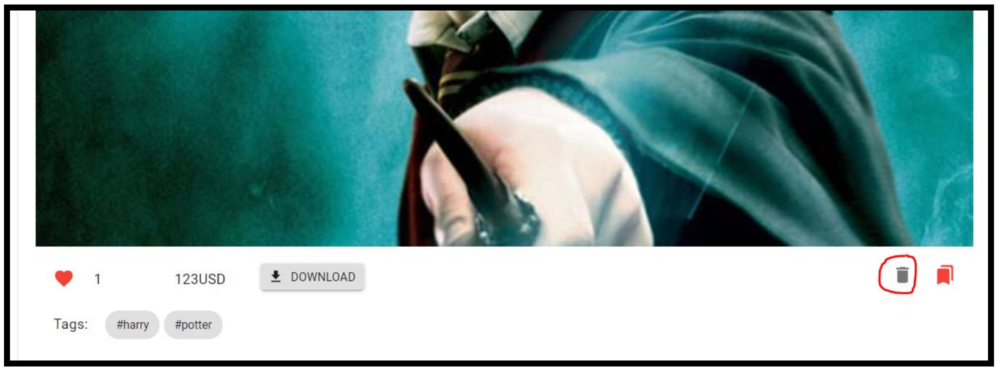


21. Remove a photo from your profile:

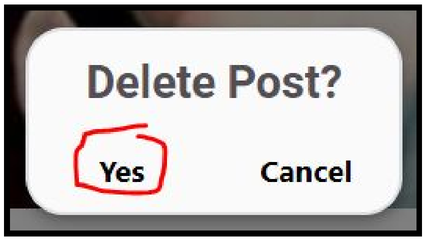
After navigating to your profile page.



Select the trash can icon of the photo you want deleted.



Click “Yes” to confirm deletion.



References

Conditional Rendering – React. 2020. Conditional Rendering – React. [ONLINE] Available at: https://reactjs.org/docs/conditional-rendering.html. [Accessed 15 November 2020].

Filestack. 2020. Filestack Transform. Powerful Image Transformations. Try Now.. [ONLINE] Available at: https://www.filestack.com/products/image-transformations/. [Accessed 15 October 2020].

Firebase. 2020. Firebase. [ONLINE] Available at: https://firebase.google.com/. [Accessed 28 September 2020]

Free API to overlay text on images. 2020. Free API to overlay text on images. [ONLINE] Available at: https://textoverimage.moesif.com/. [Accessed 15 October 2020].

Image Watermark API - Neutrino API Docs. 2020. Image Watermark API - Neutrino API Docs. [ONLINE] Available at: https://www.neutrinoapi.com/api/image-watermark/. [Accessed 13 November 2020].

Material-UI – A popular React UI framework. 2020. Material-UI – A popular React UI framework. [ONLINE] Available at: https://material-ui.com/. [Accessed 28 September 2020]

React – A JavaScript library for building user interfaces. 2020. React – A JavaScript library for building user interfaces. [ONLINE] Available at: https://reactjs.org/. [Accessed 28 September 2020]

watermark.js - GitHub. 2020. brianium/watermarkjs - GitHub Library ReadMe. [ONLINE] Available at: https://github.com/brianium/watermarkjs. [Accessed 28 September 2020]

www.youtube.com. (n.d.). Facebook Clone using React JS, Router, Hooks | Project Setup and Login Page with firebase - YouTube. [online] Available at: https://youtu.be/nJH0wUUg6EU [Accessed 16 Nov. 2020].

www.youtube.com. (n.d.). Facebook Clone using React JS, Router, Hooks | Complete Registration using Firebase - YouTube. [online] Available at: https://youtu.be/xyEXY-1e5rE [Accessed 16 Nov. 2020].