QuizTown

https://quiztown.fun

CS3216 Final Project Report (Group 7)

Daniel Lau Yew En
Hans Sebastian Tirtaputra
Shi Jing Lin
Yao Yuming

Table of Contents

Abstract

Project Description

QuizTown is a flashcard assistant with a focus on saving users' time. Our primary target user group is medicine students, and our secondary target user group is computing students.

QuizTown has 3 core features that help make flashcard revision more efficient for our users:

1. Users can mass-generate image cards by uploading .pdf, .png or .jpg files. QuizTown processes the images in the files to generate flashcards for review.
2. Users can explore collections shared by their friends and seniors and duplicate them for their own use. This allows them to skip the process of creating flashcards from scratch.
3. Users can enjoy automated spaced repetition interval tracking. QuizTown estimates user confidence to recommend intervals to users. On the homepage, users can start a quiz containing all cards with due intervals in just one click.

Product Differentiation

From our user interviews, we found that ANKI is the dominant application in the flashcard market. ANKI also supports image-based flashcards. From our market research, ANKI is indeed the market leader in image-based flashcard functionalities, as other flashcard assistants fall short of ANKI's features. However, users stated that the process to create these flashcards is tedious, as they have to create them individually, and for each flashcard, they must manually demarcate the textboxes. In contrast, QuizTown automatically generates flashcards for all the images in the user's uploaded file, requiring only minor edits from users to create a full collection from a textbook or a set of notes. In our user evaluation sessions, users consistently stated that they would prefer using QuizTown to other flashcard applications due to this difference in productivity.

Furthermore, users also expressed the hassle of using their friend's decks using ANKI. With QuizTown, since they are able to duplicate public collections, they can easily use their friends' decks with just a click of a button. With ANKI, their friends would need to export the decks, send it over to them via Telegram, and they can finally import the decks through the shared file.

Lastly, QuizTown's cloud storage functionality differentiates us from ANKI as our users can have all their collections synchronized across all their devices through linking their data with their Google account. With ANKI, the decks are stored locally which results in different versions of flashcards when they use different devices. Should they want to synchronize their decks, they would either export and import on a different device, or use AnkiWeb which only provides basic text-only card support. To download shared decks and take advantage of multimedia features, users must use the computer version of Anki. With QuizTown, users are able to access their collections and flashcards on the go as all of the data is stored and synchronized on the cloud

Project Timeline

Sprint 1

Sprint 1's focus was to:

1. Validate our value proposition.
2. Ensure that the technology is feasible.
3. Create a landing page to attract users.
4. Build an MVP to conduct user evaluation.

Sprint 1's goals were largely met, but the user evaluation had to be moved to phase 2 due to difficulties with the technologies used to process images. Two different libraries had to be used in conjunction to deliver reliable results. We also spent more time on building a more scalable frontend that incorporates caching, so that users will not experience slowdown when their collection and card counts increase to large numbers.

Challenges faced:
One of the biggest challenges we faced in Phase 1 was to ensure that our idea was technologically feasible. During the experimental phase, we tried out various Optical Character Recognition (OCR) libraries. As none of the libraries provides a 100% accuracy, we had to experiment with them and decide which would best fit our use case. Ultimately, we integrated two libraries to develop the feature which is sufficiently accurate for our users.

Sprint 2

Sprint 2's focus was to:

1. Polish QuizTown's UX based on feedback from users and the consultation session, and implement auxiliary functionalities such as notifications and one-click daily quizzing to facilitate this.
2. Conduct user testing to verify that QuizTown provides good UX for its core workflows.
3. Create and maintain an Instagram page.
4. Implement sharing of collections and browsing and duplicating of shared collections.

Sprint 2's technical goals were met, with one missed marketing goal as elaborated below. Through the design iterations in Sprint 2, we improved important elements such as the means users would be given recommended intervals to select from.

We also opted for a slight change in direction. We initially planned to conduct user interviews with 4 medical students and 4 computer science (CS) students. Instead, we conducted user interviews with 6 medical students. We decided that we want to make Quiztown a more focused product for medical students as their coursework is more knowledge-based, thus more relevant for our product. As such, we talked to more medical students to gain insights on how we can make Quiztown better for them as our primary target audience.

Sprint 3

Sprint 3's focus was to:

1. Polish QuizTown's UX based on feedback from users and the consultation session, and implement auxiliary functionalities such as notifications and one-click daily quizzing to facilitate this.
2. Conduct user testing to verify that QuizTown provides good UX for its core workflows.
3. Modify interval algorithm to support user behaviour of skipping cards they are confident in.
4. Bugfixing.

Sprint 3's technical goals were met, with one missed marketing goal as elaborated below. We improved the homepage to contain our core workflow without any distractions, and further validated our value proposition and user experience with users. We also received further suggestions to work on in a possible next sprint over the winter break. After a more polished MVP was created, we made contact with Dr Ang from Yong Loo Lin School of Medicine to get more medical users for user testing. Through this, we were able to get 4 more medical students to conduct user testing and further polish our user experience and core application functionalities.

What we missed:
For our marketing plan, instead of marketing Quiztown on Reddit's ANKI subreddit, we decided to focus our efforts on Quiztown's Instagram instead. The reasons for this decision are twofold: Firstly, we want to respect the rules of the subreddit and not get banned from it. Secondly, we felt that our marketing efforts would be more effective through Instagram as our following on Instagram are mostly made up of students from NUS Yong Loo Lin School of Medicine (YLLSoM) and NTU Lee Kong Chian School of Medicine (LKCSoM). This is in contrast to marketing to users from different parts of the world through Reddit's ANKI subreddit.

Overall

Our overall goals were largely met. A number of stretch goals, including features such as achievements and quiz statistics, had to be delayed due to more pressing UX improvements and features needed to facilitate them. It was more important for us to implement our core workflow well before delving into nice-to-have features.

Individual Contributions

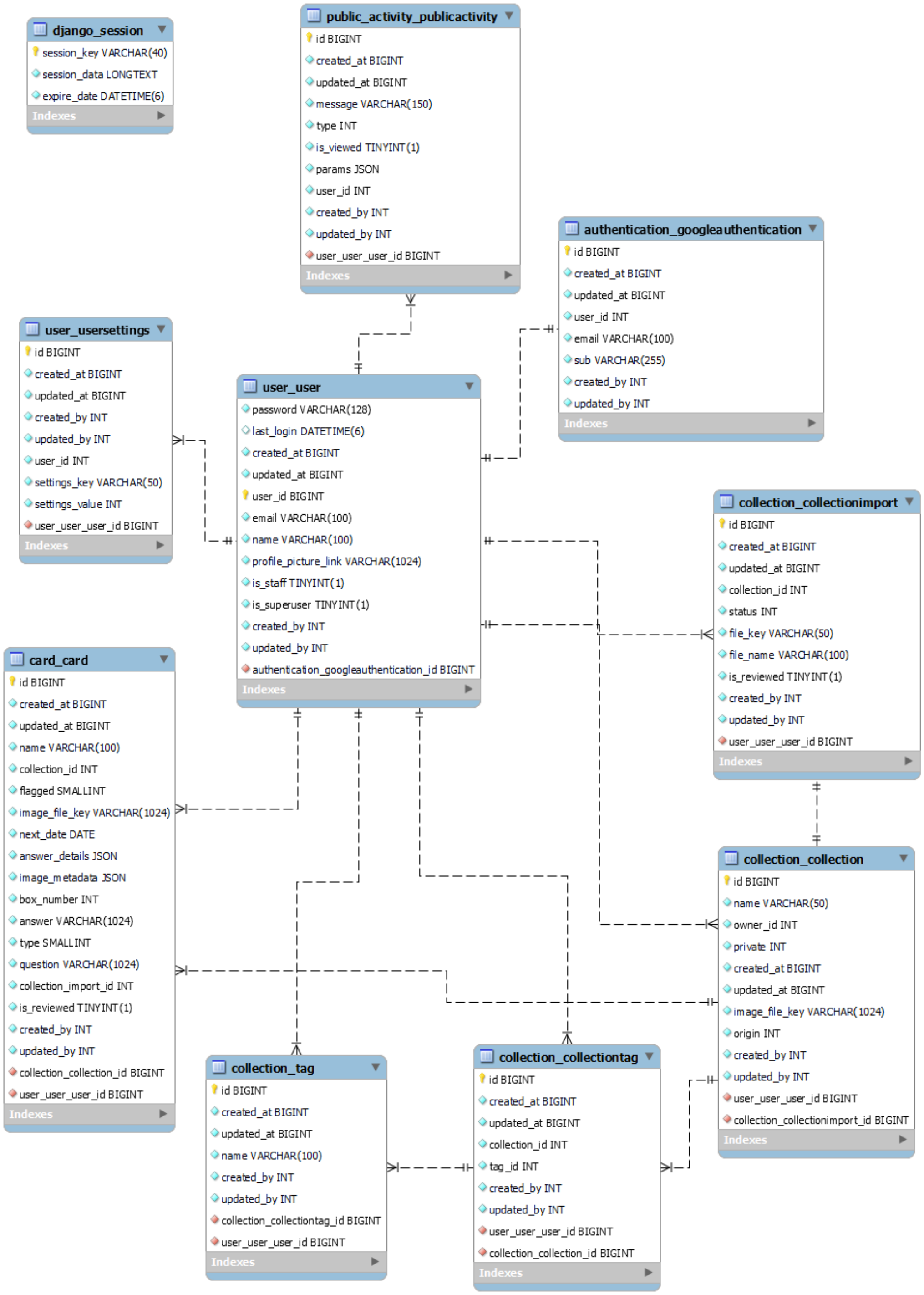
Contribution Chart

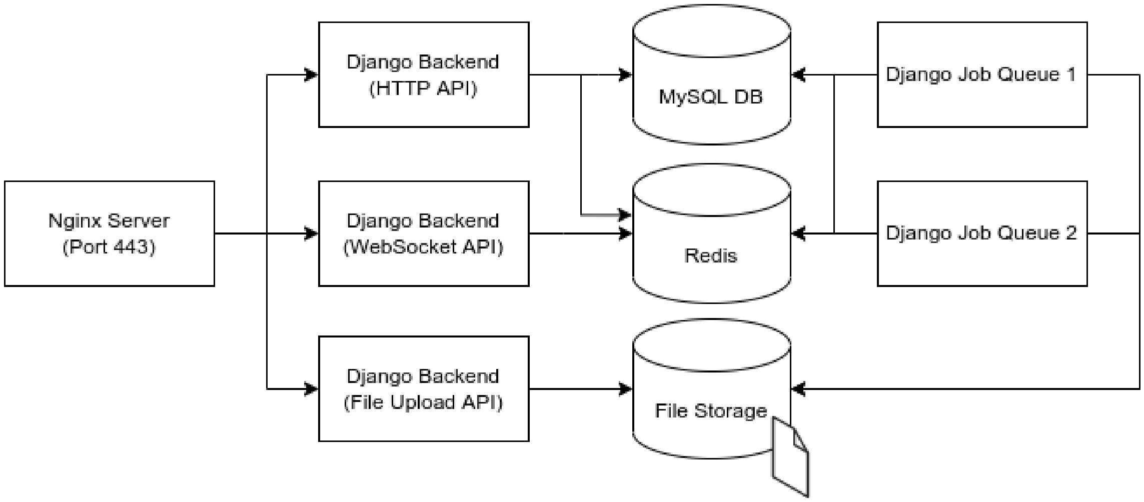
Task	Daniel	Hans	Jing Lin	Yuming
Landing Page			🟦	
User Authentication & Login	🟨			
Notifications	🟨			
Info Page			🟦	
Image Upload & Processing	🟨			
Image Flashcard Review		🟩		
Image Flashcard Display		🟩		
Text Flashcard Adding			🟦	🟥
Text Flashcard Display			🟦	🟥
Daily Quizzing & Timeline			🟦	
Spaced Rep Interval Algorithm			🟦	
Collections & Cards Pages	🟨	🟩	🟦	🟥
Collection/Card Filtering			🟦	
Discover Page	🟨			🟥
Starred Page	🟨		🟦	
Server Maintenance	🟨			
Social Media Posts & Assets			🟦	
Social Media Management		🟩	🟦	🟥
Video Filming & Editing	🟨	🟩		
User Interviews/Evaluations	2	9	4	2

Application Design

Backend

Overall Backend Architecture





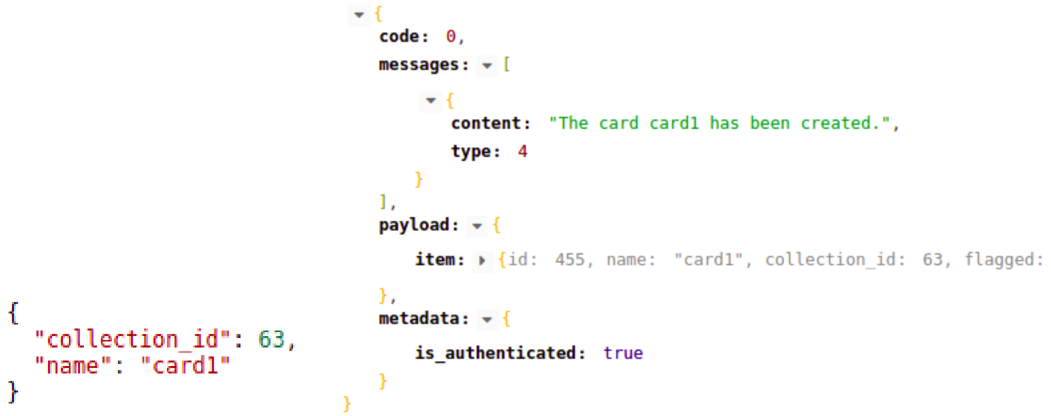
Nginx Server

The backend of the server is made up of a Nginx Reverse Proxy server, which routes each request to the respective part of the backend. The server is configured to use HTTPS, with the quiztown.fun certificate signed by LetsEncrypt.

The overall Django backend is made up of 3 parts, which is the HTTP API, websocket API and File Upload API. The HTTP API handles all requests related to CRUD, while the websocket handles the Public Activity (i.e. notification) requests of the application. The File Upload API manages all file uploads to the server.

HTTP API

The Django backend HTTP API manages all queries related to cards, collections and users, based on the database schema above (which is hosted on MySQL). It also assists in coordinating the other parts of the API, by communicating through the Redis server. The specific documentation of routes is attached in the appendix.



```
{
  "collection_id": 63,
  "name": "card1"
}
```

```
▾ {
  code:  0,
  messages: ▾ [
      ▾ {
        content:  "The card card1 has been created.",
        type:  4
      }
  ],
  payload: ▾ {
     item: ▸ {id:  455, name:  "card1", collection_id:  63, flagged:
  },
  metadata: ▾ {
     is_authenticated:  true
  }
}
```

Sample JSON Request and Response

The HTTP API backend uses the JSON format to communicate with the frontend, and it uses session tokens to authenticate users. All JSON requests contain the session token as cookie and its body contains fields that are specific to each request (e.g. card name for request to create card). On the other hand, the JSON response (as shown below) was designed to accommodate certain features in the frontend.

The JSON response is made up of 4 fields: code, messages, payload and metadata. The messages field is used to display messages to the users. This was designed with the user's experience in mind, as users would expect feedback after performing an action to better decide on their subsequent course of action. For example, after submitting a request to process a file, the backend would add a message to inform users that they would be notified once the processing is completed. Each message can be of a different type, such as to display error messages to users or to inform users that an action was successful.

The metadata is used to include additional data about the query, for example whether the request is for an authenticated user. Some of the API requests would return different results, depending on whether the user is authenticated. For example, the list collection route would return only public collections for unauthenticated users, while it will return both private and public collections when users are authenticated. Hence, there is a need to distinguish the two requests, to prevent displaying incorrect information to the users. For instance, in the case where an invalid session token is used (e.g. when it is expired), the backend will still return a valid response, but it would just be a response for unauthenticated users. Therefore, the metadata is used to help distinguish between the two types of requests, so that the frontend can serve the right information to the users.

WebSocket API

The Websocket API manages the websocket subscription to the server. Using the Django Channels library, the Redis server helps to push packets to the respective users, through the websocket protocol. This server is crucial in providing live updates to our users, particularly to alert them once their images or notes have been processed. The websocket is required to connect over the WSS protocol, and hence its contents are also secured by encryption.

File Upload API

The File Upload API manages all requests related to file uploads. Currently, the File Upload API is part of the Django Backend, but it can be easily separated as it is a separate component of the backend. The reason for this separation is that file uploads generally take up more time, as it primarily consists of IO operations. It can potentially be a bottleneck to the overall Django backend server, affecting the response time of other requests made by users. Hence, the separation would come in useful as the number of requests increase in the future.

To standardize between the different APIs, each uploaded file is given a unique randomly generated key (128-bit) to identify itself. Similar to the S3 structure, the file key will then be passed to the different components, such as the job queue, to specify which file is to be processed.

The separation of concern is also a security consideration. By separating the file upload from the original server, we would be able to impose different restrictions on the request it receives. For example, to allow for larger file uploads, we configured Nginx to allow requests up to a size of 100MB for this API. As this is not a blanket rule, the other servers would not be affected, and will drop any requests that are larger than 1MB, thus preventing large packets from malicious users.

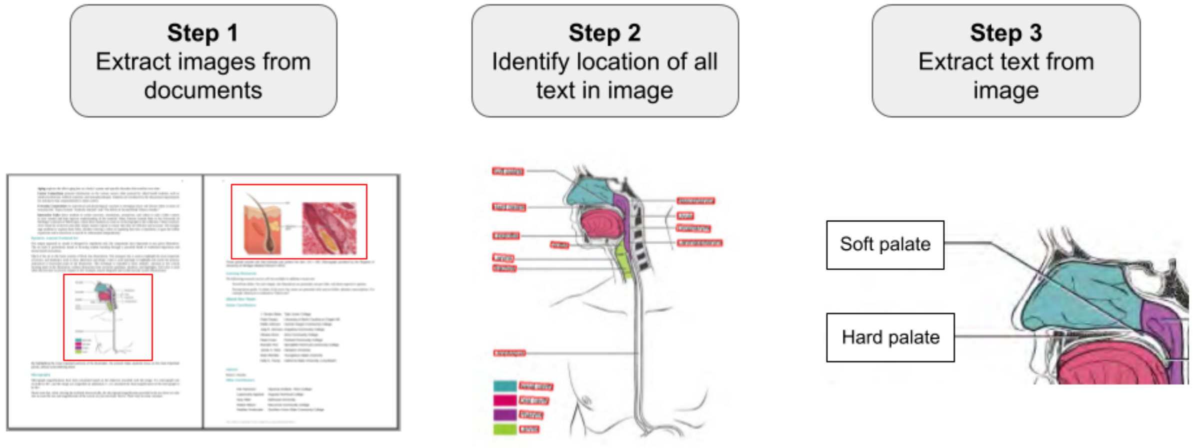
Job Queue

The last component would be the Job Queue. To process the uploaded images, each image typically takes about a minute, and a document can take up to a few minutes depending on the number of images it contains. As the time taken to process each image is relatively long, a job queue is required to efficiently manage the jobs and processes. The HTTP API server will send a request to the job queue through Redis, to enqueue the task to process a document. The job queue will then process each task sequentially, using the algorithm specified below. Once completed, the job queue will send a message to the respective user, by communicating with the websocket API server through Redis.

The job queue model was selected to support scalability. Each worker for the job queue is an independent instance on the server, and thus a new worker can be easily deployed to increase the throughput of the processing. Currently, we have two workers deployed on the server, which allow for two jobs to be processed concurrently.

Another possible method to handle jobs is to start a new thread off the backend server, when the server receives a request to process long tasks. However, in this case, there might be an issue of race condition, as the different threads would be sharing the same resource (e.g. PyTesseract and PaddleOCR) to process the image. Another downside of the thread model is the possibility of creating an infinite number of threads, if the number of threads is not properly controlled (e.g. using thread pools). Hence, we adopted the job queue model instead to process the different tasks in the server.

Image Generation Algorithm



Step 1
Extract images from documents

Step 2
Identify location of all text in image

Step 3
Extract text from image

We designed the above algorithm to process documents (e.g. notes, textbooks) uploaded by the users. First, we had to extract all images from the documents, such as PDF. After extracting all images, we process each image by first identifying the location of text within the images. This methodology was selected as it uses a deep learning model (trained by PaddleOCR), which has a higher accuracy in identifying location of texts within images, in comparison to the traditional OCR counterparts. At this point, the algorithm also filters out some images (i.e. images that have less than two strings of text), to ensure that only images that are diagrams will be captured by the application. The particular number was selected as images with only 1 text option would not be useful when used together with our drag-and-drop interface.

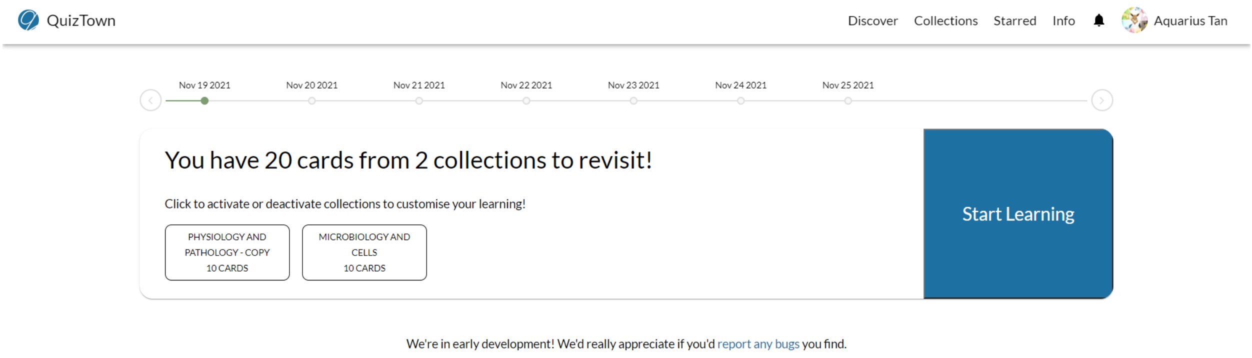
Subsequently, for each identified location, we identify the corresponding text in Step 3, using PyTesseract. The rationale for using an additional library is that PyTesseract was better at identifying words with the appropriate space between them. Hence, we had to combine the two libraries to achieve a higher overall accuracy for the algorithm.

Although the underlying libraries allow for multiple languages, the algorithm currently only supports text in English. This is because we found that by restricting the language used, the accuracy of the combined approach would be higher. And since our users mainly study using English, this would be a viable trade-off as the algorithm would work better.

Frontend

For brevity, only pages involved in the core workflows are covered in this section.

Homepage



You have 20 cards from 2 collections to revisit!

Click to activate or deactivate collections to customise your learning!

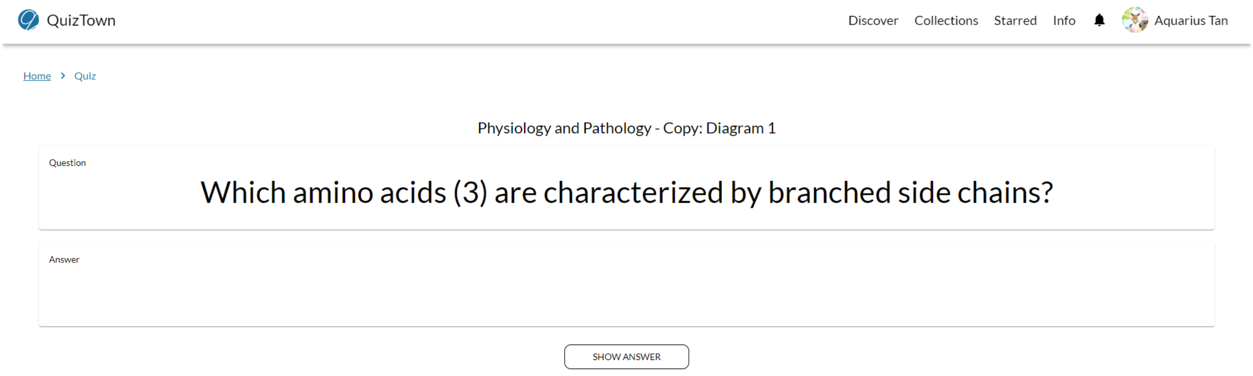
PHYSIOLOGY AND PATHOLOGY - COPY
10 CARDS

MICROBIOLOGY AND CELLS
10 CARDS

We're in early development! We'd really appreciate if you'd report any bugs you find.

Our homepage focuses on facilitating the user's core workflow: quizzing themselves on the cards due today. Users can enable/disable collections they wish to do by clicking on the associated box, and start their quiz via the large "Start Learning" button. If they wish to, they can also do their cards in advance by moving along the timeline above the main card. This is useful for targeted revision involving specific collections for a paper the next day.

Quiz



Physiology and Pathology - Copy: Diagram 1

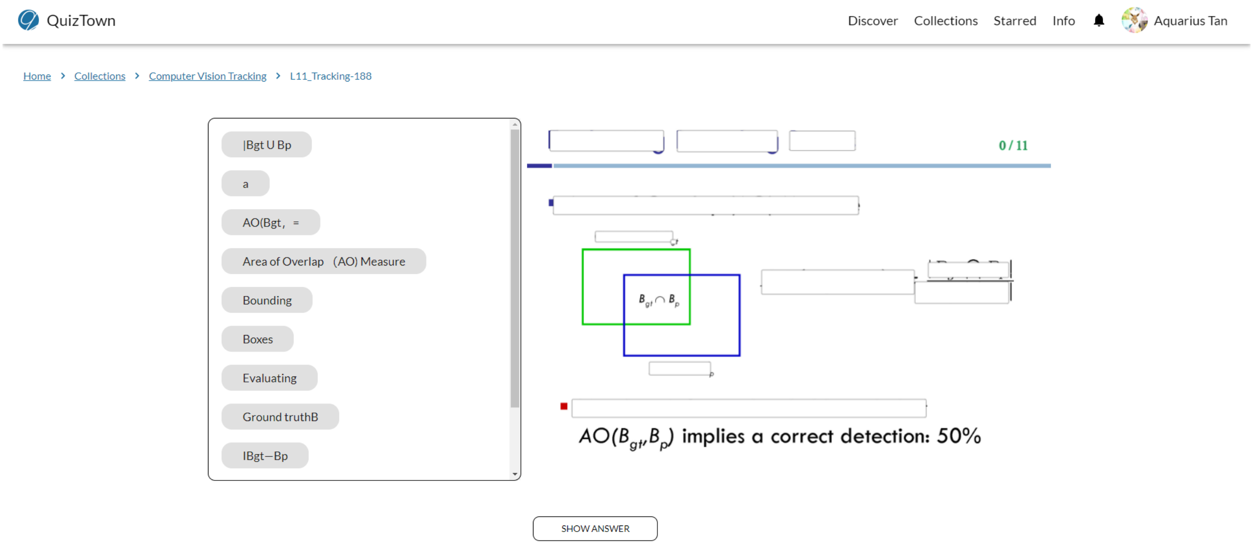
Question

Which amino acids (3) are characterized by branched side chains?

Answer

Text-based Interface

The interface for text-based cards follows largely from ANKI, as it is not our core value proposition. However, we felt that it was an important feature for the completeness of QuizTown.


QuizTown Discover Collections Starred Info Aquarius Tan

Home > Collections > Computer Vision Tracking > L11_Tracking-188

|Bgt U Bp

a

AO(Bgt, =

Area of Overlap (AO) Measure

Bounding

Boxes

Evaluating

Ground truthB

IBgt−Bp

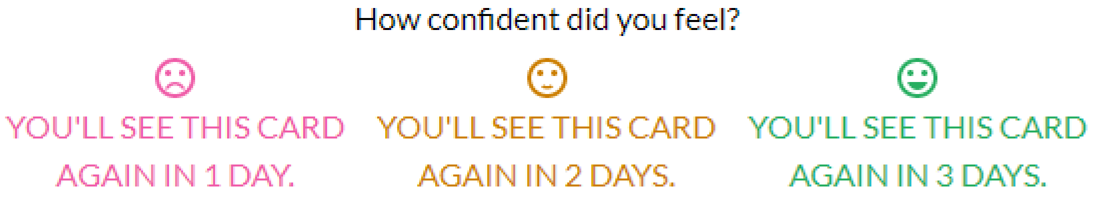
0 / 11

AO(B_{gt},B_p) implies a correct detection: 50%

SHOW ANSWER

Image-based Interface

Our interface for image-based cards allows users to drag-and-drop answers onto the empty boxes for a more fun and interactive quizzing experience. This is facilitated using FabricJS.


How confident did you feel?

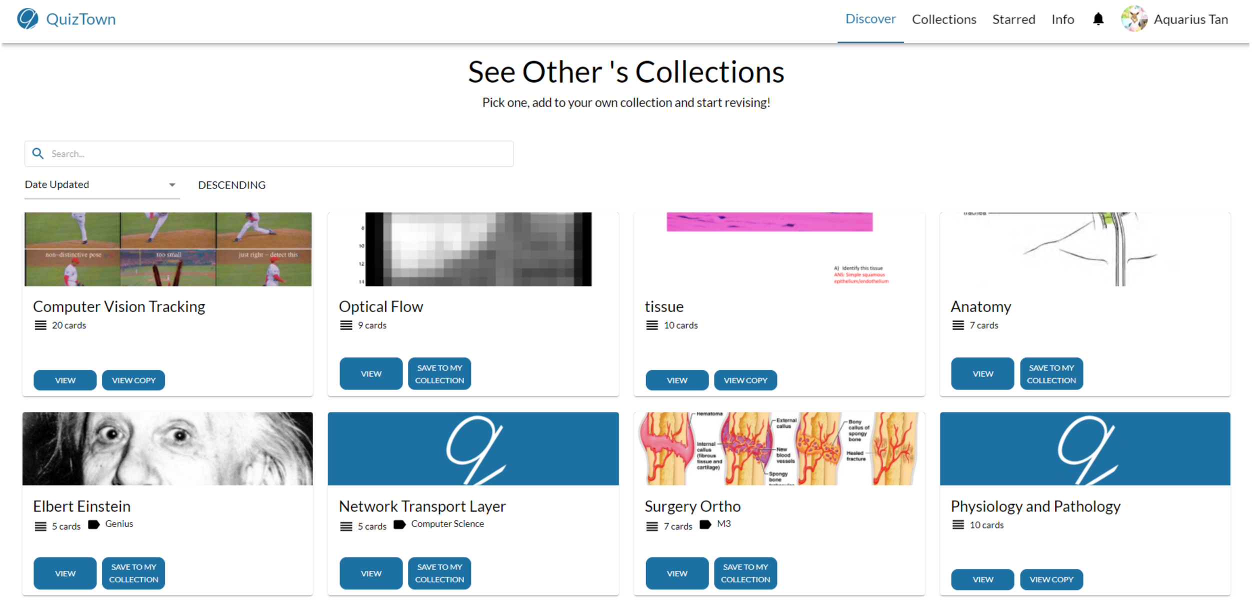
YOU'LL SEE THIS CARD AGAIN IN 1 DAY.

YOU'LL SEE THIS CARD AGAIN IN 2 DAYS.

YOU'LL SEE THIS CARD AGAIN IN 3 DAYS.

Completing a card (by drag-and-dropping all the answers correctly, or clicking “Show Answer”), will display options for the user’s next interval. The first option is always the same, as users will want to review the content as soon as possible if they are not confident. The other options are tailored based on how much time the user spent on the card as well as the number of incorrect and correct attempts. Once the user selects an interval, they will be brought to the next card in the quiz.

Discover



See Other 's Collections

Pick one, add to your own collection and start revising!

Computer Vision Tracking
20 cards

Optical Flow
9 cards

tissue
10 cards

Anatomy
7 cards

Elbert Einstein
5 cards Genius

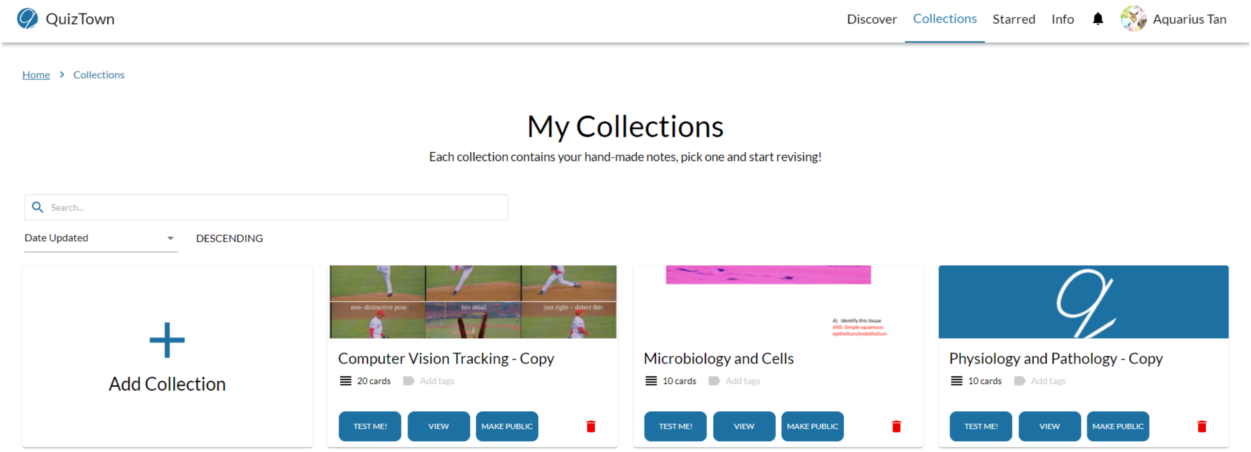
Network Transport Layer
5 cards Computer Science

Surgery Ortho
7 cards M3

Physiology and Pathology
10 cards

On their discover page, users can find collections other users have shared. They can try out the collections and duplicate them to their own collection. Searching and sorting are provided to facilitate finding a friend's collection.

Collections



My Collections

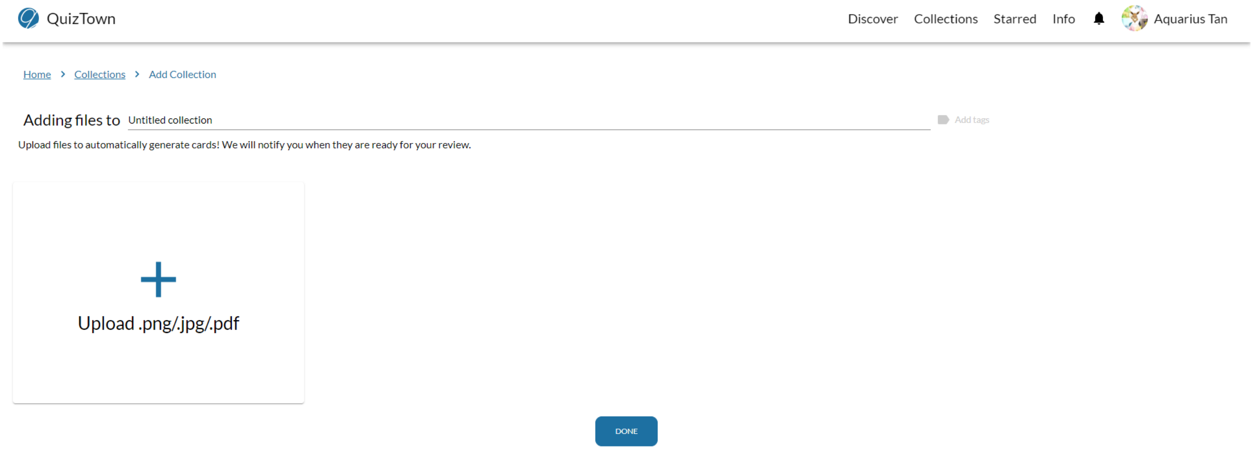
Each collection contains your hand-made notes, pick one and start revising!

Computer Vision Tracking - Copy
20 cards

Microbiology and Cells
10 cards

Physiology and Pathology - Copy
10 cards

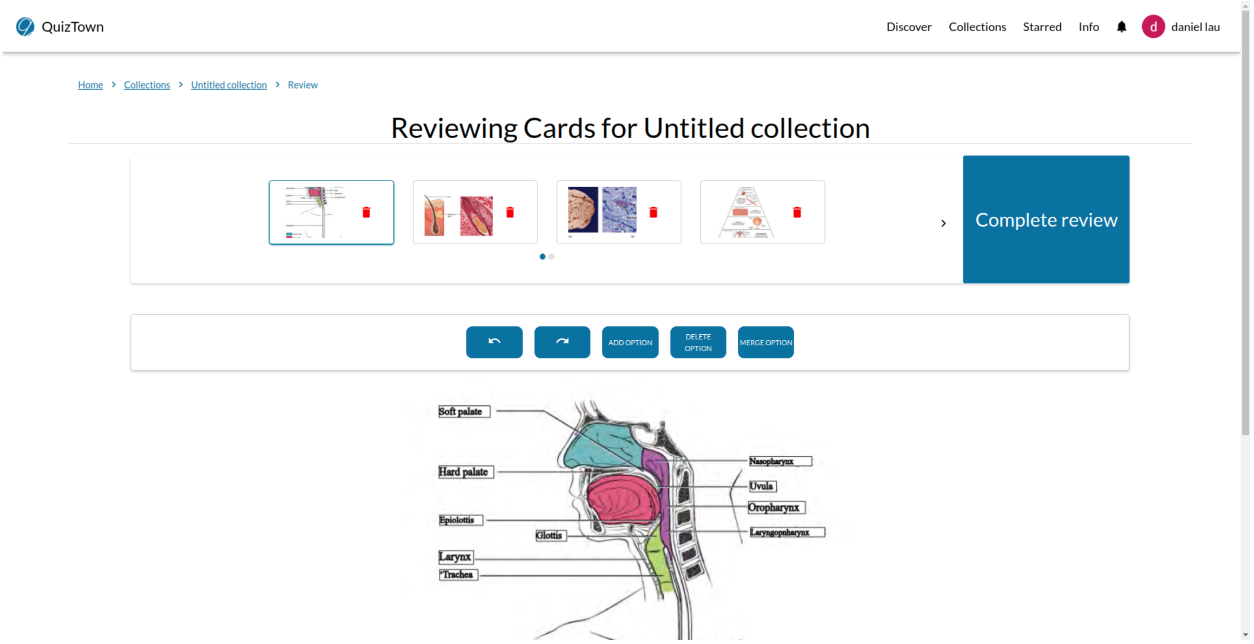
On their collections page, users can manage their collections. They can add a new collection, test themselves on a specific collection, and choose to make their collections public.



Adding files to Untitled collection

Upload files to automatically generate cards! We will notify you when they are ready for your review.

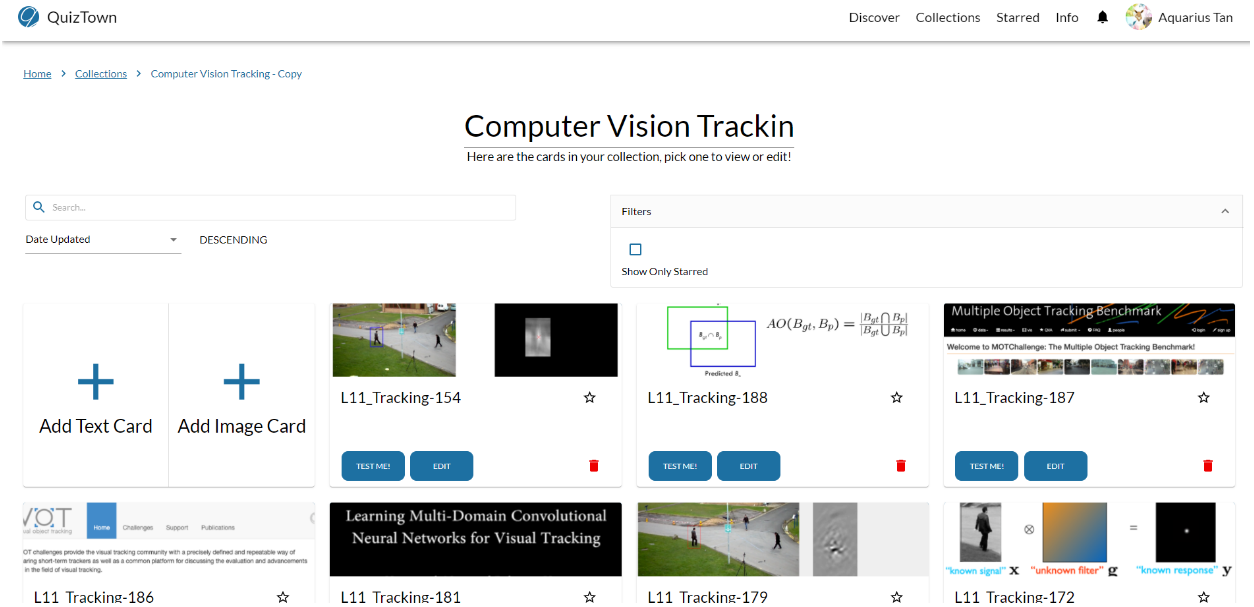
When the user clicks “Add Collection”, they are brought to this page where they can upload a .png/.jpg/.pdf to start a collection off. They can also choose to immediately create the collection without any cards.



Reviewing Cards for Untitled collection

Once the file is processed, users can review the generated cards. Here, they can add/delete boxes and merge boxes. Once they are satisfied, they can complete their review and the generated cards will be added to the relevant collection.

Cards



Computer Vision Trackin

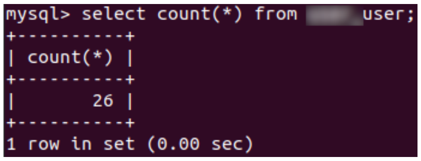
Here are the cards in your collection, pick one to view or edit!

On this page, users can manage their cards. They can add a new card, view or edit a card, and test themselves on a particular card.

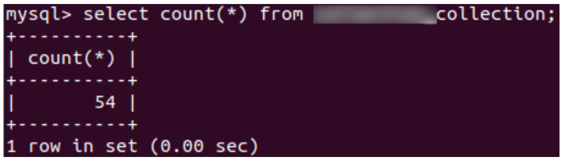
User Statistics

Website

Below are our insights from our server. As of 19 November 2021, we have a total of 26 registered users and 54 collections that were created by them.

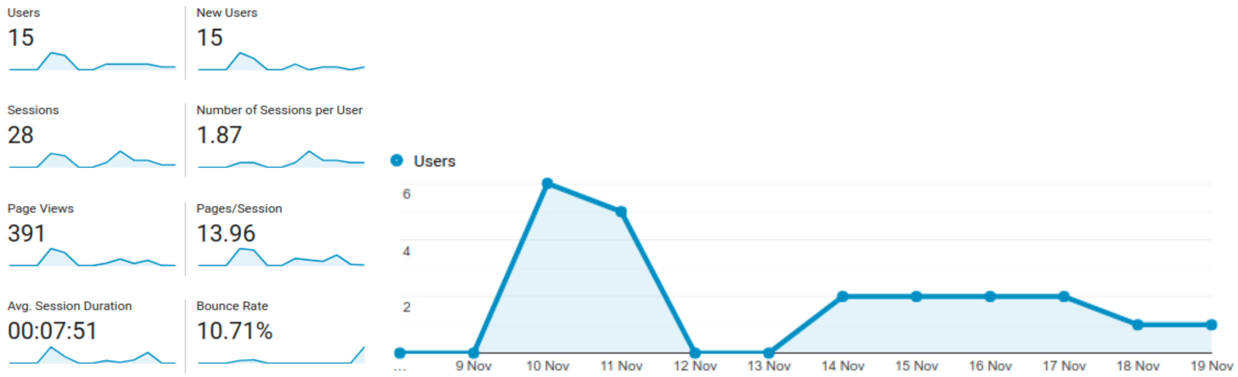


```
mysql> select count(*) from [redacted]user;
+----------+
| count(*) |
+----------+
|       26 |
+----------+
1 row in set (0.00 sec)
```



```
mysql> select count(*) from [redacted]collection;
+----------+
| count(*) |
+----------+
|       54 |
+----------+
1 row in set (0.00 sec)
```

In terms of Google Analytics, as of 19 November 2021, we have a total of 391 page views and 15 users over the period that we have deployed. The discrepancy is due to the late integration of Google Analytics into our application.



Users 15

New Users 15

Sessions 28

Number of Sessions per User 1.87

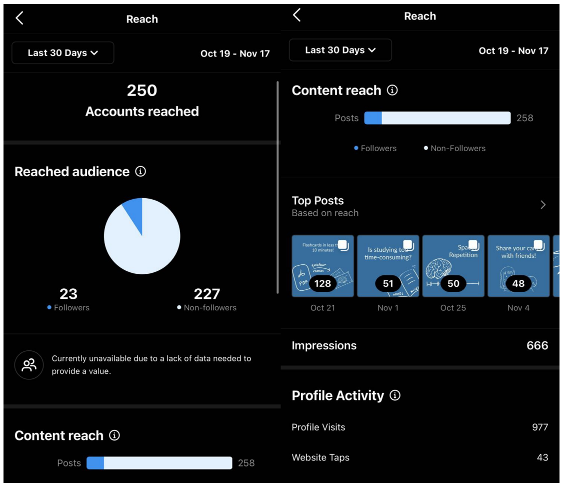
Page Views 391

Pages/Session 13.96

Avg. Session Duration 00:07:51

Bounce Rate 10.71%

Instagram



Reach

Last 30 Days ⌄ Oct 19 - Nov 17

250
Accounts reached

Reached audience ⓘ

23 ● Followers

227 ● Non-followers

Currently unavailable due to a lack of data needed to provide a value.

Content reach ⓘ

Posts 258

Reach

Last 30 Days ⌄ Oct 19 - Nov 17

Content reach ⓘ

Posts 258

● Followers ● Non-Followers

Top Posts
Based on reach

Oct 21	Nov 1	Oct 25	Nov 4
128	51	50	48

Impressions 666

Profile Activity ⓘ

Profile Visits 977

Website Taps 43

Through the span of our online marketing campaign on Instagram (@quiztownfun), we managed to get 43 website visits and reach 250 accounts.

Future Plans

The first future addition is the ability for users to collaborate on creating new collections. From our earlier user interviews, users told us about the trouble of using their friends' decks in ANKI. In QuizTown, we introduced a solution - the duplicate public collections feature, which allows our users to easily use their friends' collections with a simple click of a button. From our latter interviews, our users expressed that this feature is indeed helpful, and they believe that it will bring much convenience when they want to do revision preparation together with friends. Due to the significance of this collaboration idea, our team decided that we should bring the solution to a higher level, to not just allow users to make copies of their friends feature, but to allow collaboration among users when creating the same single collection. This means once a change is made to the shared collection, all the collaborators will be able to view the changes made, and to decide whether they want to keep the changes in their own copy. This removes the hassle of needing to delete the older version and making a copy of the updated collection each time some cards are edited or newly added.

Next, we will also allow users to import notes from different file types. We realize that many of our potential users are currently using ANKI, and we want to help them make a seamless transition to using QuizTown, which is much lightweight, has better UI and better support for image cards, just to name a few good points. Therefore, importing cards from ANKI is another future feature that our team decided to bring out.

Last but not least, our group also plans to allow handling of videos in QuizTown. From our meeting with Dr Ang from YLLSoM, we realized that videos are quite often used in teaching to help explain certain concepts better. We hope to allow not just static images, but also allow zooming in and out of videos for students to better understand the micro aspect of the structures. Unlike image cards, video revision cards will not have answers to them. Enabling videos in QuizTown will allow the users to better organize their revision materials into just a few collections. We will still allow users to choose their confidence interval based on how well they understand and memorize the content in the videos, so that they can better plan out their studies and still benefit from the spaced repetition algorithm.

Insights

A valuable lesson that we learnt from building Quiztown is the importance of getting our product out fast so that you can get real users to try it. This allows us to get more real feedback to iterate and improve on. Since we were building an application that was targeting primarily medical students, and none of us were medical students, many times our discussions would be based on what we think our users wanted. We found that sometimes, we don't know what our users want. The best way to find out is to prototype quickly and get the users to try it out. Thus, we found that being able to iterate quickly upon receiving feedback is important in polishing our product in such a short project timeline.

Furthermore, we learnt that support from individuals associated with the target group is extremely important. As we are not part of our main target group, our connections with medical students proved extremely valuable, as they were able to further connect us with more medical students interested in participating in user interviews and/or using the product. Although developmental work is important, gaining support is also important to get more feedback and a larger user base to further attract more users.

Another insight that we gained from this project is the importance of consistent and clear communication, both at the individual and project level. At the individual level, each of us had a lot on our plates in addition to the tight project schedule. Because of this, communicating clearly with our teammates helped to keep us all on the same page. It helps to prevent conflicts due to different expectations between team members. At the project level, having a clear timeline combined with project management is necessary for us to successfully complete the deliverables of Quiztown. Instead of spending time each meeting to discuss what the next steps are, we were able to move on to the next milestones quickly because we already had a clear timeline planned. It is also important for the timeline to include the marketing aspect of Quiztown on top of the technical developmental aspect. Through this, we can concurrently develop the product and get user feedback for future iterations. Because of the tight schedule of the final project, it was necessary to run the developmental and marketing works in parallel once we completed our MVP. When sticking to the planned timeline, using project management tools like Trello's Kanban board helped us manage our tasks efficiently. It also helped us to clearly allocate tasks and see our milestone progress. We were then able to make necessary adjustments to our timeline because we were able to foresee Quiztown's progress.

Working in parallel and working together is important in helping us push out many features in a short period of time. We prioritize completing any blocking tasks so that all four developers can work on new features in parallel. When any of us had something blocking us, we would inform the rest of the team in our group chat. This allowed the rest of the team to assist and unblock the person. Because of this, we increase the productivity of our team as we work in parallel and minimize the amount of time that any one of us is blocked.

Appendix

Acknowledgements

Uncle Soo for linking us up with Dr Ang from YLLSoM.

Dr Ang and Sophie from YLLSoM for getting some medical students for us to set up some user interviews in sprint 3. Through this, we were able to better reach a part of our target audience which were medical students. Subsequently, we were able to iterate on the given feedback and further polish QuizTown.

[Link] Compiled Customer Contact Reports

https://docs.google.com/document/d/1w8tVesjNrEWKdctJPBOvewqAVAO5cjui_t0nkOCnAME/edit?usp=sharing

[Link] Backend API Documentation

https://quiztown.fun/api/v1/