Final Project Report

collectify

App Link: https://collectify.me

CS3216 Software Product Engineering for Digital Markets
AY21/22 Semester 1

Group 8
Chan Qin Liang — A0208035N
Liu Shuyang — A0206325N
Marcus — A0194088R
Teo Jun Xiong — A0183852X

Table of Contents

Description of collectify

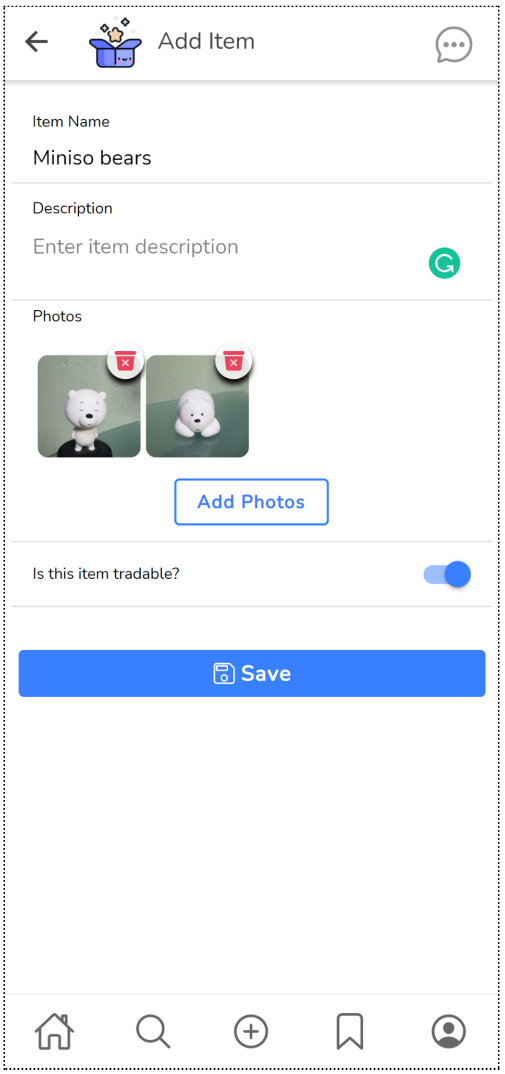
The act of collecting comes with many benefits – relaxation, stress reduction, sense of pride, sense of community. Some articles even suggest that it helps build your organisational and observational skills. In the United States, a study found that 42% of millennials and 37% of Gen Xers collect physical items as a hobby or investment in 2021.

Introducing collectify, an app that helps collectors track, build up and showcase their collections. No news, no articles, no fluff. Only collections.

Users can display their collections in a variety of categories, and view the galleries of like-minded collectors. This app aims to bring together communities of collectors worldwide and provide an easy way for them to show off their prized collections while appreciating similar collections that others have built up. If they find something they desire, they may even initiate a trade on our platform.

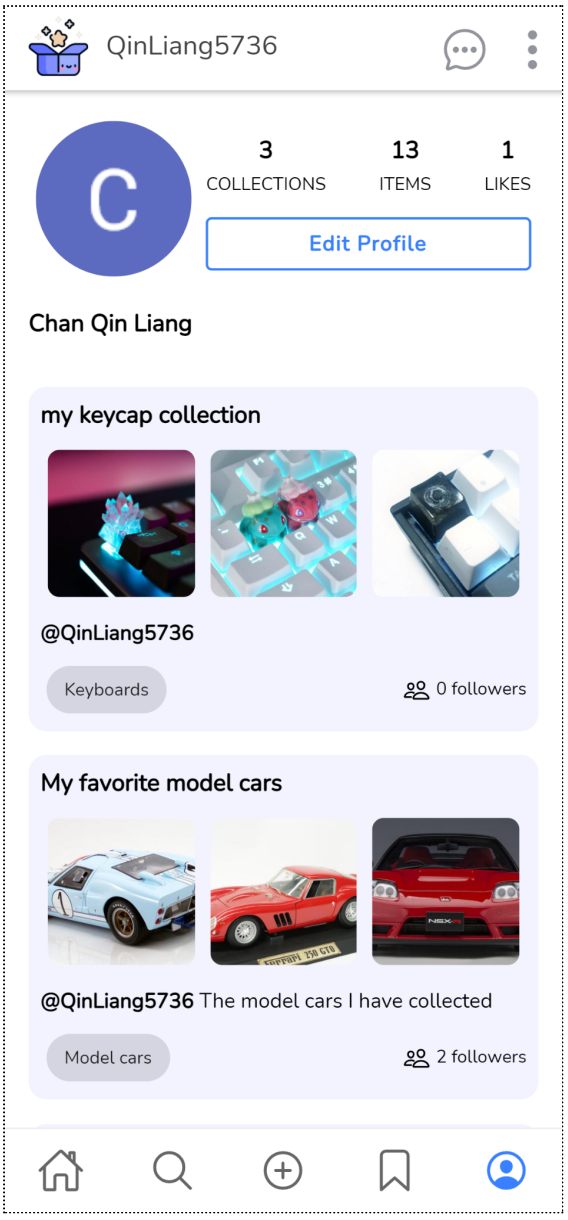
Target Users

Our intended target user was people that are collectors and also use social media. The reason is that our application is essentially a social media that is catered to collectors. Many of these users use existing platforms like Facebook, Instagram, or Reddit to share, buy, and sell their collectibles, but have expressed pain points in using such platforms, which we will address in the later sections.



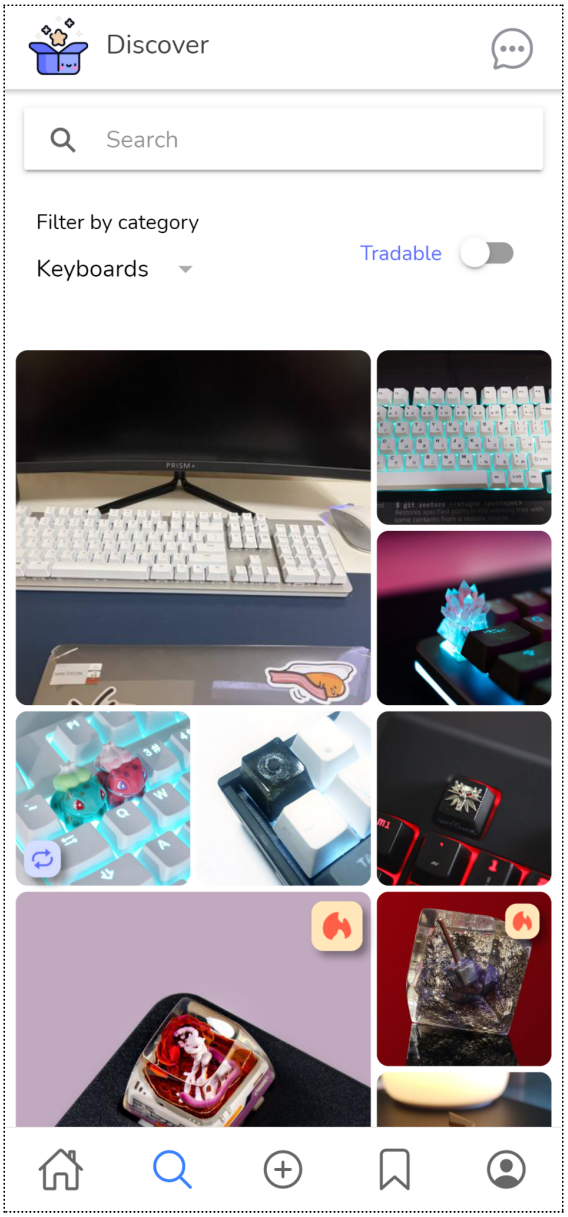
Core Features

Add and manage your collections
Users can create collections of various categories and add items to them with just a photo and a name to label them. They can also easily edit the information of the collections or individual items after creation.



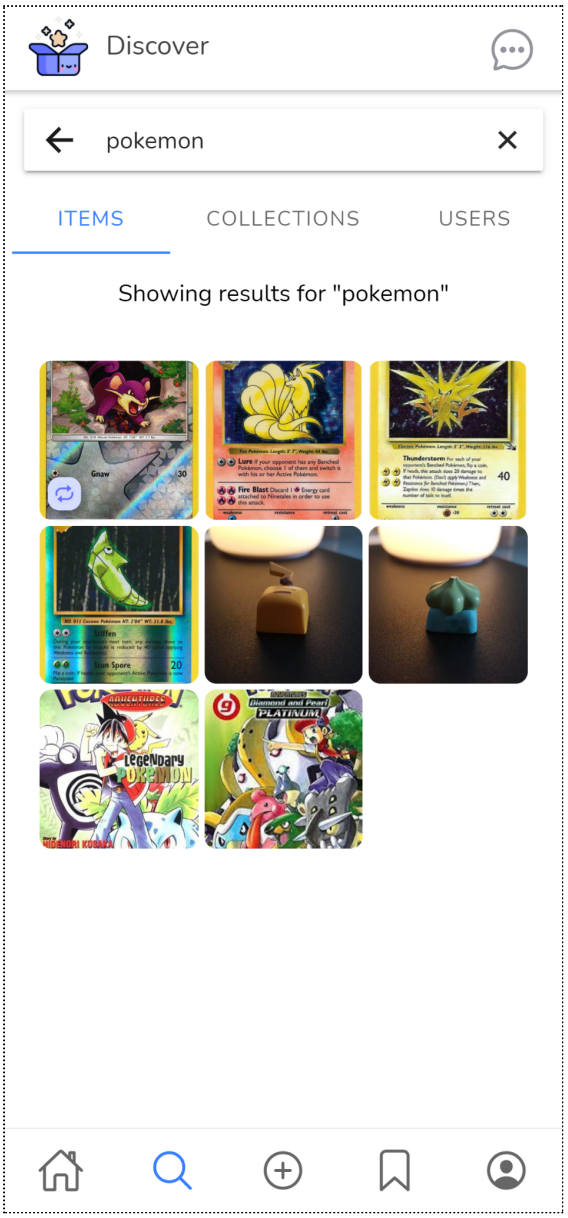
View a preview of all your collections from your profile

Since users may collect various types of collections, they can easily go back to their profiles to view any of them.



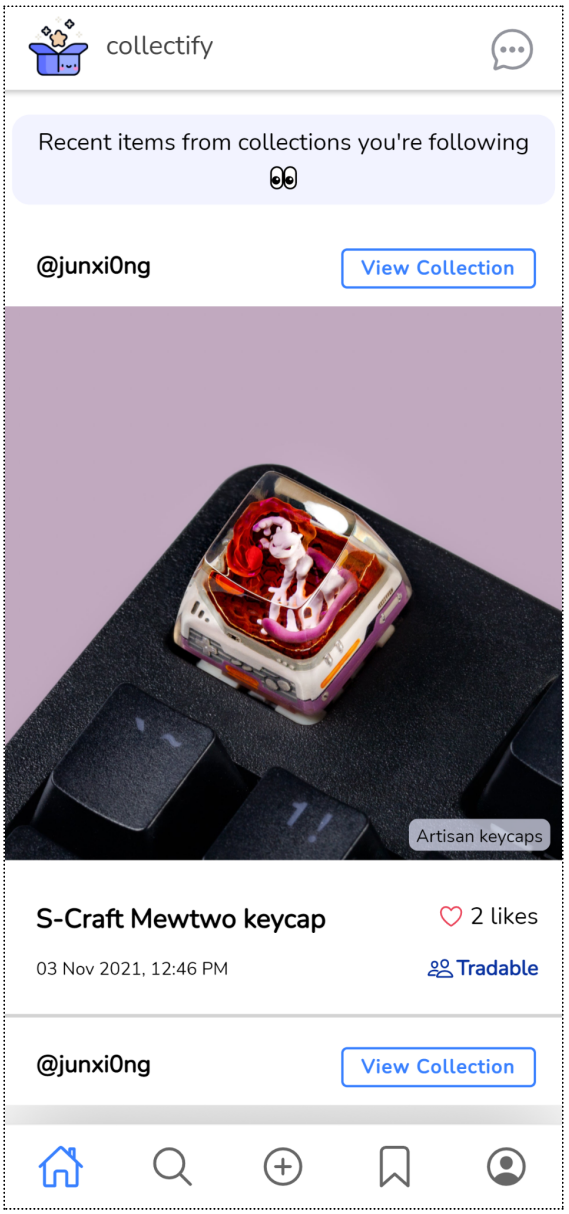
Discover new collectibles

Want to know what others are collecting? You can easily do so via the discover tab, where you can either see any new collectibles that others post or filter by a specific category if you are only interested in those. If you are just looking for new collectibles to obtain for your own collections, you can also opt to just view the items that have been marked by their owners as tradeable.



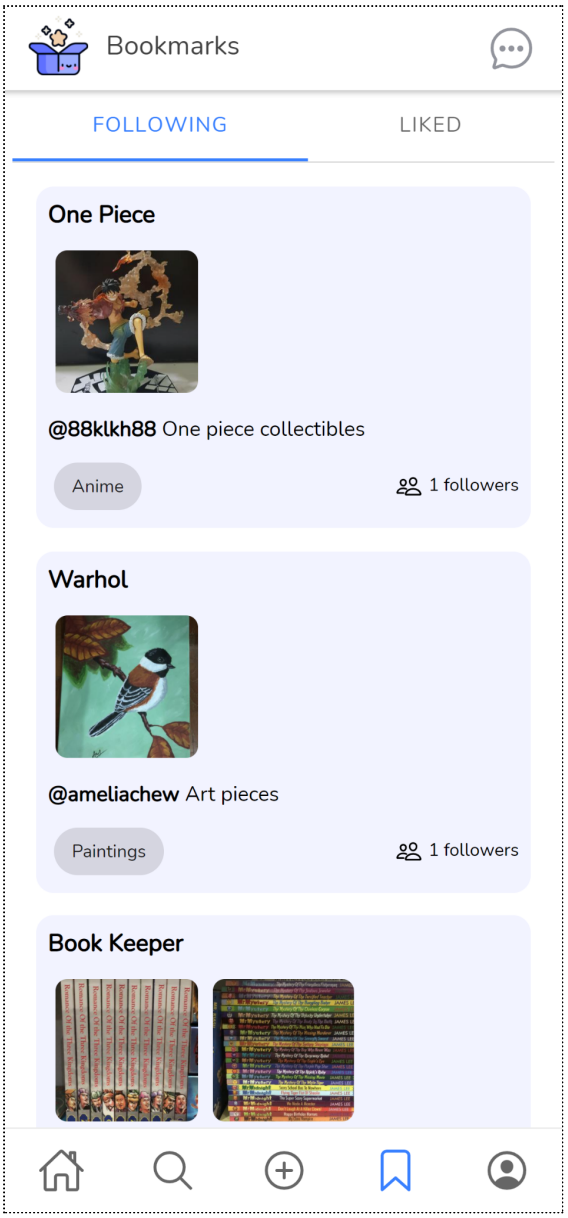
Search for specific items, collections, or users

Already know what you are trying to find? collectify makes it easy to search directly for it, right in the same discover tab.



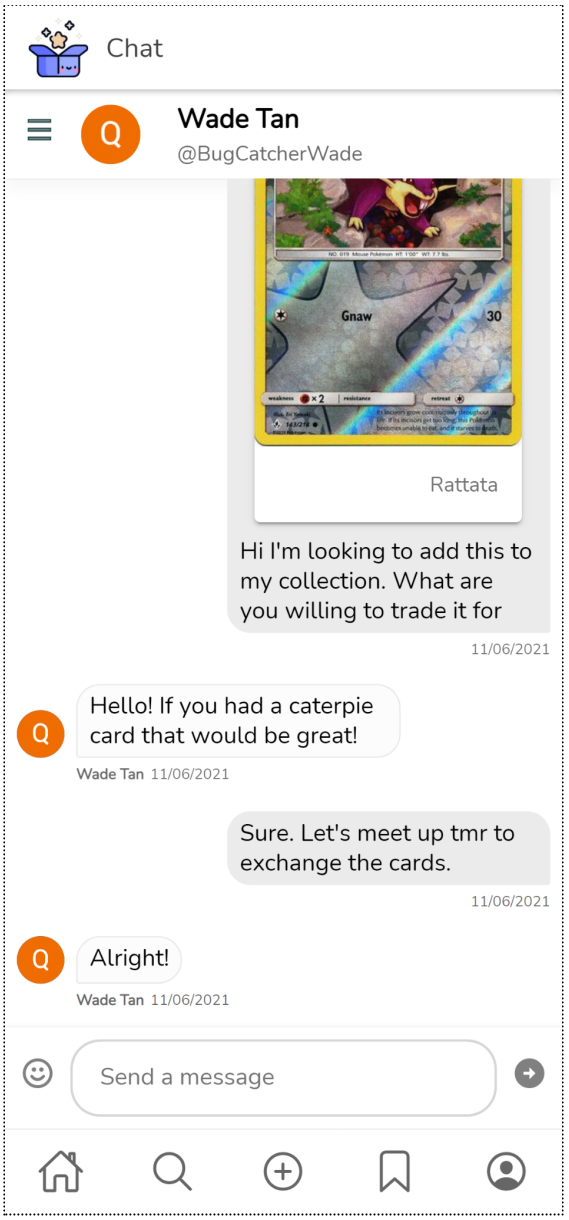
Get updated on collections you follow

See a nice collection and want to stay updated on any new items added? Simply follow the collection, and be able to view new images from all those collections directly from the feed on the home page.



Revisiting your followed collections and liked items

Want to save something for later? Follow collections or like items and revisit them easily under your bookmarks.



Chat in real-time

Want to ask about a certain item, or even initiate a trade? collectify allows collectors to initiate discussions on individual items, where you can privately discuss about the item with the owner.

Existing Applications

Existing collector-specific applications:

Hobbydb — One of the most prominent applications for collectors, but they do not have a mobile application (apart from a barcode scanner) and they are more of a marketplace and catalogue rather than a place for people to share their collections. From our competition analysis, the existing mobile applications (detailed below) are insufficient, limited, and poorly designed.

MyCollections — The user interface is outdated and cluttered, and it is difficult to pinpoint the main user task for each screen. The content is displayed like a text document, making the application feel like a collection of text documents. While there is some social aspect (friends and friends collection), the absence of a feed, likes and communication functionality limits the social interactivity between users. However, we do appreciate the detailed and highly customisable item creation, but some level of progressive disclosure would be required so as to not overload users with extraneous fields.

Memento Database — The key features of Memento Database are its clean and minimal user interface as well as its high customizability. However, the only social aspect it provides is the ability to share a certain item. As its name suggests, it is more similar to a database with a UI layer.

Existing non-collector specific applications that collectors use:
From our application, there are a few parallels that can be drawn to features of existing applications:

Social aspect: Social media platforms are interactive technologies that facilitate the sharing of information, ideas, interests, etc. via virtual communities. This entails features such as a public profile for each user and some way to interact with other users (e.g. through posts and comments). On that end, collectify allows users to showcase their galleries via their profile, and chat with other users on specific items of interest.

Public gallery: Similar to a user profile page, a public gallery can be used to showcase a user's interests. For example, Instagram stands out as it is a photo and video sharing platform where every interaction with the platform involves a visual, be it viewing someone's profile, uploading a post, or just scrolling through the feed. Since the existing popular social media is not specifically for collectors, users would have to go through content of all types when scrolling through their personal feed or when visiting a profile

page if the profile is not specifically created for showcasing collections. Users have also mentioned that due to the lack of filters other than the search function, it is more difficult to find content that they desire.

Trading: At the moment, popular platforms to facilitate the trading of collectibles are Discord, Facebook groups, Facebook marketplace, and Carousell - all of which have opt-in interest groups and discovery mechanisms (tags). For Discord channels and Facebook groups, users have to post information about trades, as a message/post. On Facebook marketplace and Carousell, users usually prefix posts with “WTT” (want to trade) on item listings. While collectify does not specifically facilitate trade at the moment, collectify provides the platform in terms of the chat feature to allow users to negotiate trades among themselves.

What makes collectify special?

There simply is no dedicated collection-sharing application out there that has the level of social interactivity one expects from such a platform. From our user surveys, we have found that most collectors do not use these dedicated applications, but rather common social media platforms such as Reddit, Instagram, Facebook, and Discord. However, these users have pointed out various flaws of using such platforms, such as the difficulty in finding what they want and having their feeds cluttered with everything else.

Hence, we believe that there is a gap to be filled in this market. collectify fills that gap by providing a platform where users’ collectibles are separated into distinct collection galleries, where users can follow specific collections to be updated only on those collections when viewing their feed, without any other fluff. Users can also discover new collectibles both through search and filtering by categories that they are interested in. collectify also provides the chat platform to facilitate discussion and trade between collectors, allowing collectors to have an all-in-one platform catered to manage and discover collections at their fingertips.

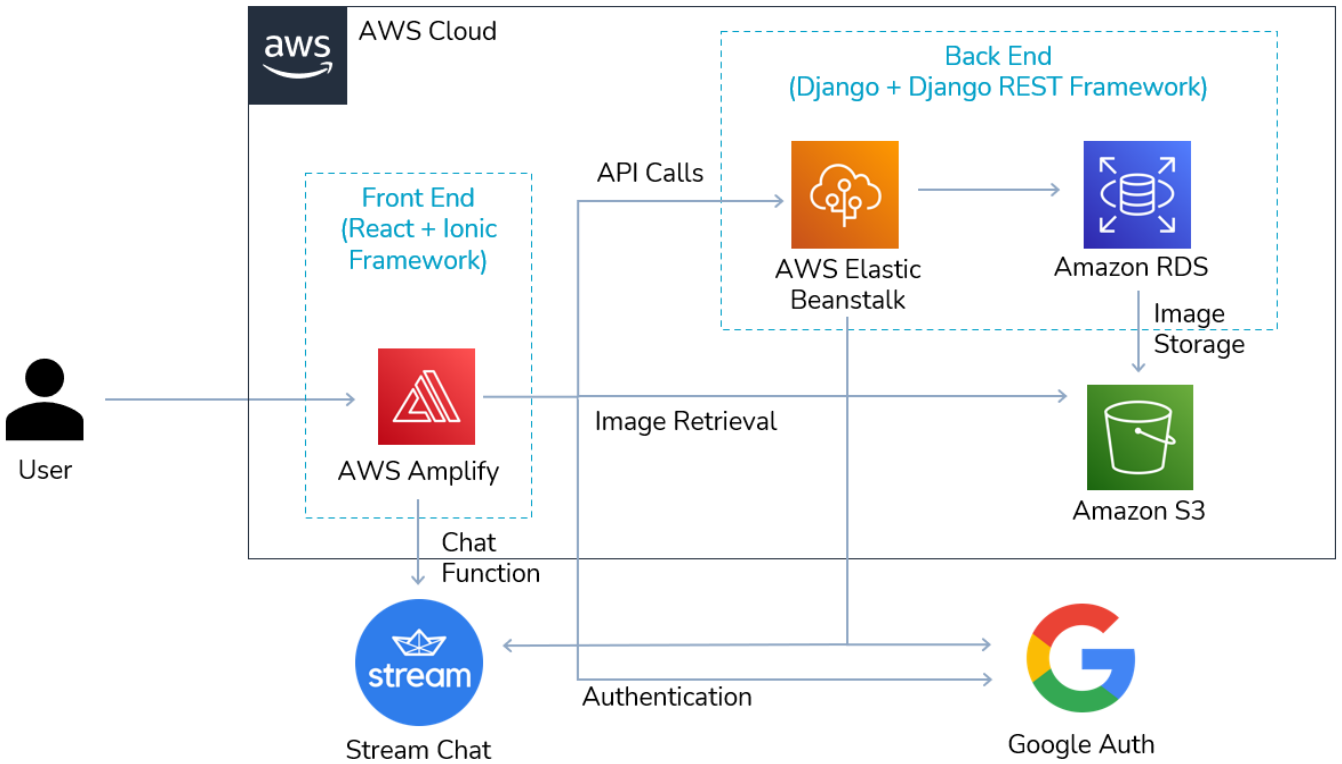
Why PWA

We chose to implement collectify as a Progressive Web Application (PWA) as this lowers the barriers to entry for using our application – the user just needs to use their browser to access our application without installing anything. Furthermore, a PWA allows users to post links to their collectify collections on existing social media platforms (and for other users to view those collections) without having to install a native mobile app.

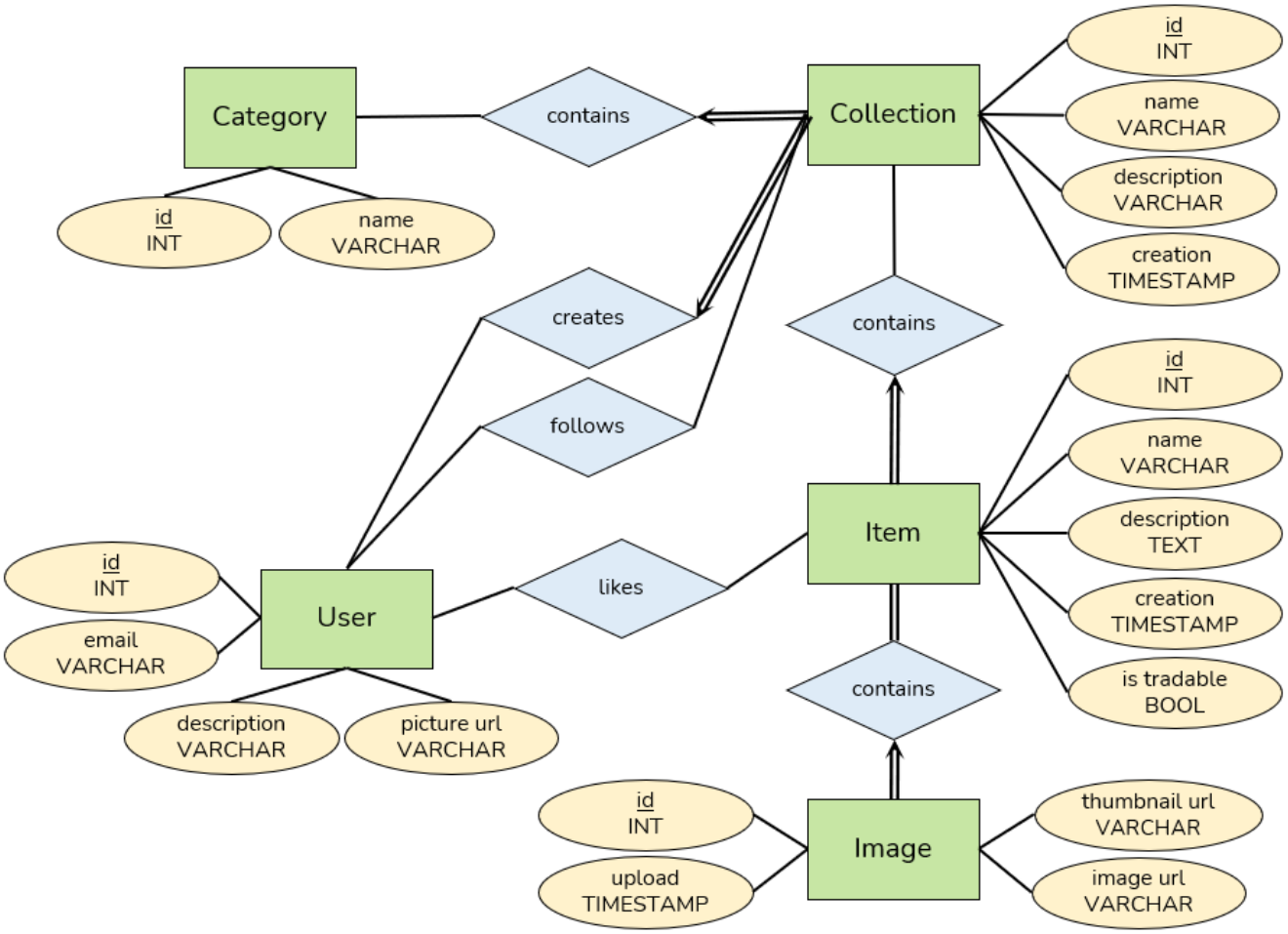
We optimised the website for mobile use as we determined that most people access existing social media through their phones through our user survey. Furthermore, it is more convenient for users to upload photos of their collections directly from their phones. Nonetheless, we have made the desktop version of the application usable and clean just in case users also want to access it from their desktop. That being said, from our user surveys, the respondents were split equally with regards to whether they preferred a mobile-friendly website or a mobile application. Hence, we implemented our application using a cross-platform library (Ionic) so that we can easily use the same codebase to build native apps when we decide to deploy them on mobile.

Application Design

High-level Architecture



Entity Relation Model



*Attributes provided by the default Django User model have been omitted

API Schema & Specification

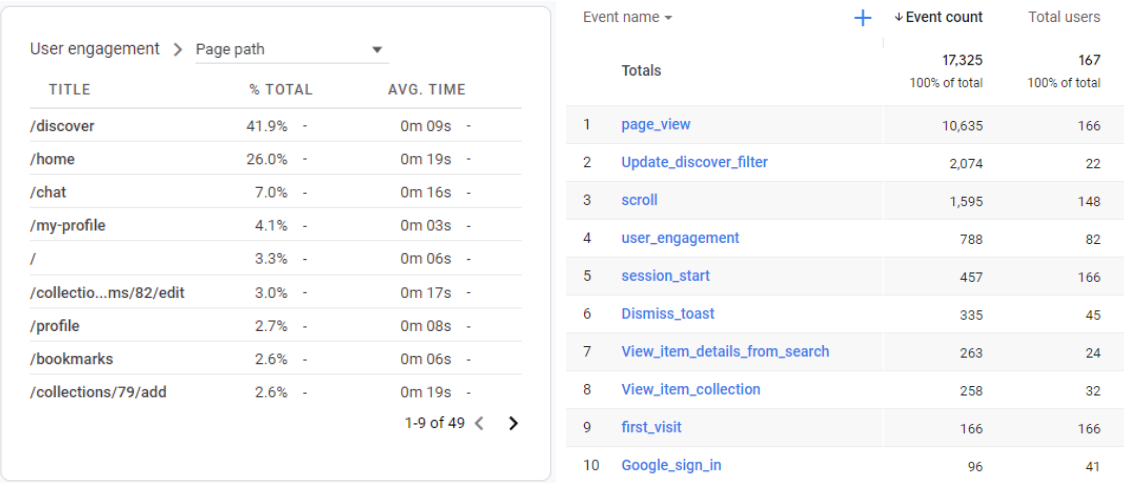
The backend of collectify is resource-oriented. Our API consists mainly of CRUD operations on the resources described in the above ER model, with each endpoint supporting multiple methods which correspond to CRUD operations.

As such, we have chosen a REST API as it is best suited to our needs. In REST, endpoints are designed around resources and the HTTP methods used at each endpoint describe the action to be performed on the resource, which is exactly what we require.

Our API documentation can be found here: API v3.0

Analytics

Google Analytics was added to track and report the web traffic flow as well as user interactions with collectify. The report can be viewed here.



User engagement > Page path

TITLE	% TOTAL	AVG. TIME
/discover	41.9%	0m 09s
/home	26.0%	0m 19s
/chat	7.0%	0m 16s
/my-profile	4.1%	0m 03s
/	3.3%	0m 06s
/collectio...ms/82/edit	3.0%	0m 17s
/profile	2.7%	0m 08s
/bookmarks	2.6%	0m 06s
/collections/79/add	2.6%	0m 19s

1-9 of 49

Event name | ↓ Event count | Total users

	Totals	17,325 100% of total	167 100% of total
1	page_view	10,635	166
2	Update_discover_filter	2,074	22
3	scroll	1,595	148
4	user_engagement	788	82
5	session_start	457	166
6	Dismiss_toast	335	45
7	View_item_details_from_search	263	24
8	View_item_collection	258	32
9	first_visit	166	166
10	Google_sign_in	96	41

Google Analytics allowed us to identify which pages most users accessed, and which pages users spent the longest time on. The screenshot on the left shows the top 3 pages in terms of time spent on it, and it is interesting to see the chat page included there. The fact that it has one of the highest average times spent also indicates the importance of a chat feature to allow users to communicate and trade with one another.

By adding handlers to all the points of user interaction (buttons, switches, etc.), we are able to gain valuable insight on what are the most accessed parts of the user interface, serving as a means of feature evaluation. For instance, the frequent usage of the discover filter indicates that many users are using it to filter for the category of collections they are interested in. With this insight, we can better rank and prioritise feature development in the future.

Lighthouse Report

The lighthouse report (HTML) can be viewed here. As explained above, for the final project, we designed collectify as a PWA as a balance between native mobile experience and development speed. As such, we used Google Lighthouse to ensure that collectify is PWA-certified. collectify achieved a score of 9/9.

Review of Milestones

The team's milestones were categorized into three different sprints, with each sprint lasting for two weeks. In terms of development, each sprint serves as an internal deadline for a specific set of features, by the end of which we aim to accomplish. In our proposal, we had our sprints planned as follows:

Sprint 1 (Week 7 - 8)	Sprint 2 (Week 9 - 10)	Sprint 3 (Week 11 - 12)
• Authentication • Profile • Category • Collection • Item • Images	• Trade / Chat • Followers • Recommendation	*(This sprint was intentionally left empty so that any unfinished features from previous sprints can be overflowed to the next)*

Sprint 1 Review

Development

By the end of sprint 1, we completed our Figma UI design, API schema design, and database schema design. We have also implemented the required endpoints in our backend and implemented the necessary components for our sprint 1 features in our frontend.

In this sprint, we did not manage to complete the integration of the frontend and backend. Thus, our app remained unusable at this point. This was attributed to the fact that this sprint coincided with our midterm weeks (Week 7 and 8). This sprint involved the most work, as it requires not only planning and designing but also implementing a starting base for our code.

As mentioned in our first progress report, we predicted that the subsequent sprints will be faster and easier to implement since it mainly involves extending additional features to an already existing codebase. This was indeed true.

Customer Contact

In order to validate our chosen problem, we created a Google form survey targeting collectors. The survey result showed that a majority of the collectors do not know of any suitable platforms to showcase their collections, and existing platforms do not perfectly fulfil their expectations for showcasing collections. We also validated the need for our features through this survey. For each feature, a majority of them agree that the feature was necessary. Spotlight, however, was considered unnecessary and thus, our team went with a Home feature that displays recent items from collections user follows.

Sprint 2 Review

Development

By the end of sprint 2, we have extended our features to include Follow, Like, Search, and Home Feed features. We have also implemented the required endpoints in our backend and implemented the necessary components for our sprint 2 features in our frontend.

We realised that the Trade/Chat feature involves additional complexities that we did not foresee. Thus, we decided to focus on completing the other features planned for this sprint. Luckily for us, we left sprint 3 as a backup sprint in case any features require more time. The Trade/Chat feature development was extended into sprint 3.

Unlike sprint 1, our team was able to complete the client-server integration for the implemented features as well as deploy them.

During the progress meeting of this sprint, our app was unsatisfactory due to two reasons:

- There was no meaningful, realistic sample data in the app. Upon starting the app, the discover page was empty without users and posts and thus, the entire app looked rather plain.
- There was a bug while attempting to upload a photo. This was revealed to be an Nginx config issue on our server which defaulted to a request size limit of 1 MB. By then, we have only tested the flows using images that are smaller than 1 MB.

Following the progress meeting, we immediately convened a meeting to discuss rectifying the two issues above. We managed to rectify the issues and received further comments from Uncle Soo regarding the slow loading of the Discover page due to large images, and specific UI designs that can be made more intuitive.

These issues were addressed and rectified in the next sprint. This sprint also taught us that proper testing and considering all the possible types of cases is very important. From this experience, we tested our implementations more carefully in the final sprint.

Customer Contact

In this sprint, we conducted several user interviews in order to validate our app design.

When we tested the initial 2 users after sprint 1 features were deployed, the users felt that the interactions were quite intuitive overall and the design seemed clean, though there are still several bugs in the website. The primary issues faced was that the users were not redirected back to the collections page after they submitted an item, and that

the usage of badges for the categories seem to suggest he could add multiple categories when in reality he could not.

After reviewing the responses, we have fixed the bugs faced by the users and improved the user experience based on feedback. After these changes and the changes from sprint 2 have been finalised and deployed, we tested another 2 users on both the old and new features, and made further improvements and polishments in sprint 3. The user feedback was extremely helpful and contributed significantly to the app's overall design.

Marketing

In this sprint, we have set up a Facebook page and an Instagram page and seeded them with posts to garner attention.

Sprint 3 Review

Development

This sprint involves finishing the chat feature, which was partially implemented in the previous sprint. Since we had quite a lot of time to spare after finalizing the chat feature, we were able to further polish our app in preparation for the 19th STePS.

On the frontend, we polished the UI and UX, including those addressed by Uncle Soo from the previous sprint. In addition, we requested a user interview session with Su Yuen, who gave very impactful insights regarding the user experience collectify provides. The main takeaways were the importance of a social element and to consider what a user wants to do on a certain page, and then streamline that process.

On the backend, we implemented compressed image thumbnails to address the long load times of the Discover page and implemented a popularity score for each item.

Marketing

We made most of our marketing efforts in this sprint. In preparation for the 19th STePS, we prepared a presentation, a poster, and a demo video. The poster and demo video were also shared through public channels on Facebook and Telegram to further boost our app's publicity and attract more users. In particular, Facebook has a lot of collector community groups of various types, which makes it a great platform to find collectors.

Individual Contributions

Name	Role	Contributions
Chan Qin Liang	Backend	● Setup Rest Framework configuration, custom permissions and authentication schemes ● Implement Token-based authentication and integrated token storage and handling on frontend ● Design and implement API endpoints for users, categories, collections, followers ● Implemented Stream Chat backend ● Created and implemented search strategy ● Optimised image load speeds with thumbnail generation
Liu Shuyang	Backend	● Django project setup and configuration ● Design and implement API endpoints for item, image, likes and item search ● Integrate Django with S3 storage ● Handle deployment and configuration of AWS including Elastic Beanstalk, S3 and RDS ● Prepare and present in-class presentation
Marcus	Frontend	● Implemented search by collection, item, users ● Implemented likes and follows ● Integrated items and collections endpoints ● Implemented edit and adding items, collections
Teo Jun Xiong	Frontend	● Designed UI on Figma ● Handled social media pages and publicity ● Integrated chat API and StreamChat UI ● Improved UX by adding GIFs to reflect system status ● Handled frontend deployment on AWS Amplify ● Implemented Google Authentication and Google Analytics

Future Plans

collectify has achieved the milestones set out in our proposal and progress reports, and as such, most of the future plans described in our proposal will remain.

Monetization
As it stands, collectify is free to use. As the number of users and data stored on our cloud database increases, infrastructure costs and the cost of external APIs (StreamChat) will increase. To generate revenue, we will adopt similar styles of advertisements as Carousell.

For starters, our app will contain in-app purchases for collectors who plan to trade their items to pay to boost some of their listings to the top of search results for a certain period of time (e.g. $0.99 for a 1-day boost, $1.99 for a 3-day boost), increasing the visibility and outreach of their listings.

We can also serve banner ads, which can be in the form of Google Ads or advertisements directly from merchants that sell collectable items (e.g. Adidas for sneakers), which gives these merchants direct access to their target customers. With the integration of mobile ads, it opens up the opportunity to allow users to purchase an ad-free version of our fee with a flat fee to improve their user experience.

User Retention
User acquisition and retention are our top priorities, and we have been able to increase our user base by advertising collectify in collectors Facebook groups, subreddits, and Telegram chats. However, user retention is not ideal. To improve this, we will integrate web push notifications to provide updates to encourage users to return to collectify, and include more social elements, such as adding group chats (or group pages). Additionally, we will require community moderators to remove offensive content on collectify. This provides users with a safer and more comfortable environment, hence contributing to user retention.

Commercial Partnership
Collaborating with brands is a growth target since brands will be able to advertise their products (which often are released in batches or collections, e.g. fall 2021 collection) on our platform, increasing collectify's user base and outreach.

Safe Trading and Payment Integration
With a growing user base that allows for the possibility of monetization, we also need a robust system that ensures the safety and security of users who choose to trade on our app. We will need a small team of administrators to manage issues and conflicts that may arise in the trading process.

Currently, collectify's chat feature provides an in-app solution to negotiate terms of trade. To better facilitate the trading process and integrate buying and selling (a common function of collector groups are BST: buying, selling, trading), we can consider a built-in payment system similar to CarouPay, where payment is only released to the seller upon receipt of items. This reduces the likelihood of scams occurring.

Insights

Software Development
Separation of Concerns
Before we even started implementing the frontend and backend code, we held a meeting to define a specific set of API that will be used for the client to communicate with the server. Thanks to this, the frontend and backend development could be implemented in parallel with the API specification in mind. Though trivial, this taught us how beneficial having separation of concerns is, as our work could be done much faster while ensuring that the frontend and backend codes are compatible and integrable with each other since the API specification was already clearly defined.

Testing
This was the most important technical lesson for us in this entire project. While we did frequent testing, it is also important to make sure that you are giving sufficient coverage to all the possible edge cases your app can face. For example, during our user interview and progress meeting, we discovered that there is a bug with the Nginx configuration in the server that limits request sizes to 1 MB by default. This prevented our user from uploading images that are greater than 1 MB. While testing the flows, we only tested with the same image which was way below 1MB. Thus, this is an important lesson for us — testing is very important because very often, things can behave in an unexpected manner. This is especially true when dealing with critical systems. In our subsequent sprint, we put in more effort to make sure that the flows were all well tested with all classes of edge cases considered.

Don't Repeat Yourself (DRY Principle)

On the frontend, we realized that some React codes will be reused in other places. To avoid repetition, we outlined some components that could be implemented as reusable:

- The `CollectionForm` component will be used for adding and editing collections
- The `ItemGrid` component will be reused for displaying liked items, collection items, and item search results.
- The `CollectionCard` component will be reused in profile, bookmarks, and collection search results.

Having a certain degree of reusability in some components saved us many lines of code and reduced the time it took for us to develop the pages.

On the backend, we also reused serializers where possible. For instance, we have separate endpoints for “item” (to return items belonging to a specific collection) and “item search” (to filter and return all items by certain criteria). These serve different purposes, but we were able to reuse the serializers from “item” for “item search”, for both the basic and detailed item view. More information can be found in our API. This not only saved us some lines of code, but more importantly standardized the return format of our API even across different endpoints.

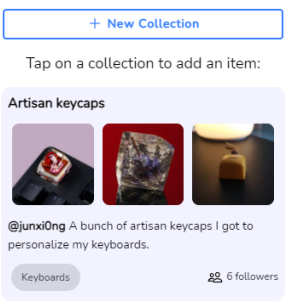
UI Design

We focused on designing a usable and intuitive user interface instead of spending time on creating beautiful assets and animations (due to the lack of a designer). Many of our iterations resulted in improvements to the user experience and the intuitiveness of our user interface. We had to conduct many rounds of user evaluation to understand how smoothly users were able to achieve certain tasks (e.g. add an item, search for items) and assess its usability based on the usability heuristics for interaction design such as:

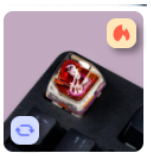
- Visibility of system status — reflecting that their item is being added instead of showing a blank page.
- Error recognition, diagnosis, and recovery — usage of toasts to inform users of errors and how to prevent them.
- Aesthetic and minimalist design — UI was designed to be minimal and clean to keep the focus on collections rather than miscellaneous visual elements.

Designing for mobile

One important insight was understanding which component was suitable for mobile. One example was using radio buttons to select a collection to add an item to, which was not intuitive on mobile, and required users to make an additional step (confirmation). After iterating the design, this interaction was replaced with a simple tap, leveraging on common mobile interaction design.



Using obvious indicators



Another example was that tradable items were indicated by a simple blue border, and users either did not realise its existence or had no idea what it represented. This was replaced by GIF icons, which better brought across its meaning.

Streamlining user flows of critical tasks

We also realised the importance of user efficiency, especially in performing critical or common tasks. For example, adding collections and items originally required users to go to their profile. This user flow was neither intuitive nor efficient. To improve this, we added a button in the bottom tab bar which allows users to quickly add a collection or item. Users can simply tap on a collection and add an item to that collection. Consolidating both functionalities also made it easy for users to remember where they can add items from.

Matching users' expectations

Search was a feature users thought was critical. However, our initial implementation was limited, e.g. searching "bear" while on the items tab will only return items with names that contained "bear" (and not collections with "bear" in its name). This linear and limited search implementation went against users' expectations because users gave feedback that they wanted to be able to see items that aren't necessarily named "bear", but could be part of a collection named "bear". We tweaked the search algorithm to be more flexible, and further improving its flexibility is a future plan.

Project Management

Frequent meetings

In our proposal, we planned for 2 meetings a week on Wednesdays 9 pm and Sundays 2 pm (this was later changed to Saturdays 2 pm). In hindsight, this was a great decision. The frequency of meetings was just right, allowing us to regularly sync up the progress of the frontend and backend teams, while at the same time not being overwhelmingly frequent. The meetings gave us an opportunity to update each other on our progress, as well as raise any concerns or difficulties faced. We also discussed, planned and assigned subsequent tasks during those meetings. When necessary, we also scheduled additional meetings in between to resolve more urgent issues or tasks.

Managing Schedule/Progress

We also gained important insight into managing project progress. From the start, we defined a realistic feature set for our project. Even so, we anticipated unforeseen delays and allocated the whole of sprint 3 as a buffer. We ended up needing this buffer in order to finish implementing the chat feature, as well as polish the UI and implement

thumbnails to improve the loading speed of images. This taught us the importance of defining internal deadlines, but also being flexible to reactively shift these deadlines based on our progress.

Prioritising features

From the project, we also realised how important it was to prioritise features wisely during our project planning. There were some features that came with much more complications during implementation which slowed down our progress in the sprints. Ideally, we would want to pick a minimal set of features that can be implemented for a working MVP, and get it implemented as soon as possible so that users can start trying out our application. Only after those features have been implemented should we start with the more complex features.

Marketing and user acquisition

Lastly, we also found that marketing and user acquisition is very difficult. For a social application like collectify where users visit the app expecting to see other users, there needs to be an initial influx of users to gain momentum. Even though we started customer contact and marketing efforts relatively early, we still found it difficult to acquire users. Furthermore, many of the users that visited our website did not post any of their own items, but rather just viewed the existing items and left. We realised that the total number of users does not really matter as much, and what matters is the users that actually come back to our application again and again. We found that most of these users are acquired only when their peers also become users or via word-of-mouth recommendations.