CS3216 Assignment 3 Final Report
Shi Jie, Si Kai, Curtis, Daphne

Quack

Milestone 0: Describe the problem that your application solves. (Not graded)

a. Problem:

While many social networks allow people to connect through name, common topics or interests, not all forms of communication have this common denominator to link people together: Imagine having to ask people around you if they wish to share a 1-for- 1 Starbucks promotion, are you going to shout across the room filled with 100 people or are you going to ask every single person in the room? Is there an easier way to connect with people around you?

b. Solution:

An app that allow the user to connect with people in his surroundings. They can now chat on a topic that is relevant to both of them, using their location as the common denominator to connect them. We achieve this by creating an application that allows user to create a chat box that is pinned to a location, and can be detected by other users in the vicinity. Hundreds of strangers can now communicate with each other in this public group chat, something that used to take a long time and a lot of effort to do so in the past.

Milestone 1: Describe your application and explain how you intend to exploit the characteristics of mobile cloud computing to achieve your application's objectives, i.e. why does it make most sense to implement your application as a mobile cloud application?

Quack is a mobile cloud application that allows the users to find chat rooms that were set up in their current vicinity. By showing users chat rooms in their immediate vicinity, Quack aims to exploit the highly mobile nature of mobile cloud application users, so as to provide a highly localised and immersive experience. Furthermore, the real time chat functionality works best with mobile application users that would most likely use the app on the go.

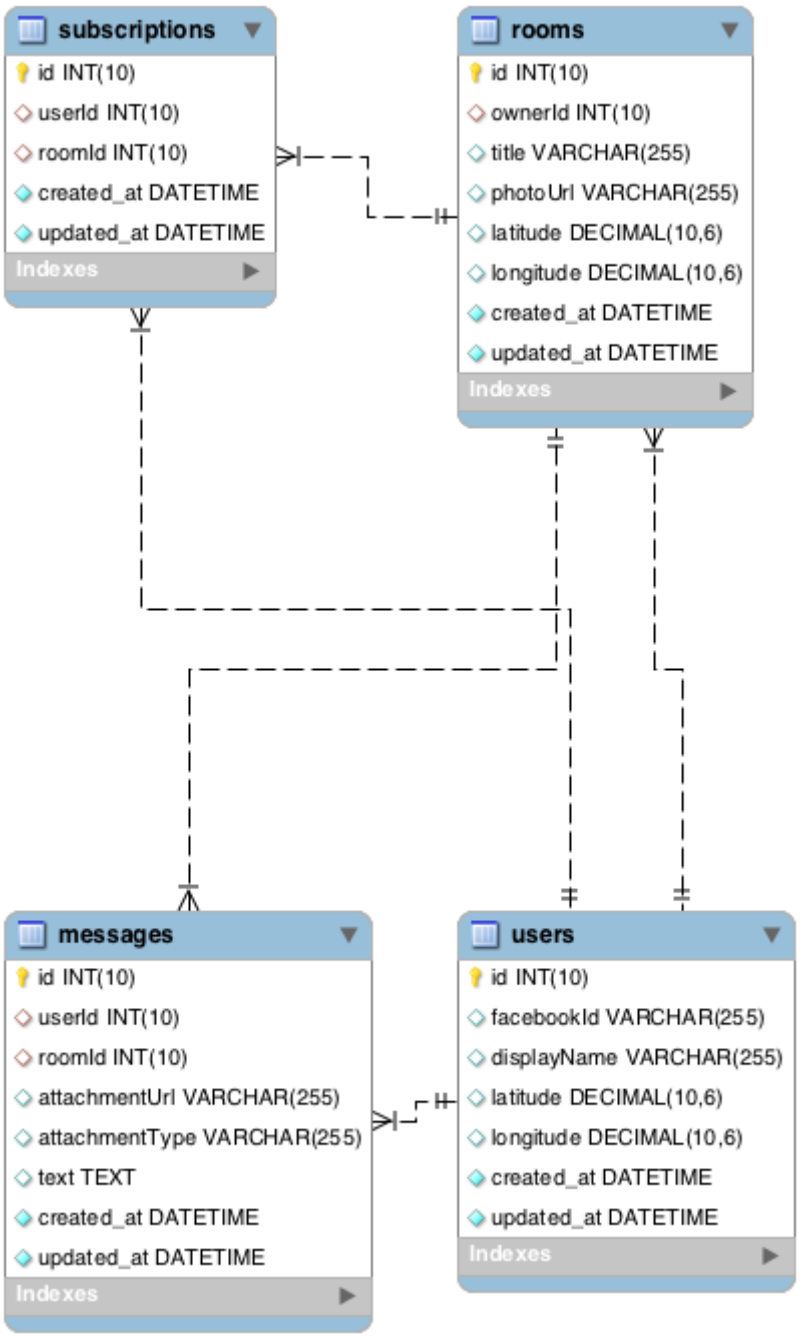
Milestone 2: Describe your target users. Explain how you plan to promote your application to attract your target users.

Note: Our application's fundamental purpose is to act as a communication tool, and hence it a means to an end for the user. This means that our users could have various types of motivation for communicating with other people. In this case, we shall use the example of sharing a Starbucks promotion as a scenario to describe how we plan to recruit our users.

Our target users for this scenario are the younger generations to middle-age who are active on the social networks, have expenditure power and yet want to save money. Hence, during Starbucks promotion, we can start a chat group for people to hook up with others who wish to share the promotion and putting up a sign outside the shop to raise awareness of the chat group. This is resource-effective and brings across the value of our application to our user through a personal and 'tangible' experience.

In addition, our application has the sharing function which our user can invite their friends from other social media platforms into the chat to allow our user base to naturally expand.

Milestone 3: Draw an Entity-Relationship diagram for your database schema.



Milestone 4: Design and document all your REST API. The documentation should describe the requests in terms of the triplet mentioned above. Do provide us with a brief explanation on the purpose of each request for reference. Also, explain how your API conforms to the REST principles and why you have chosen to ignore certain practices (if any.)

All the REST API can be found in the repository at /api-server/README.md. Instead of copying the entire REST API here, we would like to discuss about how our API conforms to the REST principles, and why we may have chosen to ignore certain practices.

Users API

Conforms to REST:

PATCH /users

Does not conform to REST:

POST /authenticate

Explanation: The authentication POST request accepts the Facebook Access Token sent by the user and test if it is active. If so, it either creates a user, or finds a user in the database with a matching Facebook Id. Finally, it constructs a JWT and returns it to the user. In this case, the POST request only conditionally creates a user resource. We choose to do this, instead of breaking it up to conform to the REST principle, so that the client need not make several round trip requests to the server.

Rooms API

Conforms to REST:

GET /rooms
GET /rooms/:roomId
POST /rooms

Explanation: We do not plan to allow users to PATCH or DELETE rooms at this point in time. As rooms are meant to be “public property” in the Quack app, we need to prevent maliciousness on the part of the room admin, who would have free reign in changing the room’s title or deleting chat history, if these APIs were permitted.

Messages API

Conforms to REST:

GET /rooms/:roomId/messages
POST /rooms/:roomId/messages

Explanation: The Messages API is scoped to a Room as the client should never need to fetch all messages that are not scoped to a room. Similarly, it should not be possible for users to post a message without defining a corresponding room. Like the Rooms API, Messages are treated as “public property”, thus the absence of a PATCH and DELETE API for messages. At this point in time, given that only text messages are permitted, that query remains fast and the response remains fast for most use cases. Thus, we do not see the need to strongly limit the number of messages fetched, but is under consideration when the need arises.

Milestone 5: Share with us some queries (at least 3) in your application that require database access. Provide the *actual SQL queries* you use (if you are using an ORM, find out the underlying query) and explain how it works.

```
SELECT *, (6371000 * acos(cos(radians(0)) * cos(radians(latitude)) * cos(radians(longitude) - radians(0)) + sin(radians(0)) * sin(radians(latitude)))) AS distance
FROM `rooms`
HAVING `distance` <= 100
ORDER BY `distance` ASC;
```

Explanation: This query calculates the distance between the user and the room and retrieve all the rooms that are less than 100m apart, using the Haversine formula to compute the great-circle distance. This is the location-based query, the highlight of our application. While this query is relatively expensive, we still chose to use this due to importance of geolocation accuracy, given the short radius provided for room discovery. In the future, if the number of rooms grow to an unsustainable level, we may consider using a less accurate, but faster trigonometric approximation.

```
SELECT *
FROM `users`
WHERE `id` = 1;
```

Explanation: This query get the user data from the user id.

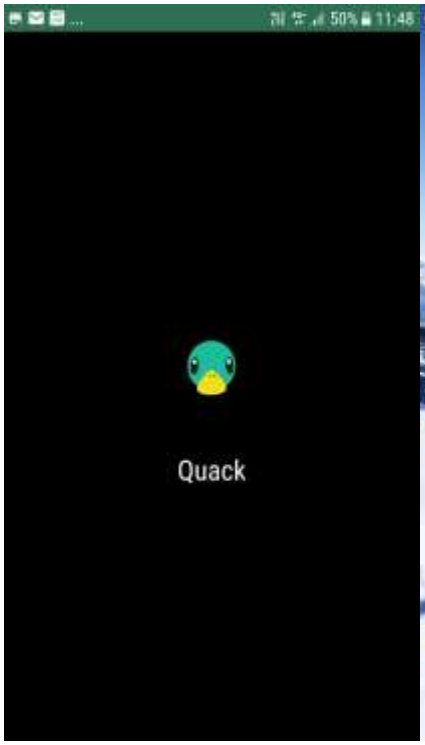
```
SELECT TOP 1 *
FROM `messages`
WHERE `roomId` = 1
ORDER BY `id` DESC;
```

Explanation: This query gets the last message of a particular room. This is useful for seeding preview information about the room in the dashboard, before the websocket takes over.

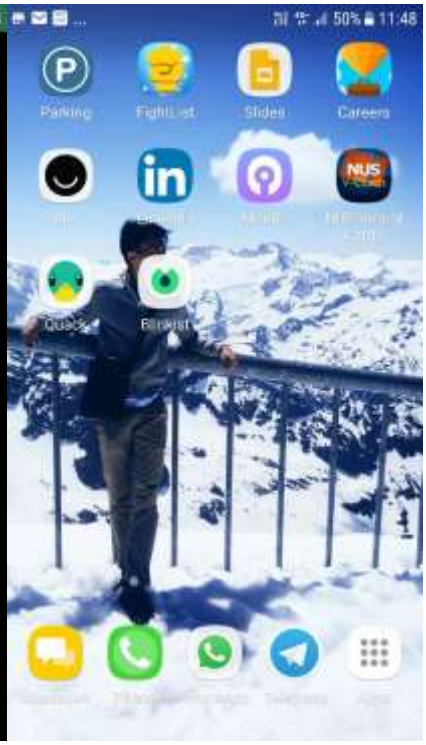
```
SELECT *
FROM `messages`
WHERE `roomId` = 1;
```

Explanation: This query gets the messages from a specific room.

Milestone 6: Create an attractive icon and splash screen for your application. Try adding your application to the home screen to make sure that they are working properly. Include an image of the icon and a screenshot of the splash screen in your writeup. If you did not implement a splash screen, justify your decision with a short paragraph. Add your application to the home screen to make sure that they are working properly. Make sure at least Safari on iOS and Chrome on Android are supported.



Splash screen



Icon

Milestone 7: Style different UI components within the application using CSS in a structured way (i.e. marks will be deducted if you submit messy code). Explain why your UI design is the best possible UI for your application. Choose one of the CSS methodologies (or others if you know of them) and implement it in your application. Justify your choice of methodology.

Since we are already using Vue.js as our front end javascript framework, we also made use of vue-loader's scoped CSS feature, which allows us to write CSS that is exclusively scoped for each Vue component. This is achieved by vue-loader running the CSS we wrote through PostCSS which rewrites our css into scoped classes by adding specific ids as attributes.

A comparison gotten from vue-loader's scoped CSS page is shown below:

Before:

```
<style scoped>
.example {
 color: red;
}
</style>

<template>
 <div class=”example”>hi</div>
</template>
```

After:

```
<style>
.example[data-v-f3f3eg9] {
 color: red;
}
</style>

<template>
 <div class=”example” data-v-f3f3eg9>hi</div>
</template>
```

We also made use of the Quasar Framework, which provided us with Mobile first UI components with first class Vue support. Quasar also provided us with many useful CSS helpers that decreased the amount of CSS we had to write.

Milestone 8: Set up HTTPS for your application, and also redirect users to the https://version if the user tries to access your site via http://. HTTPS doesn’t automatically make your end-to-end communication secure. Name 3 best practices for adopting HTTPS for your application. Explain the term "certificate pinning" and discuss the pros and cons of adopting it.

Use strong key exchange

For most public sites, choosing either the classic ephemeral Diffie-Hellman key exchange (DHE) or its elliptic curve variant, ECDHE, is generally considered optimal. Other key exchange algorithms, on the other hand, are generally insecure in one way or another. For example, the RSA key exchange, while still very popular, doesn't provide forward secrecy.

Encrypt everything

Ensuring that HTTPS is enforced across the entire site is considered best practice. Even mixed content (serving both TLS and non-TLS content) can allow an active man-in-the-middle to hijack a user session.

Use HTTP Strict Transport Security

Users that access the HTTP version of the site are vulnerable, even if the server eventually upgrades the user to use HTTPS. Using HSTS will allow the client to know that the server should always be accessed with HTTPS, and thus rewrite the request before it is even sent.

Pros and Cons of HPKP

Certificate pinning aims to allow an site or server administrator to pin some certificates, thus declaring that only these certificates are valid for some duration. The motivation behind HPKP stems from the idea that Certificate Authorities, whom have the ability to issue certificates for a website, may be compromised at some point in time, and thus issue fraudulent certificates that satisfies the certificate chain of trust.

While HPKP is great in principle, there are many issues that come with it. A site or server administrator, if he misconfigured the pins, may end up locking legitimate users out of the site for a long duration. Furthermore, with the ability to pin certificates, there are discussions about how it may be used by malicious actors to perform ransomPKP - silently pinning the site after obtaining access into it, then removing the pinned keys from the server after some time, and only reinstating it after their demands are satisfied.

Milestone 9: Implement and briefly describe the offline functionality of your application. Explain why the offline functionality of your application fits users' expectations. Implement and explain how you will keep your client synchronised with the server if your application is being used offline. Elaborate on the cases you have taken into consideration and how they will be handled.

Due to our application being a chat application that is mainly used for live interactions with people nearby while connected to the internet, we have only enabled the other auxiliary views for the user when he/she is offline. Moreover, our nearby rooms feature mandates that users should only be able to discover and chat in rooms near them.

However, the application does not break since it is cached by the browser and still functions as per normal, except that no nearby rooms will be fetched. In future iterations, we will be looking at persisting the front end store and enabling users to still look at the past messages of the rooms that they last chatted in before they went offline.
After much consideration and discussion, we believe that the main chat functionality should require an online connection from the user, after all. This is because our chat app aims to create an immersive, geolocation-based chat room community, that allows nearby users to interact in a meaningful way with each other. In order to provide that, we use websockets with fallback to long polling, to provide real time interactions between user. We believe this fits best into what the user expects when they use Quack.

With respect to offline functionality, we choose to use the offline functionality of the service workers in ways that further enhance the user's' experience in the Quack chat app. First, the service worker is used to cache the app shell and its assets. This allows users to load the page quickly without having to re-fetch all the assets on page load. Thus, users need not wait for long

page loads and server requests before they can access the chat rooms and use Quack. Additionally, the service workers know when and how to notify the user of an app shell update through an alert. This prompts the user to refresh the page to receive the latest app shell, without breaking the app even if the user chooses not to do so immediately.

In addition to caching the app shell, the Vuex store also temporarily keeps a record of the messages and rooms that the user's client has seen in its current session. Thus, the user can continue navigating around the app and see messages that it had recently seen, even with intermittent connectivity.

Milestone 10: Compare the advantages and disadvantages of token-based authentication against session-based authentication. Justify why your choice of authentication scheme is the best for your application.

Token-Based Authentication (JWT)

Advantages:

- Stateless, scalable since server doesn't have to store the tokens
- Less susceptible to client-side manipulation since they are signed on the server side
- Don't have to meddle with CORS and can even pass it to downstream services

Disadvantages:

- Increased overhead on HTTP Requests as more information is stored in JWT

Session-Based Authentication (Cookies)

Advantages:

- Low overhead on HTTP requests as it is usually just and Id

Disadvantages:

- Stateful, which means server needs to store all active sessions somewhere.
- Susceptible to CSRF, XSS attacks
- Can only be used on the domain that it is generated from

Our Decision:

We chose to use token-based authentication as it is more scalable for the backend server and allows us the flexibility to add downstream services that might require user identification in the future.

Milestone 11: Justify your choice of framework/library by comparing it against others. Explain why the one you have chosen best fulfils your needs. Lastly, list down some (at least 5) of the mobile site design principles and which pages/screens demonstrate them.

Front end frameworks:

- Vue.js (Chosen)
  We chose Vue.js as Si Kai and Curtis have read through the docs and felt that it was an ecosystem that is well-documented and not fragmented. Si Kai has had a good experience with Vue.js in the previous assignment and decided that it would allow the team to build out a functional and scalable UI relatively quickly. Since the core team also manages their router and state management solution, the landscape of Vue.js did not appear to be as fragmented as the well-known big boy React.js. While Curtis has experience with Angular, he had the flexibility of doing both frontend and backend.
- Quasar Framework (Chosen)
  Looking around for a mobile UI framework, Si Kai ran through Vuetify, Onsen UI and Framework 7 and finally Quasar Framework. We finally decided to go with Quasar as it had first-class support for Vue.js while we couldn't even get a hello world with Framework 7's so called vue integration library. (The latest Framework7 + Vue is broken). Quasar also had better documentation, which helped the team tremendously, since it was the first time everyone was building a product with the mobile interface as priority.
- Angular (Not chosen)
  While Angular was a decent framework to use, we believe that the learning curve for Angular would be much higher than that for Vue, as it was a "heavier" framework. Furthermore, most of the developers were more comfortable with Javascript than Typescript. While static type checking is useful for many project, the short timeline for assignment 3 implies that the pros of a static type checker may not justify the cons of having to learn and develop in Typescript.
  React (Not chosen)
  While somewhat subjective, the recent discussion about React's (as well as many of Facebook's open-source libraries and frameworks) BSD + Patent trojan horse strategy was not to our liking. Given the current fragmentation of the community, plus the expectation of further fragmentation over this issue, we believe that React may cause problems down the road, even if not in the immediate future - especially if we want to take this application beyond the scope of this assignment.

Mobile site design principles:

We have learnt quite a bit about mobile site design from the Quasar documentation as well as while using their UI components. Here are five examples that we would like to share as we think they greatly improve usability.

- Pull to refresh
  In the main view of Quack where a user will be able to find nearby rooms, we have made use of the Pull to refresh component from Quasar to enable them to use the familiar pull to refresh gesture to refresh the list when they have changed their location.

- Side nav drag out from left on mobile
  On android phones and the desktop version of Quack, you will be able to open the side nav from the main nearby rooms view in two ways. Either by hitting the hamburger button on top, or by sliding your finger from the left corner of the screen towards the right. This is to mimic the side panels that mobile applications usually have and to allow the user to have a smoother mobile experience.

- Wider scroll bar in chat
  Thanks to Quasar Framework, there will be a wider scroll bar in the chat view for certain mobile browsers. This is to allow users to use their thumb to navigate through a chat room where there are too many chat messages.

- Ripple
  We have added the Ripple effect to the Facebook Login button on the landing page to give the user a more responsive experience as touching the button will send a ripple throughout the button. The center of the ripple starts from the point of contact.

- Side nav collapse in mobile mode
  When looking at Quack in desktop mode, you will realize that the side nav is open. As you resize the browser to a smaller screen and eventually to that of a mobile size, you will see that the side nav is collapsed to allow the user to make the most out of the small screen when enjoying the Quack application.

Milestone 12: Describe 3 common workflows within your application. Explain why those workflows were chosen over alternatives with regards to improving the user's overall experience with your application.

Workflow 1: Inviting friends to our application

We allow the users to invite friends not just to download our app but to invite them to a particular chat. The reason is that users wishes to share the content of the application (more interesting) and not the application itself. Compared to sharing the application, sharing the content of the chat with their friends helps them arrive at their objective (getting their friends to see the ongoing conversation) faster. As a result, this workflow gives the user more control and efficiency, hence better user experience.

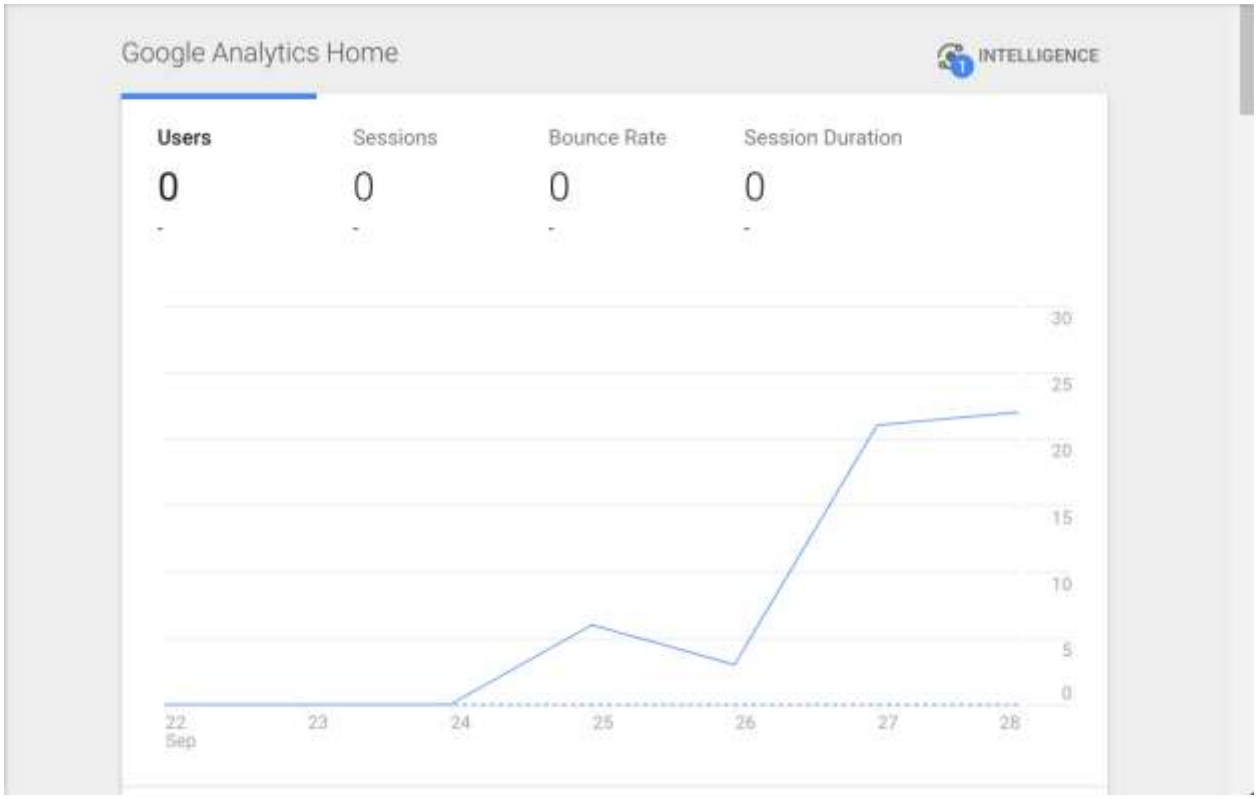
Workflow 2: Anonymity control

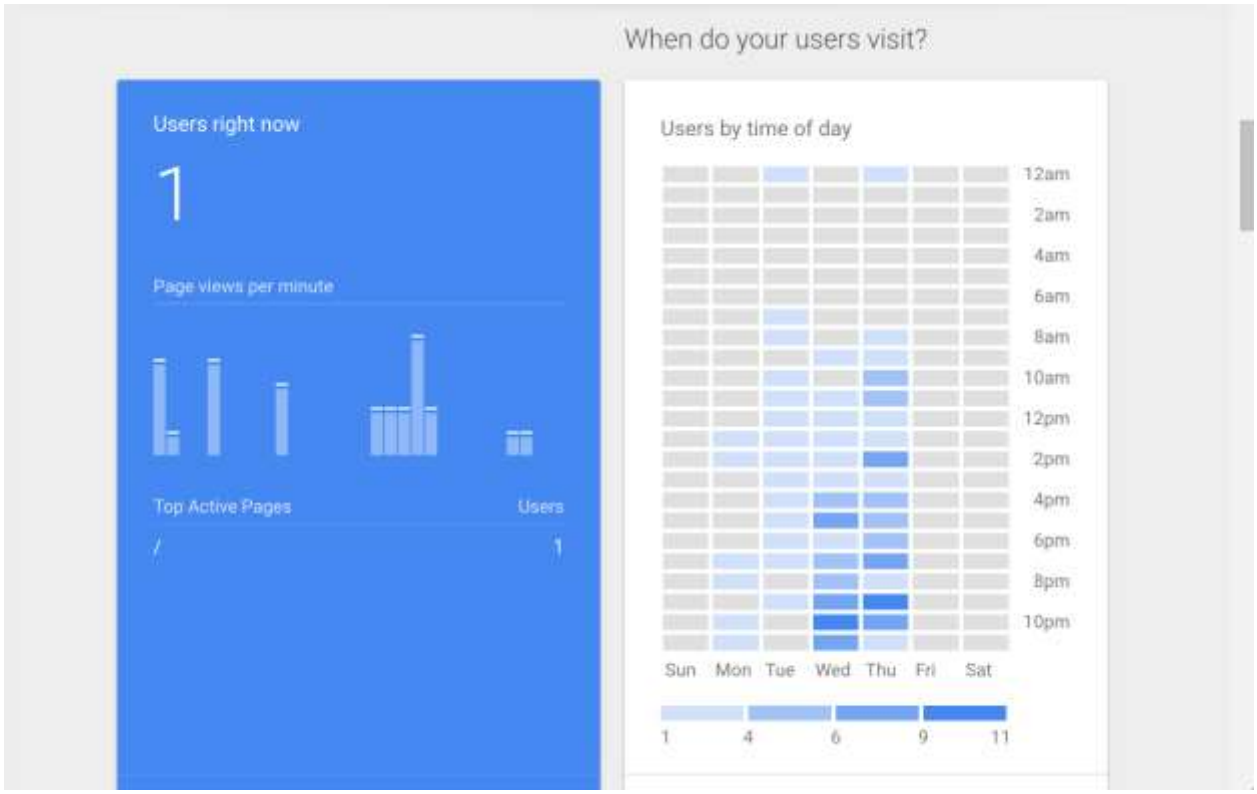
As our users strongly value the anonymity of his account, we allow them to have full control and easily change their userID in the menu page, rather than burying it deep in the menu function which might cause users to think that we are trying to prevent them from going anonymous. Having full control over their ID allows users to freely express themselves without fear of judgement, while at the same time still ensures that they will act responsibly since we will still be able to track and ban the user if they act otherwise.

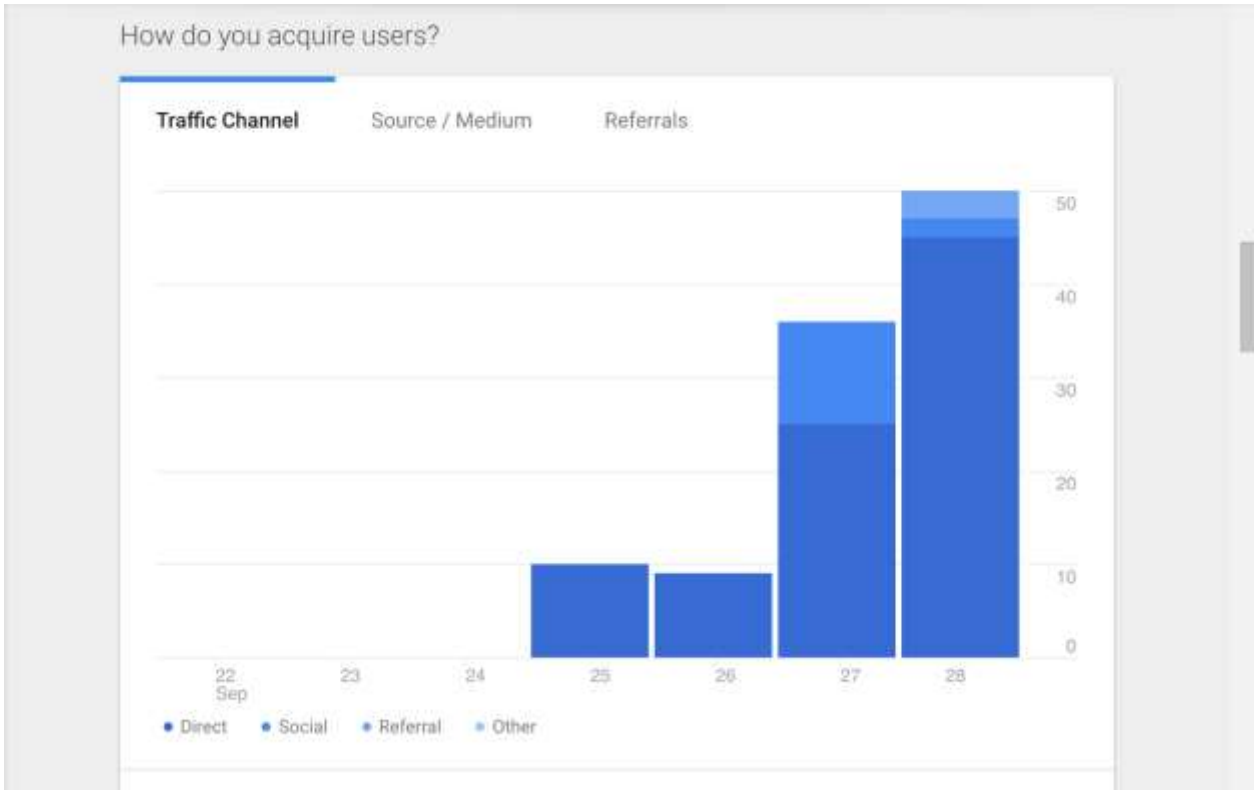
Workflow 3: Creating new chat

To improve the user experience, the homepage cannot be too cluttered and the chat groups must be clearly defined. As such, between allowing users to freely title their chat groups, we impose a character limit to ensure that they succinctly name their chat group. Should we not implement the character limit, the chat groups would have titles that are too long and cut off before other users are able to understand what the chat group is about, causing user confusion and frustration.

Milestone 13: Embed Google Analytics in your application and give us a screenshot of the report. Make sure you embed the tracker at least 48 hours before submission deadline as updates are reported once per day.




When do your users visit?
Users right now
1
Page views per minute
Top Active Pages
Users
/
1
Users by time of day
12am
2am
4am
6am
8am
10am
12pm
2pm
4pm
6pm
8pm
10pm
Sun
Mon
Tue
Wed
Thu
Fri
Sat
1
4
6
9
11

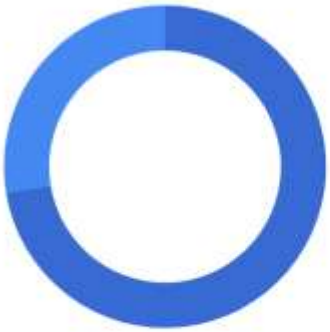

How do you acquire users?
Traffic Channel
Source / Medium
Referrals
50
40
30
20
10
0
22
Sep
23
24
25
26
27
28
Direct
Social
Referral
Other

Where are your users?

Sessions by country

What are your top devices?

Sessions by device



Desktop
72.4%

Mobile
27.6%

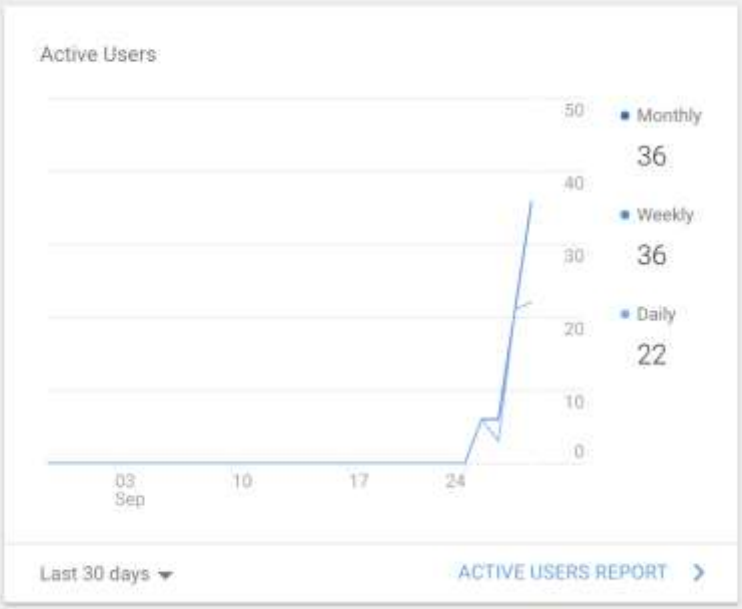
What pages do your users visit?

Page	Pageviews	Page Value
/	951	$0.00
/login	446	$0.00
/chat/1	287	$0.00
/chat/20	159	$0.00
/chat	105	$0.00
/chat/share	80	$0.00
/chat/21	67	$0.00
/chat/2	65	$0.00
/chat/22	58	$0.00
/chat/1/share	30	$0.00

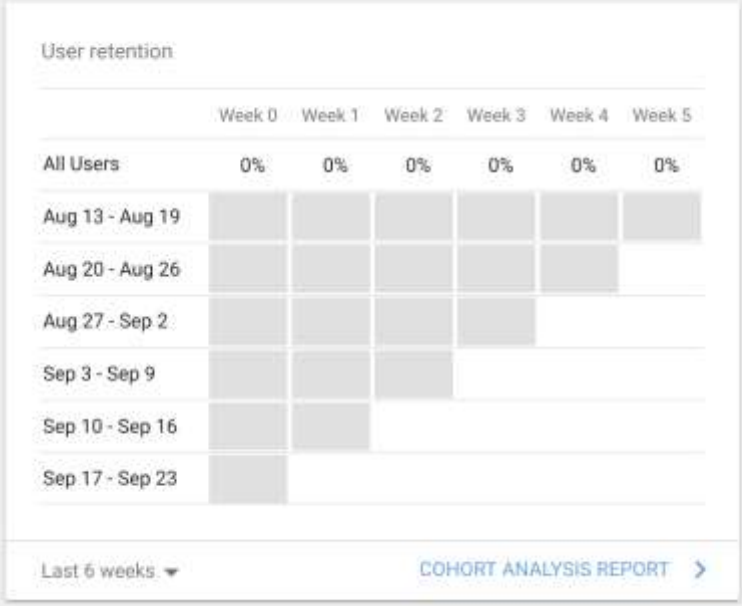
Last 7 days

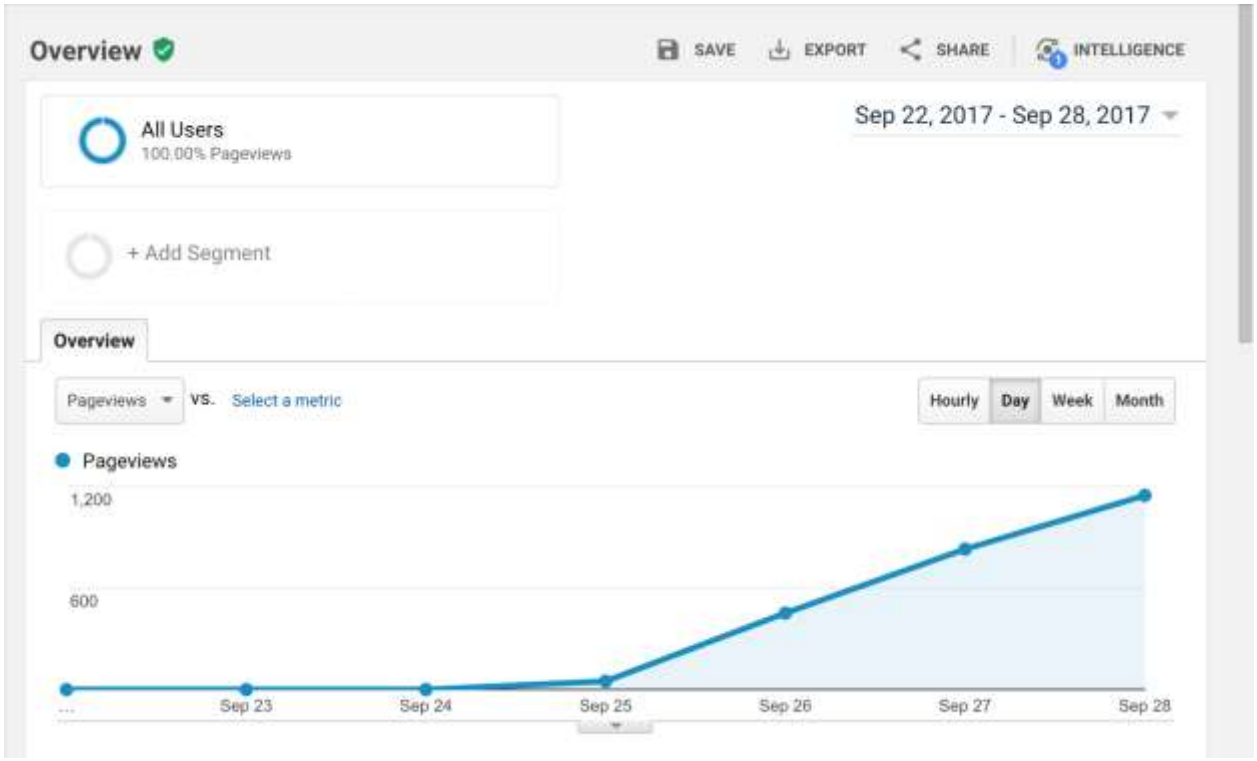
PAGES REPORT

How are your active users trending over time?

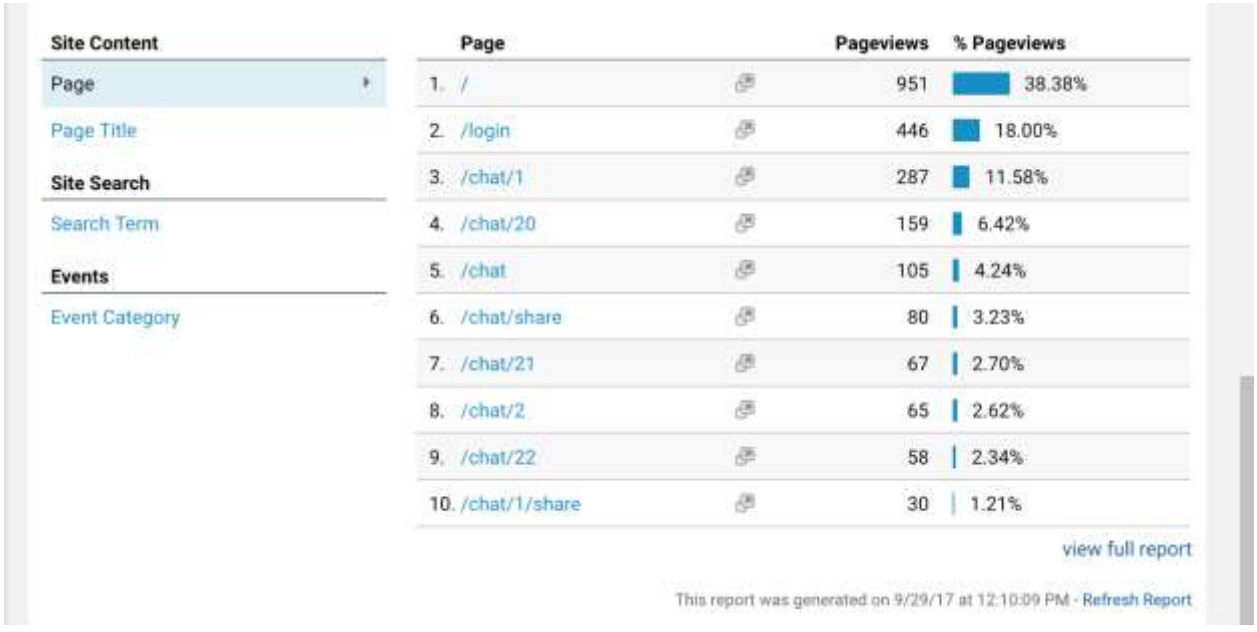


How well do you retain users?




Overview
SAVE
EXPORT
SHARE
INTELLIGENCE
All Users
100.00% Pageviews
Sep 22, 2017 - Sep 28, 2017
+ Add Segment
Overview
Pageviews
vs.
Select a metric
Hourly
Day
Week
Month
Pageviews
1,200
600
Sep 23
Sep 24
Sep 25
Sep 26
Sep 27
Sep 28


Pageviews
2,478
Unique Pageviews
277
Avg. Time on Page
00:01:14
Bounce Rate
11.43%
% Exit
4.24%


Site Content
Page
Page Title
Site Search
Search Term
Events
Event Category
Page
Pageviews
% Pageviews
1. / 951 38.38%
2. /login 446 18.00%
3. /chat/1 287 11.58%
4. /chat/20 159 6.42%
5. /chat 105 4.24%
6. /chat/share 80 3.23%
7. /chat/21 67 2.70%
8. /chat/2 65 2.62%
9. /chat/22 58 2.34%
10. /chat/1/share 30 1.21%
view full report
This report was generated on 9/29/17 at 12:10:09 PM - Refresh Report

Page	Pageviews	Unique Pageviews	Avg. Time on Page	Entrances	Bounce Rate	% Exit
	2,478 % of Total: 100.00% (2,478)	277 % of Total: 100.00% (277)	00:01:14 Avg for View: 00:01:14 (0.00%)	105 % of Total: 100.00% (105)	11.43% Avg for View: 11.43% (0.00%)	4.24% Avg for View: 4.24% (0.00%)
1. /	951 (38.38%)	81 (29.24%)	00:01:28	48 (45.71%)	16.67%	5.57%
2. /login	446 (18.00%)	65 (23.47%)	00:00:39	38 (36.19%)	2.63%	6.05%
3. /chat/1	287 (11.58%)	26 (9.39%)	00:01:15	8 (7.62%)	0.00%	3.14%
4. /chat/20	159 (6.42%)	5 (1.81%)	00:01:36	1 (0.95%)	0.00%	1.26%
5. /chat	105 (4.24%)	17 (6.14%)	00:01:37	4 (3.81%)	25.00%	2.86%
6. /chat/share	80 (3.23%)	1 (0.36%)	00:01:39	0 (0.00%)	0.00%	0.00%
7. /chat/21	67 (2.70%)	5 (1.81%)	00:01:14	0 (0.00%)	0.00%	0.00%
8. /chat/2	65 (2.62%)	7 (2.53%)	00:01:50	0 (0.00%)	0.00%	3.08%
9. /chat/22	58 (2.34%)	4 (1.44%)	00:00:56	0 (0.00%)	0.00%	0.00%
10. /chat/1/share	30 (1.21%)	1 (0.36%)	00:00:12	0 (0.00%)	0.00%	0.00%

Milestone 14: Achieve a score of at least 90 for all categories and include the Lighthouse html report in your repository.

Report is attached in repository as 'www.quack.press_2017-09-29_22-48-17.html'

Milestone 15: Identify and integrate with social network(s) containing users in your target audience. State the social plugins you have used. Explain your choice of social network(s) and plugins. (Optional)

The social network we are targeting is Facebook. Facebook has over 2 billion monthly active users. To login with Facebook means that we do not have to store the user's username and password. To the users, this means that they do not have to remember another set of username / password. In addition, we can easily obtain the user's name (which they can change subsequently).

For the social plugins, we encourage users to share their chat with their friends to expand user base. We choose Facebook, Google +, Twitter and Weibo as the primary base to share posts. They are common social networking platforms and each of them have a significant number of user base. If user use our application as a web, they can copy the chat url and share with their friends in any social networking platforms.

Milestone 16: Make use of the Geolocation API in your application. (Optional)

Our application is centered around the geolocation API since each room is given a latitude and longitude value based where it was created. The main screen shows rooms that are within a 100m radius from the user's current location.

Other than using the browser's Geolocation API, we also used the Haversine formula in the backend to evaluate which rooms are considered nearby.