

Hitt

Lokaskýrsla

13. Maí

Höfundar:

Davíð Fannar Ragnarsson

Matthías Finnur Vignisson

Stefán Örn Ómarsson

Leiðbeinandi:

Ásgeir Freyr Kristinsson

Prófdómari:

Þorgeir Ómarsson

Table of contents

1. Introduction	4
1.1. Why make a location based social app?	4
1.2. Why the name Hitta?	4
2. What is Hitta?	5
3. Technical description	6
4. Navigation of components	7
5. Database	8
5.1 Firestore	8
5.2 Cloud storage	10
5.3 Authentication	11
5.4 Transactions	11
6. Work agreement	12
7. Timetable status	13
8. Charts and diagrams	16
8.1. Time spent in hours	16
8.2. Time spent in hours on each task	17
8.3. Class diagram	18
8.4. Flow chart	19
9. Risk analysis	20
10. Design	23
10.1. User testing	23
10.2. Interview script	23
10.3. Result from interviews	25
10.3.1. Informal testing	25
10.3.2. Formal testing 1	25
10.3.3. Formal testing 2	26
10.4. Visual design	27
10.5 Some images from the in app UI.	28
10.6. The evolution of the UI	29
11. User groups	30
11.1. User groups	30
11.2. User groups table	30

12. User requirements	32
13. User stories	33
13.1. User	33
13.2. Host	33
14. Use cases	34
15. System requirements	40
16. System tests	40
16.1. Transactions test	41
16.2. Database limit test 1	41
16.3. Database limit test 2	42
16.4. Database limit test 3	42
16.5. Thoughts about scalability for the future	43
17. Progress summary	43
18. Sprint plan	45
19. The future of Hitta	57
20. Final words	58

1. Introduction

This report is about our work in the final course of our B.Sc. degree in computer science. We will go in detail over our progress, design, setup and other aspects in regards to our project. The team consists of 3 members; Davíð Fannar Ragnarsson, Matthías Finnur Vignisson and Stefán Örn Ómarsson. At the beginning of the semester there were several companies that offered projects for students to do in cooperation with them. Davíð and Matthías wanted to do something new from scratch and therefore decided on developing Hitta. We then recruited Stefán and in the following months developed our vision for Hitta.

1.1. Why make a location based social app?

We were really excited about the idea from the start when Davíð pitched it. There are not a lot of location based social apps situated in Iceland so we saw potential for it in the market. We've heard about some location based applications for music events for example, but no app that offers as much variety and interaction as Hitta does. Likewise we've heard of other students that have already tried to develop similar applications. In the end we decided to take on the challenge and make the best location based social app in our given time frame.

1.2. Why the name Hitta?

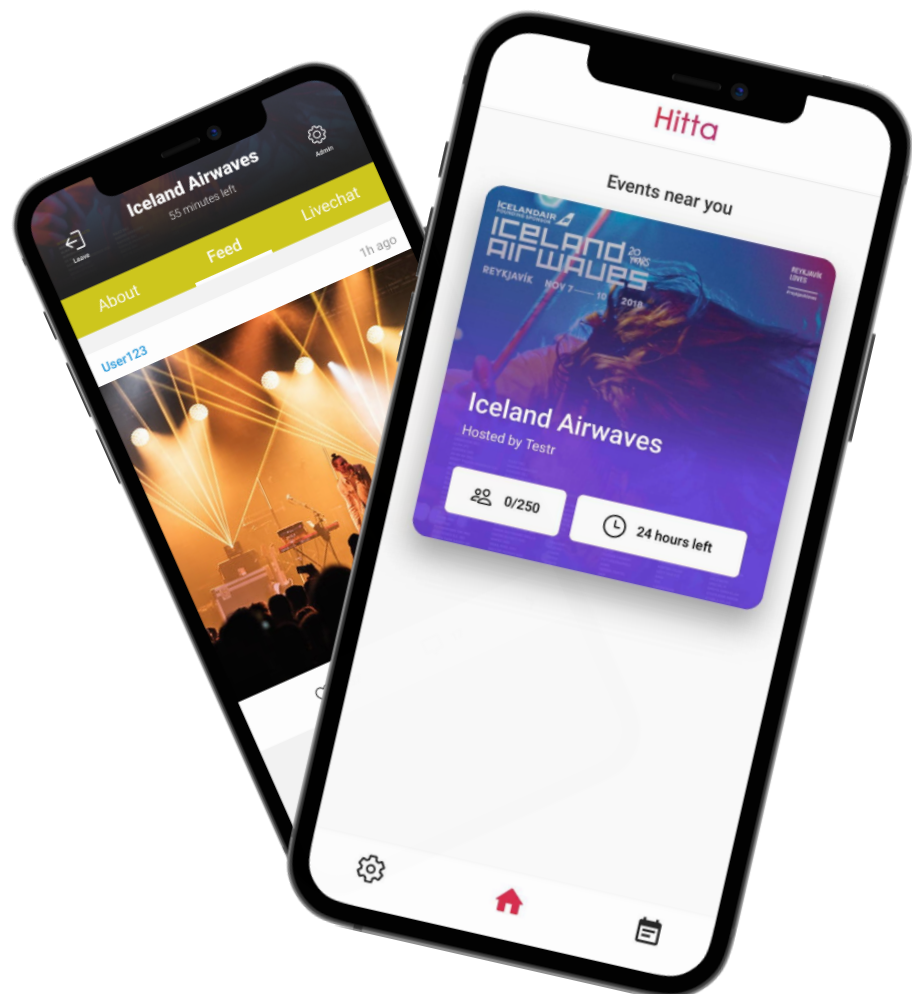
We came up with the name Hitta early on, which simply means "meet" in Icelandic. We think it's a simple and short name which also manages to describe what the app is about. If we plan to take the app to foreign countries we believe that the simplicity of the name can make it more memorable and more likely to catch on.

2. What is Hitta?

What can you do with Hitta, how does it work for the average user

Hitta is an app that connects people that share one thing in common, their location. Hitta creates an ecosystem at a specific location, allowing users to interact in a closed group which is exclusive to their location and reason for being there. Hitta's goal is to elevate the experience of events and bring the guests closer together by allowing users to communicate and share experiences with people they normally would not be able to. We believe that in the future, big events will look at Hitta as a necessity to ensure a great experience for their guests.

Why is Hitta important for events? Imagine you are camping with your friends, but you forgot to bring a pump for your air mattress. Of Course, you can walk across the camping site asking people for a pump but Hitta allows you to connect to all those people instantly. So, you open up Hitta and see that there is an event for this campsite. You join it and post a status asking if anyone can lend you a pump. You get a comment from another user telling you that he can lend it to you. You can now meet up with that other guest and thanks to Hitta, you will not be sleeping on the ground tonight!



3. Technical description

How does Hitta work on a technical level, here we dive a little deeper into things

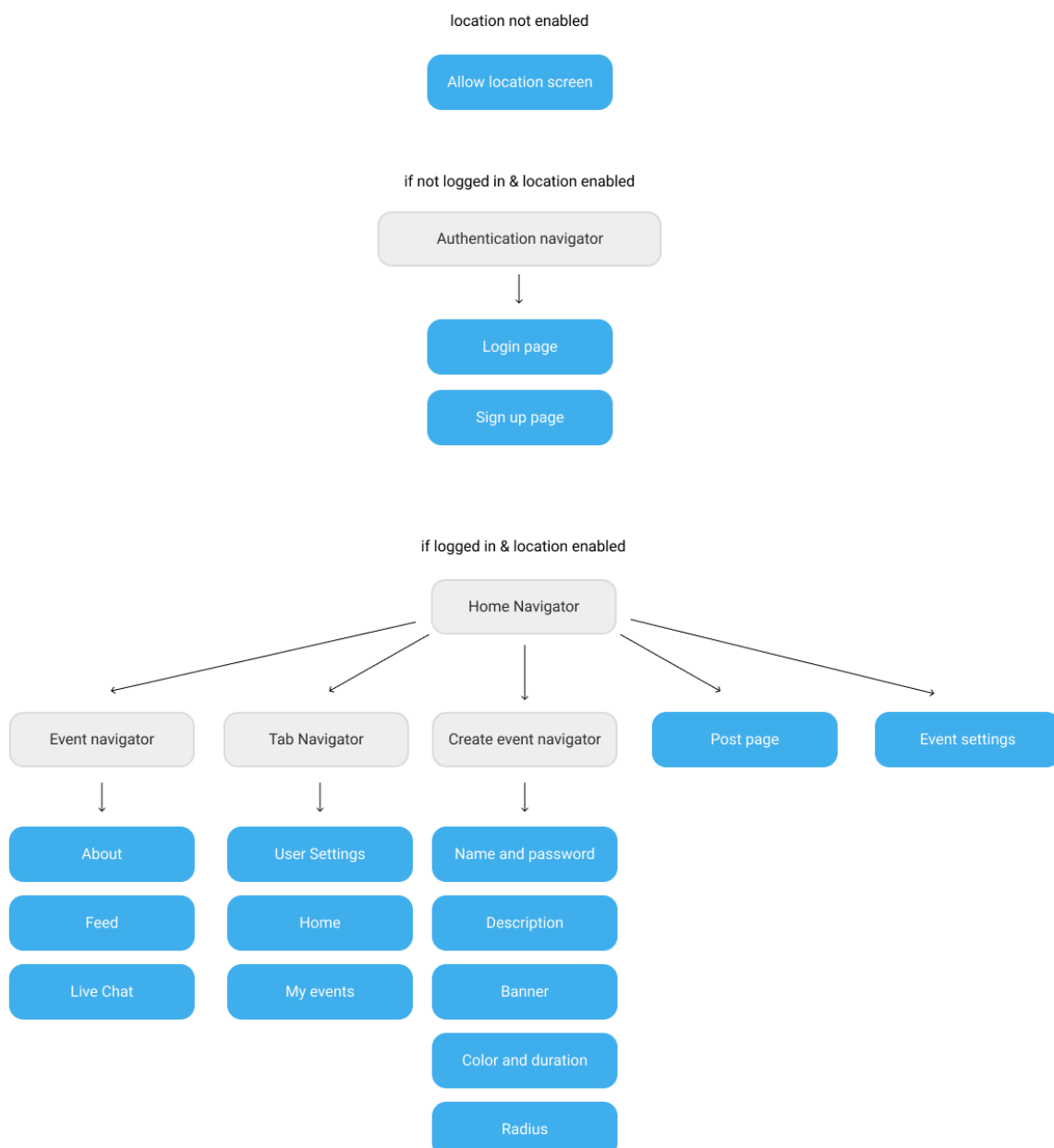
Users can either create or join an event. Events are created in specific locations where only users that are in range can join them. A user then becomes a guest of that specific event. Inside the event, a guest can post a photo, status, comment on posts or chat in a livechat which everyone can see and take part of. When a guest leaves the event location, he gets removed from the event and can no longer see what takes place inside it.

Besides joining groups, users can also create their own events. The creator then becomes the host of that event and can modify the event settings and moderate it (ban/unban users). The event will be created at the host's current location where he can also specify the radius of it. The lifetime of events can vary, but when that time runs out, every information and data in it will be deleted. Hosts can specify this lifetime when they create the event among other settings.

Events can be made public or private, when a user is in the events range they will be displayed to him as either private or public. Private events require a password set by the host. We've implemented restraints on event creation regarding the size of events and the lifetime. The size of the event dictates how many people can join it.

4. Navigation of components

The picture below describes how different components and navigators interact with each other, where navigators are colored grey and pages blue. All components in Hitta are functional components. In order to use the application in general the user has to allow location tracking, since the application is location based.



5. Database

Hitta uses Firebase to store all user generated data. Firebase offers a wide range of services, e.g. Firestore, Cloud storage and Authentication. We use all three of these services to make Hitta happen.

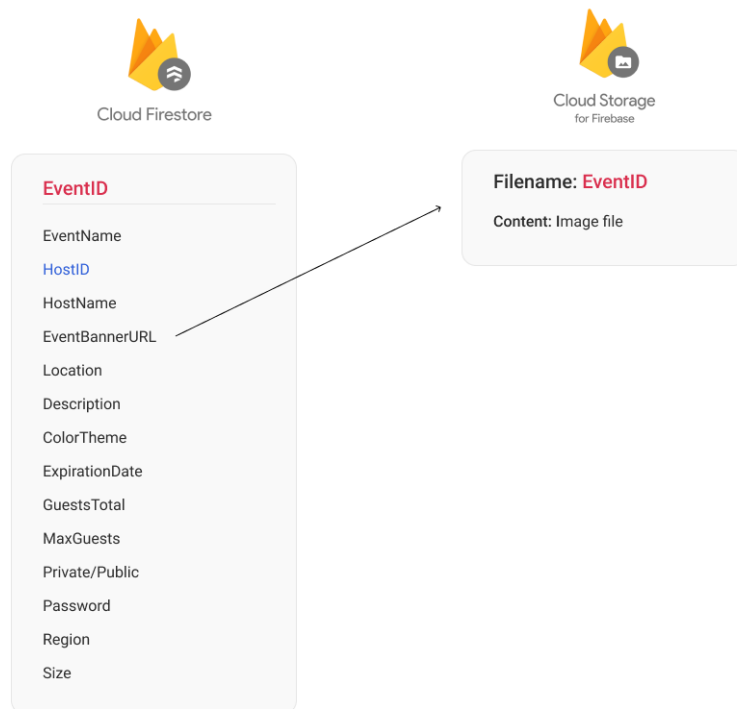
5.1 Firestore

Firestore is a flexible and scalable cloud database that allows us to listen to real time data updates. It stores conventional data like strings, numbers, arrays etc. In our case, we store most of our data there. Because Firestore offers real time data updates, development of the live chat functionality was fairly easy. We however try to minimize listening to real time updates across the app to reduce bandwidth for our users.

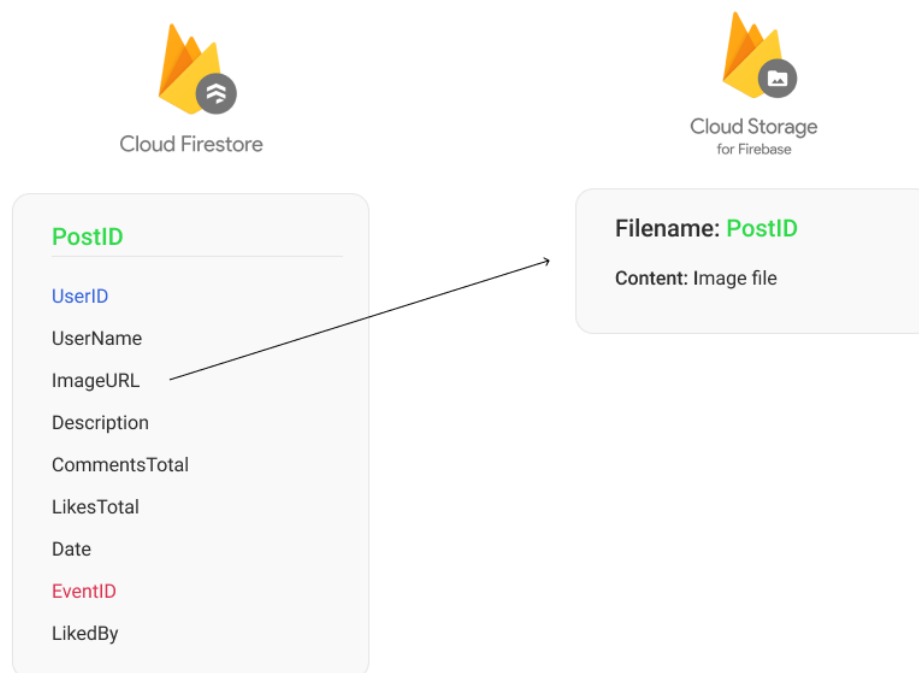
Firestore separates data into collections. These collections can be thought of as arrays of documents. Our database is split into 7 collections:

- Events
- Users
- Posts
- Comments
- Replies
- Reports
- Livechat

We can look at a few examples of how this data is stored. The following picture shows an event document which is part of the events collection mentioned above. The event is identified by a unique identification number and stores all its variables in the document except for the banner image. It instead stores a link to a banner image, while the image itself is stored in the cloud storage.



The following picture shows how posts are stored in the database. A user can post a status in an event with or without an image. Similar to events, the post holds a link to the image, while the image itself is stored in the cloud storage. A notable feature to the post structure is the “likedBy” field, which stores the identification numbers of all users which have liked the post. This makes sure that each user can only like a post once.

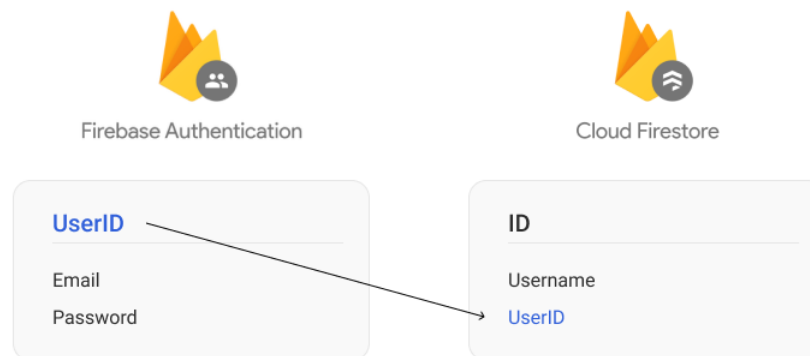


5.2 Cloud storage

Cloud storage stores all user generated images, e.g. banners for events or images for posts. These images are uploaded to the storage and like mentioned above, their link is stored as a variable for either post or event.

5.3 Authentication

Firebase offers an authentication service to authenticate users. When using Hitta, users need to register with an email, username and password. Firebase authentication takes care of storing the email and password in a separate and more secure database. On the contrary, usernames are stored in the regular firestore under the collection named users.



5.4 Transactions

Firestore offers the functionality of sending transactions to the database. We use transactions to increase or decrease numbers in the database, e.g. total likes, total comments or current number of guests in an event. Let's take an example, if a user likes a post, the like counter should increase by one.

If we would use a regular query to get the data, increase the like counter by one and then send the updated data to the database, another user could have done the same, at the exact same time. This would lead to the like counter only increasing by one like, since they fetched the data at the same time and both increased the same number by one.

A transaction on the other will try to update the like counter only if the data has not changed since it got it in the first place. If the like counter changes in the meantime, the transaction will fail and try again infinitely until it succeeds. This makes sure that the like counter will eventually show the real like count.

6. Work agreement

Explanation on how we've planned this semester working on this project Hitta as a team

We have a rough plan of our sprints and our progress will be documented in Google docs. Sprint meetings are held twice a week on Discord, Thursdays and Sundays. Each sprint is 7 days long and this format has been working passably. We needed to move some of our workload to the last three weeks since other classes interfered with our work. We estimate 13.75 hours of work in each sprint for every member. This number will increase in the last 3 sprints and we estimate that 50% of the application will be developed in that period.

The application will be developed in the React native / Expo framework and firebase will be used for data storing and networking. Code is stored on Github and we have a strong requirement for our members to only push readable and well written code.

The end result is a working iOS app that runs inside expo with an active database. Since app deployment to the App store can take some time, a working app using expo and firebase will be sufficient for this project.

7. Timetable status

Shows how much each individual has been working in hours throughout the semester

On May 13th, currently in sprint 14.

Date	Stefán	Matti	Davíð
1.19.2021	2	3.5	3.5
1.21.2021		3	3
1.22.2021	2.5	2.5	2.5
2.2.2021	3.5	5.5	5.5
2.3.2021	5		2.5
2.7.2021	3	3.5	3.5
2.18.2021	4	3	6
2.19.2021	5	4	6
2.21.2021	5	5	5
2.22.2021		1	7
2.27.2021	1		2
2.28.2021		2.5	4.5
3.1.2021	3	3	6.5
3.2.2021		5.5	
3.3.2021	2.5	7.5	7
3.4.2021		6	6.5
3.10.2021	1	1	2.5
3.14.2021	2	2	3

3.15.2021			7
3.16.2021	3	3	5.5
3.19.2021	2	4.5	10
3.20.2021	3	3	9
3.21.2021	3.5	3.5	7
3.22.2021	3	0.5	6.5
3.23.2021	4	4	4
3.24.2021	2		3
3.26.2021			2
3.29.2021	9	1	
3.30.2021	3.5		3
3.31.2021	7	3	4
4.2.2021	3	1	
4.3.2021	6		3
4.6.2021	3	7.5	2
4.7.2021	6	8	1
4.20.2021	3	5.5	3.5
4.21.2021	3	7	7
4.22.2021	4	8	10
4.23.2021	6	7.5	11
4.24.2021	2		4.5
4.25.2021	6.5	4.5	5.5
4.26.2021	10.5		7.5

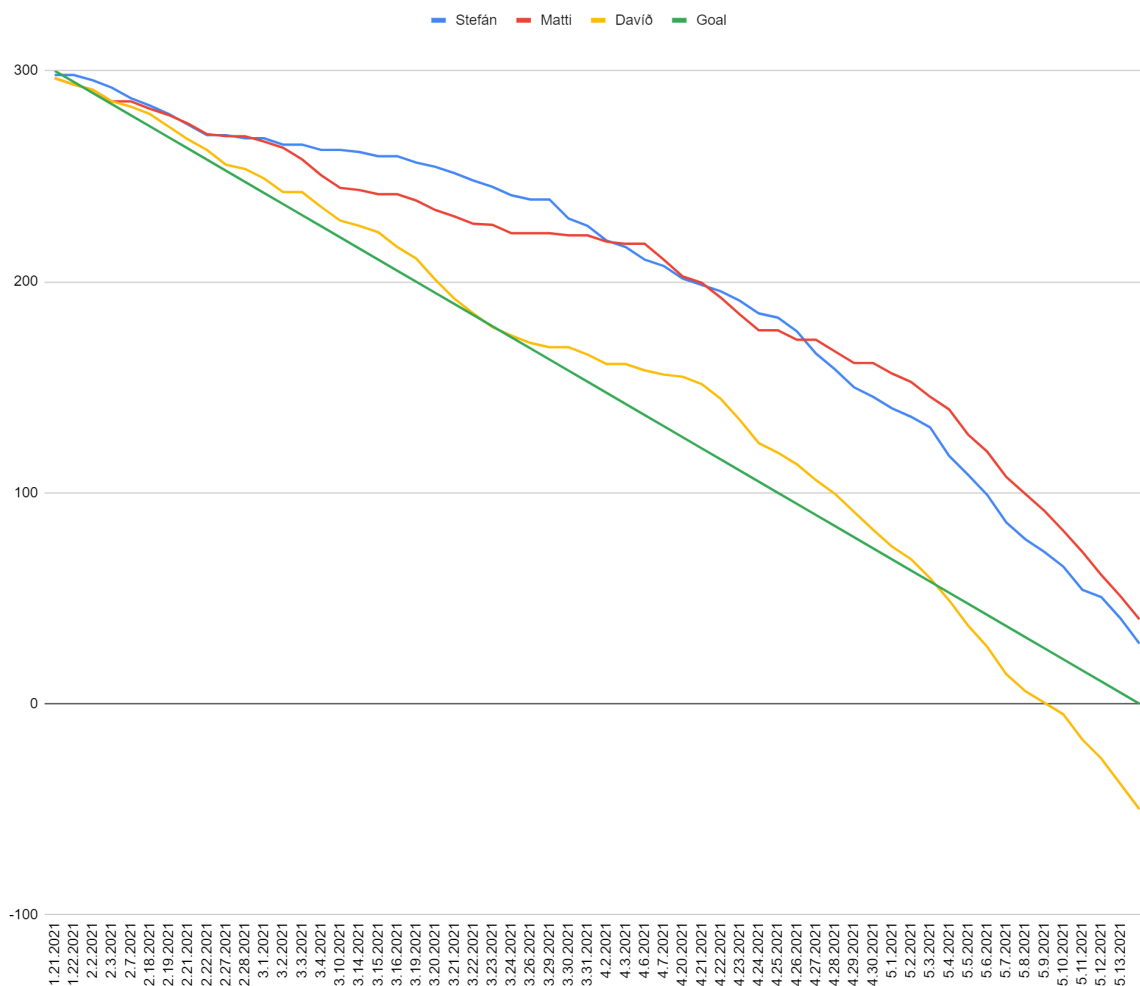
4.27.2021	7.5	5.5	6.5
4.28.2021	8.5	5	8.5
4.29.2021	4.5		8.5
4.30.2021	5.5	5	8
5.1.2021	4	4	6
5.2.2021	3	5	7
5.3.2021	13.5	6	10.5
5.4.2021	9	12	12
5.5.2021	9.5	8	10
5.6.2021	13	12	13
5.7.2021	9.5	8	8
5.8.2021	8	8	5.5
5.9.2021	7	9.5	5.5
5.10.2021	11	10	12
5.11.2021	3.5	11	9
5.12.2021	10	10	12
5.13.2021	12	11	12
Total	271.5	260	350

8. Charts and diagrams

Different chart and diagrams are displayed here that show both how we've worked throughout the semester in regards to tasks and sprints and also how our system (e.g. database and flow) is set up

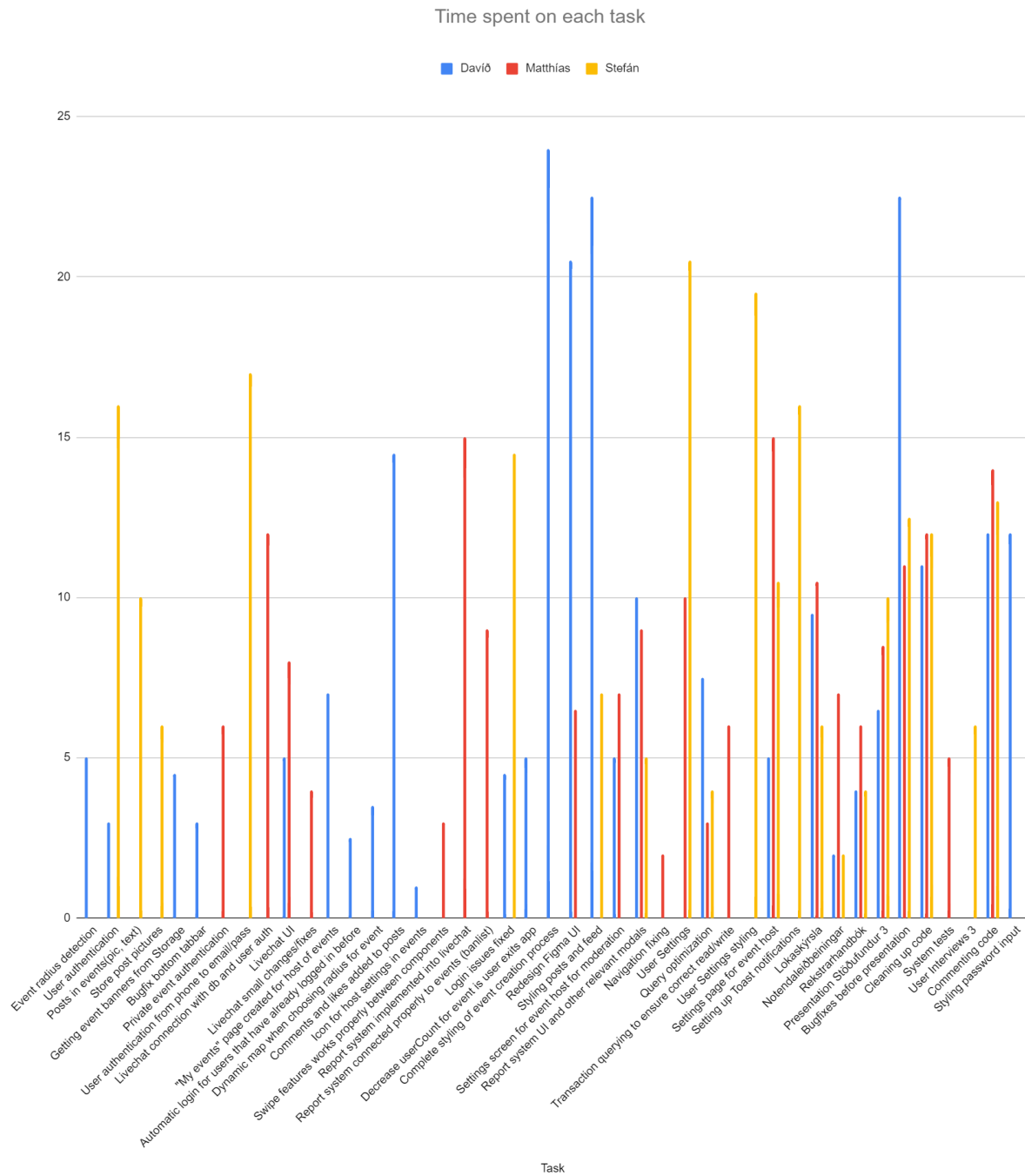
8.1. Time spent in hours

Represent how much time we've spent in hours throughout the semester compared to the actual goal in hours



8.2. Time spent in hours on each task

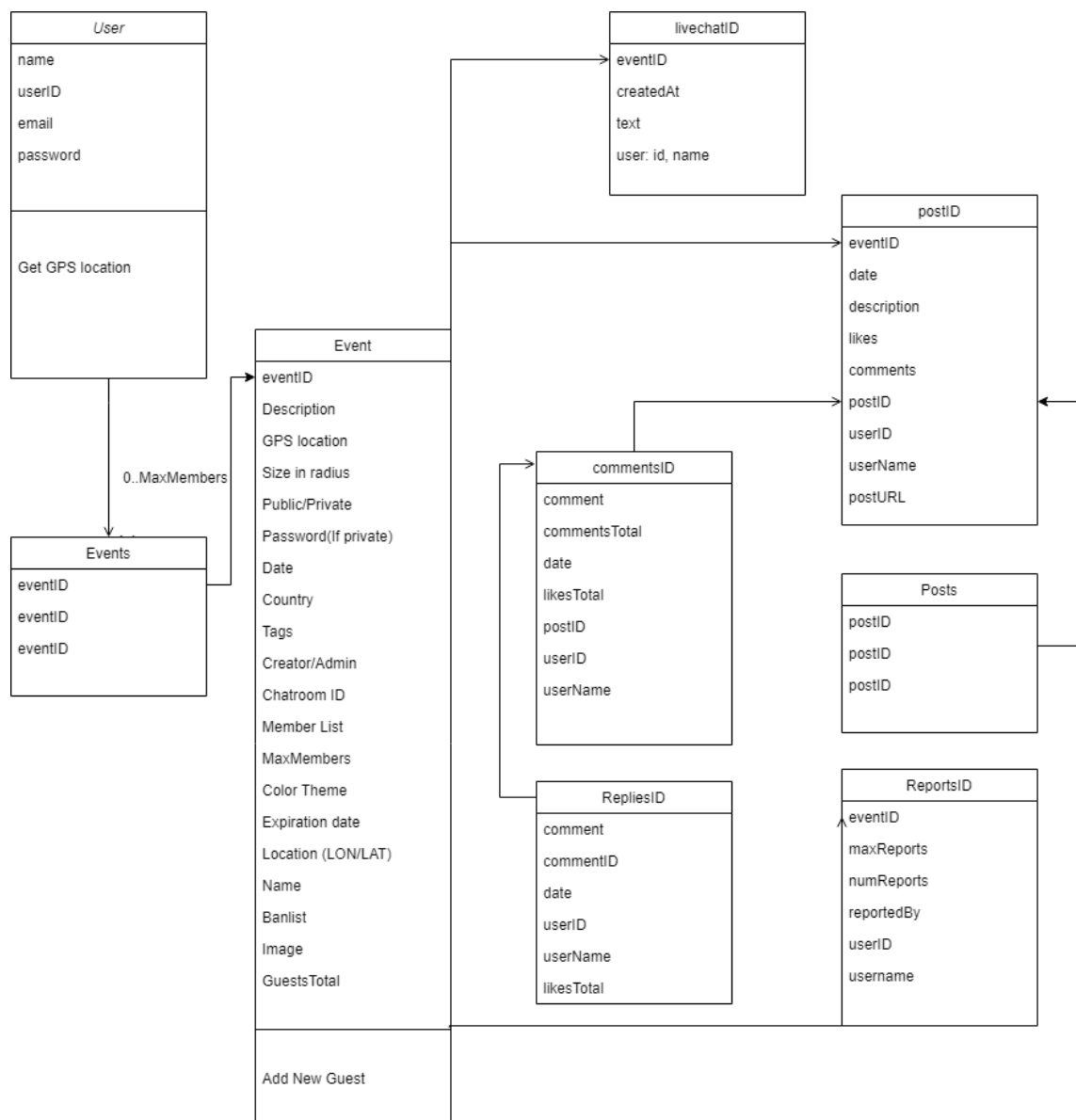
How much time has each team member spent on each task since Stöðufundur 2



8.3. Class diagram

Describes the structure of our database and relationship between different objects

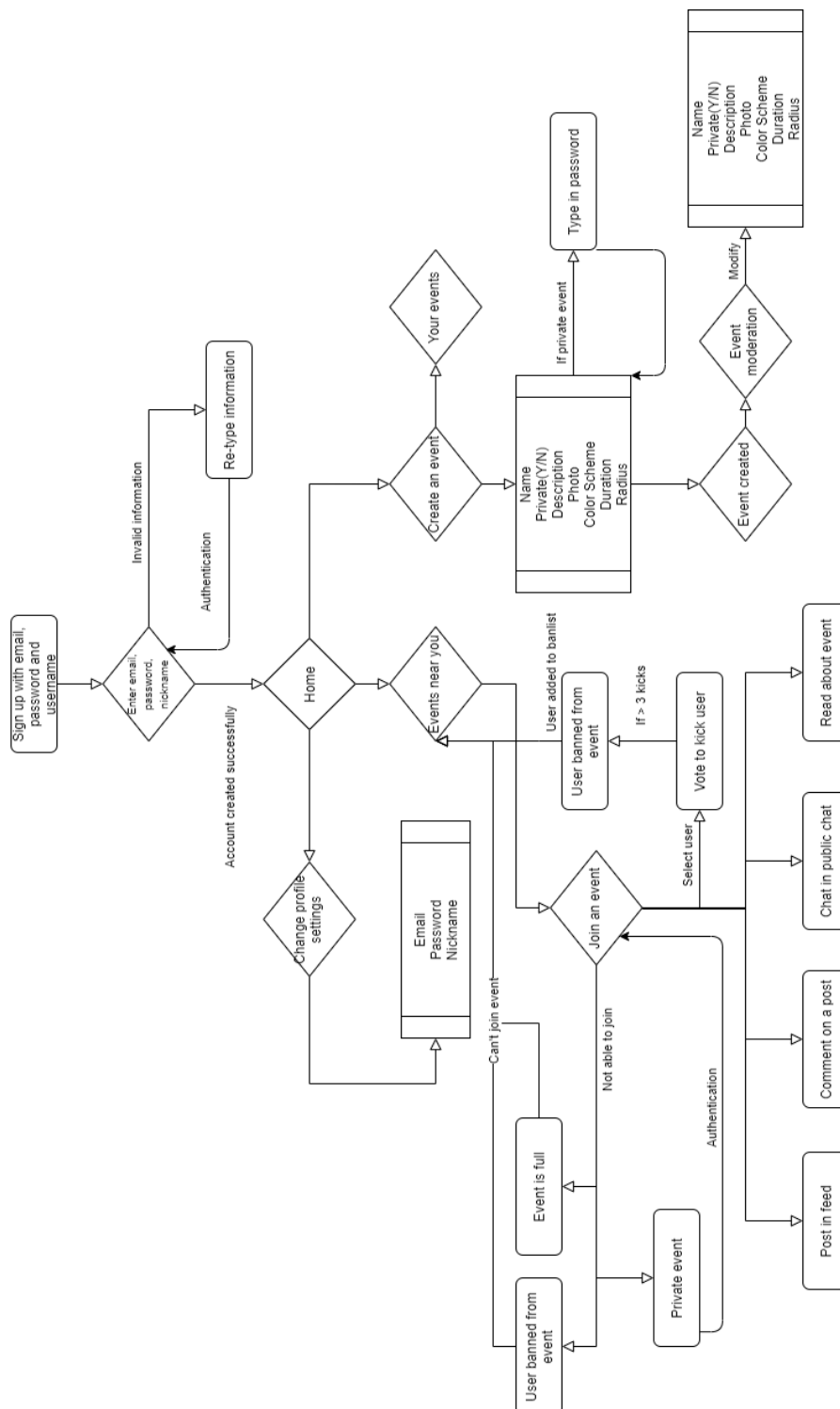
A user is created on sign up. He can then join events. Events hold variables regarding the event itself. What extends from the event is the livechat where we store information regarding every message. Similarly the posts created in the event connect to the event itself. Under posts are comments and replies which are connected to the post. Finally when a user connects to an event a report document is made, which keeps track of all reports a user gets for that event. If the user gets too many reports he is banned from the event.



8.4. Flow chart

Represents a process that users go through while using our app Hitta

Our flowchart describes what the user can do in our system and what the flow of the system is. He starts by signing up with relevant information. Then he can join an event, create an event or change profile settings. Furthermore he can chat, post and see the description of the event. If the user created an event he can then modify and moderate it.



9. Risk analysis

Describes problems we might face during development.

This is the risk analysis that we've put together over the duration of the project. All risks have a description, precaution, which is something we do to prevent the risk from happening and countermeasures, which describe how we react if the risk has become a problem. All risks are prioritized from A - D based on how much harm they can do. Finally the result of each risk can be seen in the last column.

Risk	Description	Precaution	Counter measure	Priority	Result
Quality of code	Is reading/writing data from the database optimised enough for a large user base? Are we making sure that the client is using the least bandwidth as possible?	Plan carefully how we store and read/write our data according to standards. Don't query unnecessary data.	Dive deeper into database management and querying, processing speed, limit testing more with increasing users.	A	We've optimized the queries to the best of our ability according to the latest standards in react-native/firebase.
Sickness	A team member or a close relative gets sick, or injured so a team member possibly needs to be absent for some time	Talk about it in the group, explain the situation, if the member is working on an essential task then possibly someone else can do it	Possibly code or work on anything other related to our project if possible. Or take time off as needed and come back stronger.	A	No team member or relative has gotten seriously sick which is a blessing.
Possible difficulties when developing iOS app in Windows	We might run into some restrictions developing iOS app on Windows regarding libraries/features to add	Research the topic well and look for possible alternatives	Evaluate the importance of the feature we want to add to see if it's necessary and/or find other ways to do the same/similar thing	B	We've had little to no problems with this. Main problems occurred when searching on the internet for solutions we often got answers in Xcode which is for Mac OS.
All user requirements will not be finished	We don't foresee that we will be able to finish some requirements.	Good organizing and prioritizing	Implement more important requirements first.	B	We managed to finish all requirements that we wanted to finish. Some requirements that we originally set out to implement were scrapped and therefore not implemented.

Developers variety in skillset	Developers might be good at different areas regarding code	Organize who works at coding what carefully before starting and/or be willing to let someone else code some part that they're more comfortable with	Get acquainted with the problem/code) and work onwards from that point	B	We've managed to properly divide between us tasks that we are most comfortable with. UI has been mainly in the hands of Davíð and together we've worked on other parts of the system since our skills are similar in that field.
Database implementation	Can we find a good database management system and integrate it into our application	Research different database management systems to find what best fits our product	Find a decent management system that can work with most our data but isn't exactly what we wanted	B	Davíð originally connected our app to Firebase and following that we all started working with it and now we're all comfortable working with it.
Computer crashes for one or more developers	If someone's work computer crashes in one way or another that makes it unusable	Properly update computer regularly and keep essential programs up to date as well as keep the computer free of space	Borrow spare computer/laptop from some relative or another group member if available	C	Happily we've not had any crashes. But the computers owned by Davíð and Matthías have little storage place left which has not affected the project development.
Data breach	Someone breaks into our database and gathers stored information regarding events or users	Follow safety instructions from Firebase and don't store personal details regarding our test users nor ourselves	Immediately delete all our data if we feel that's it's necessary since it's not that important for our app at this point in time	C	We've not encountered any breaches like we expected. We followed all safety notifications we got from firebase to keep our data in good shape.
Get stuck developing a key requirement of our app	If we can't finish some key requirement of our app that we originally set out to implement in our project	Calm down, take a breather, take some time to think of another approach, talk to other team members, use the internet	Carefully rethink this requirement if it's really necessary and see if we can take another approach as a team	C	It occasionally came up that some member could not completely finish some requirements, but we were vocal about it and got help from other members.
Inaccurate user and/or event GPS location	GPS location needs to be optimized in a way that we can update it regularly and that it returns accurate results	Follow documentation and enforce well written code regarding this	Implement a less accurate GPS system that roughly estimates the user location	C	Our events span some boundaries and we've checked how well the GPS works by going over the event bounds just barely and we've been kicked out of the event as desired.

Political/Government issues	If something regarding the government and/or politics prohibits some ways we're taking through our development or regarding user tests	Be semi-active on Icelandic media regarding what is going on in our community regarding the environment we're working in	Adapt to the changes accordingly and keep on working on the project	D	We've not encountered any issues regarding this.
Package Version control	React native applications use a lot of node modules that can have different versions.	Team members need to make sure to have the same versions for all packages.	Delete package.json file and run npm install again to install all the required packages again. Another way would be to individually update packages that are conflicting.	D	We've not encountered any serious problems that couldn't be fixed in a span of a couple of minutes. "Npm install" installs all new packages when pulling the project which has handled this.

10. Design

Describes how we've approached the UI design of the app for the user to use

10.1. User testing

User testing done several times over the testing of the application.

First at the early stage of the project, conducted only on the figma design where they were tasked to complete predefined tasks.

The second testing was done with friends during development to get feedback on what worked and what did not.

Third was conducted on the end state of the project to gather some feedback on functionality and look prior to final presentation.

10.2. Interview script

Interviews were conducted in Icelandic.

Kynnið ykkur, takið smá spjall til að láta notenda róast aðeins niður um hvað sem er. Fáíð leyfi fyrir því að skrifa niður/taka upp viðtalið. Kynnið síðan verkefnið.

“Hitta er samfélags app sem leyfir notendum að búa til viðburði tengda GPS staðsetningu, allir geta síðan tengst þeim viðburði og talað saman, deilt myndum og svoleiðis. Hann getur líka verið notaður til að miðla upplýsingum á staðsetningu frá sem dæmi Airwaves um hver dagskrá er.”

“Við viljum fá heiðarlegt álit um hvernig þér líst á appið og notkun þess, við erum bara að prófa upplifunina á appinu, ekki þig og þína þekkingu á því, endilega segðu okkar allar hugsanir sem skjótast upp bæði góðar og slæmar. Ef eitthvað er erfitt eða auðvelt segðu okkur svo að við getum fengið góðar upplýsingar um hverju við getum breytt.”

Spyrjið stuttar spurningar til að byrja með meðan prófari kveikir á expo og appinu. Ef þeir eru með iPhone getið þið beðið um að þeir nái í Expo Go og taka síðan mynd af QR kóðanum til að kveikja á appinu. Annars lánið þeim iPhone til að nota.

Sem dæmi:

“Við hvað vinnur þú?”

“Hve lengi hefur þú unnið þar?”

“Hvað ertu að læra?”

“Hefurðu mikla reynslu af samfélagsmiðla forritum?”

Þegar notandi er tilbúinn skal hann framkvæma verkefnin, útskýrið hvað hvert verkefni er en ekki segja honum nákvæmlega hvað á að gera, leyfið notanda að gera allt sjálfur. Þegar hann er að gera verkefnið, spyrjið hann hvað hann er að hugsa og hverju hann tekur eftir. Eftir að verkefni er búið spyrjið hve auðvelt eða erfitt það var. Hvað mætti vera betra?

Verk	Lýsing	Tími
1	Að búa til reikning og skrá sig inn.	
2	Að breyta notenda nafni / email / password	
3	Að fara í event sem settur hefur verið upp.	
4	Að setja inn póst og setja comment á annan póst, setja síðan like.	
5	Að fara inn í livechat og senda skilaboð.	
6	Að búa til sinn eigin event.	
7	Að fara inn í sinn event og breyta atriði eins og description.	

10.3. Result from interviews

10.3.1. Informal testing

Throughout the design of the application we've gotten our friends and family to test our application. From that we've gotten some good pointers, some of them are ideas that we had before but these results encouraged us to put more focus on implementing them. What we've implemented and/or fixed are the following:

- Add profile settings
- Positioning of "feed" in event navigator
- Button layout ideas while in create event process
- Host can unban users from event
- Bug where user could swipe out of event (which didn't record user count)

10.3.2. Formal testing 1

Users were tested with a prototype of our app and questioned both during and after tests.

Testers reported that navigation and UI was easy to understand. Their tasks were:

1. Join an event, go to live chat and send a message to every other member.
2. Create a private event and configure it's settings and info.
3. Discover what other events are around them in the discover page. (this page has been removed in the final product).

The reports conclude that all testers were able to complete their tasks. They however suggested ideas to implement a settings button for the user and allow event creators to specify when their events starts, opposed to the event starting immediately upon creation.

Public link to figma prototype:

<https://www.figma.com/file/7utkHrhbUqL1MvldZpHWbq/Hitta-new-UI?node-id=0%3A1>

Link to google forms questions after first user testing:

https://docs.google.com/forms/d/1qZcCvHYXYPWrYoRU_kxWVZ6_GufsHT7nv3clVQKRPwU/edit

10.3.3. Formal testing 2

Viðmæle ndur	Verk 1	Verk 2	Verk 3	Verk 4	Verk 5	Verk 6	Verk 7
1	12 s	42 s	6 s	32 s	5 s	38 s	25 s
2	15 s	52 s	8 s	27 s	10 s	92 s	42 s
3	13 s	38 s	6 s	33 s	7 s	143 s	39 s
4	21 s	45 s	7 s	29 s	8 s	47 s	38 s

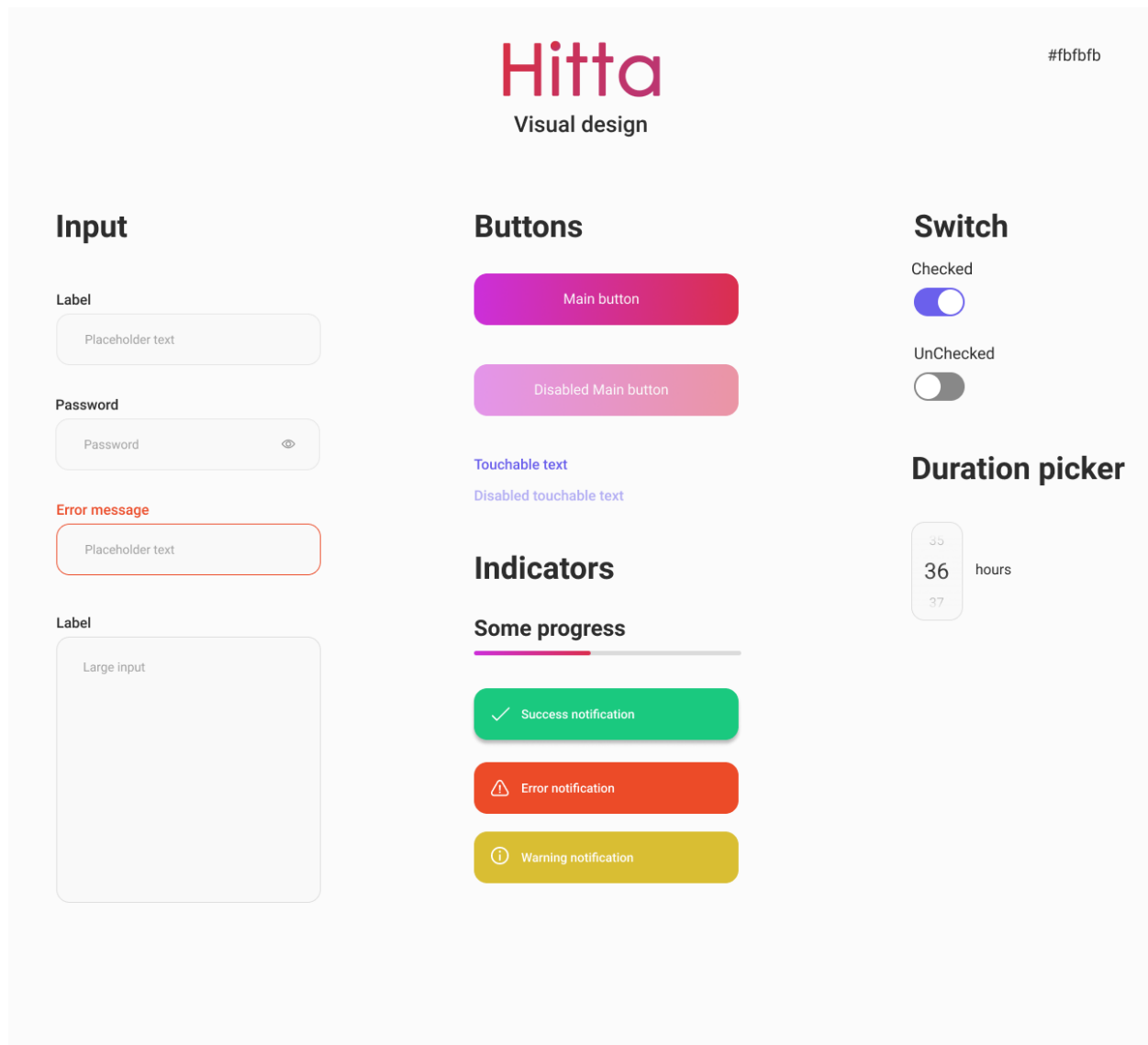
Users were all able to perform each task with minimal assistance from the interviewer, we made sure not to pressure them on time so we could see how easily/quickly people were able to use the UI to accomplish their tasks. The majority of people were able to accomplish a large percentage of the simpler tasks in a quick and easy manner. Some people took a little longer while creating an event or editing it to their needs.

Users liked the UI and the functionality of the application. They found it easy to use and understand. There was a comment on the effectiveness/use of the color theme as it only changed the color of the buttons and livechat. It also stated that the color obscured the banner image a bit but that was only from one user. One suggested an option to turn the base color scheme(white/red) and to dark/red(dark theme).

10.4. Visual design

General description of the UI.

Our main color is a red (`#DA2F4E`). This color was chosen because it psychologically represents *excitement, energy and attention*. We believe these 3 emotions represent what our users experience when using Hitta. We usually blend this color with a subtle pink color (`#CC2FDA`) to produce a beautiful linear gradient. This gradient is used across the application, e.g. buttons and loading screens. Below is the style sheet of the main components that are used across the application.



10.5 Some images from the in app UI.

Testr

Create new account

Username

Email

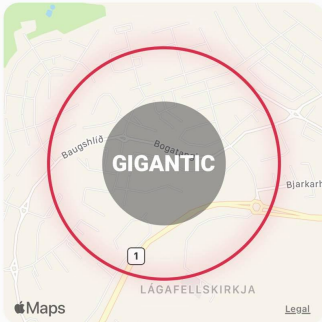
Password

Sign up

Already have an account? [Sign in](#)

Testr

Create event



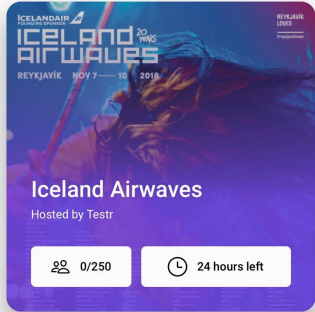
Event size

Users outside of the radius can't see or join your event.

Launch!

Hitta

Events near you



Settings Home Calendar

Post



Sá eldgosíð um daginn 🌋🌋

1 3

Testr Just now 0 Reply

Magnað 🌋🌋

Hide replies

Testr Just now 1 Reply

@Testr Sammála

Testr Just now 0 Reply

Geggað!

Add a comment Post

Testr

Create event

Event name

Public/Private event

A private event requires password from guests

Next

Testr

Create event

Color theme

Duration

Your event will disappear when the time runs out.

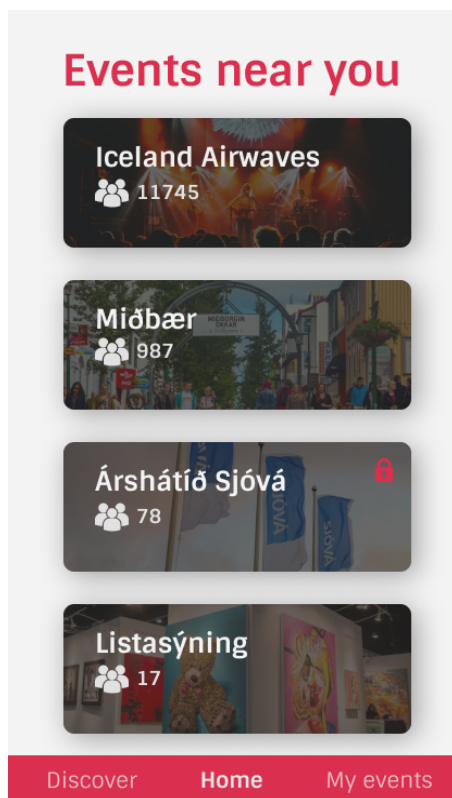
23 24 25 hours

Next

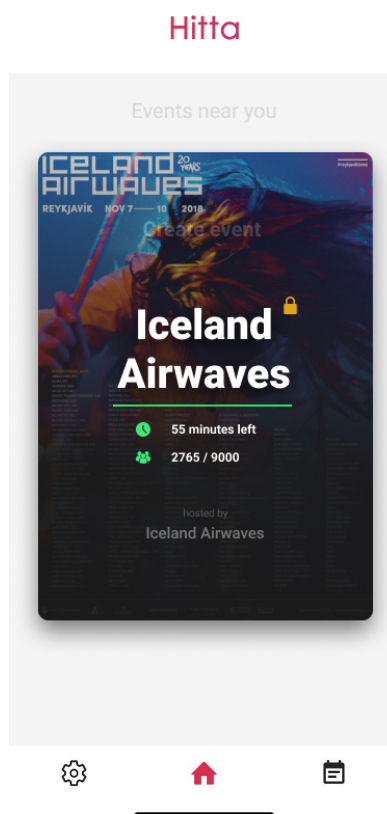
10.6. The evolution of the UI

We have made a lot of changes to our UI during development, based on user testing and because of increasing skill level in UI designing. Here you can see the home page of all three generations, where the third generation is the current state of the UI.

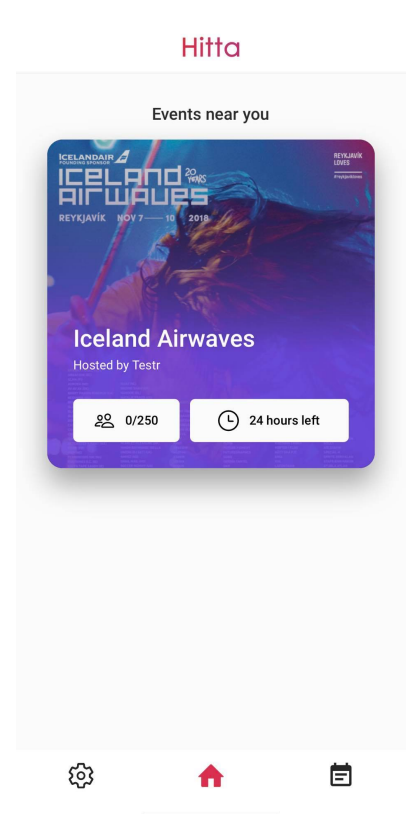
First generation



Second Generation



Third generation



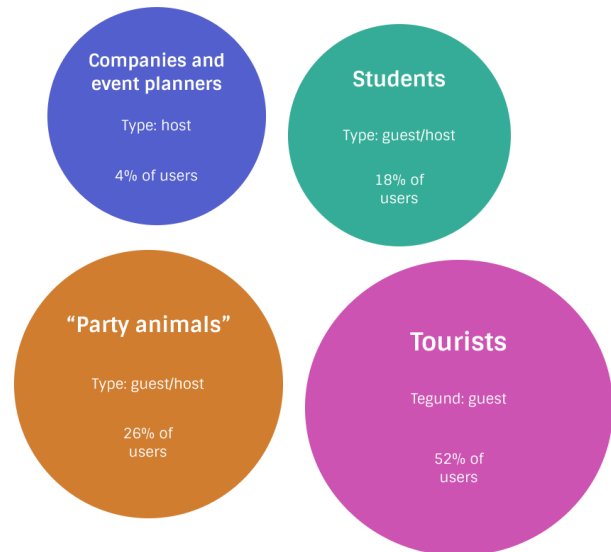
11. User groups

Describes what groups of people we've found likely to be using our app Hitta and how they will be using it

11.1. User groups

General description on our user groups.

Our user base consists of four groups, companies and event planners, students, party animals and tourists. Party animals and tourists are our biggest user groups and their requirements and desires will be prioritised in the design of the application. Companies and event planners are a small percentage of our user base but are equally important as larger groups since they will provide a large portion of the events in the application and we will therefore also prioritise their needs. Below is a user group table with information about each group.



11.2. User groups table

Different groups of people and their respective properties described

We made a user groups table to better understand what we should focus on while developing the application. Our finished product does not necessarily reflect the needs of our initial target group, which were "party animals", "tourists" and "companies" as described in the table below. We think that all ages and genders regardless of age can use the app. Though there is definitely room for improvement regarding usability for different user groups.

Name of the user group	Companies (C)	Students (S)	Party animals (PA)	Tourists in a new country (T)
The group's importance				
	1	2	1	1
Background				
Age	15-70	16-25	18-60	15-70
Gender	All	All	All	All
Education	Uni degree and under	Uni degree and under	Uni degree and under	Uni degree and under
Abilities/disabilities	Can most likely provide good moderation for their events, e.g. some of their employees moderate.	N/A	Most likely under the influence when using the app so it needs to be dummy proof.	Language barrier, events targeted to them need to be in english
General computer skills	Very Good	Very Good	Good	Good
Usage of the software				
How often	12-24 per year	2 per week	1-2 times per week(on the weekends mainly)	3 - 7 times per week
For how long	1 - 24 hours	1 - 8 hours	3 - 8 hours	2 - 5 hours
User skills	Very good general knowledge of the system	Very good general knowledge of the system	Very good general knowledge of the system	Good general knowledge of the system
User attitude	Generally very good since it offers them to host events for employees/customers	Generally very good since they can socialize with other students	Generally very good since they can socialize with other people which are also partying	Generally very good since they can explore new country and socialize with other people
Number of users	2500	10.000	15.000	30.000
Type of user	Host	Host/Guest	Host/Guest	Guest
Context of use				
Technical environment	iOS/Android Smartphone	iOS/Android Smartphone	iOS/Android Smartphone	iOS/Android Smartphone
Real environment	Used at hosted events.	Anywhere to check for events. Mainly used at the exact location of the event to socialize.	Anywhere to check for events. Mainly used at the exact location of the event to socialize.	Anywhere to check for events. Mainly used at the exact location of the event to socialize.
Main user goals				
	Be able to host an event with all relevant information.	Be able to host and join events that are school related.	Be able to host and join events to enhance the entertainment value of an event.	Be able to find and join events to experience the culture and take part in events.

12. User requirements

A list of user requirements that we set out to implement based on priority

We started out with a general idea of requirements that we wanted to implement into our app but those requirements have changed over the course of the project and their respective priority. We've gotten rid of unwanted requirements and added others. The requirements have a name that describes the requirement, then all user groups that this requirement fits with, priority from A - C and lastly y/n (yes/no) if the requirement has been implemented.

CO: Companies **PA:** Party animals **S:** Students **T:** Tourists

Number	Name	User Groups	Priority (A/B/C)	Finished (y/n)
1.	Create a public event	CO, PA, S	A	y
2.	Create name for event	CO, PA, S	A	y
3.	Mark radius from center of event	CO, PA, S	A	y
4.	Create name for user	CO, PA, T, S	A	y
6.	Join public event	PA, T, S	A	y
7.	Post in an event	PA, T, S	A	y
8.	Comment on a post	PA, T, S	A	y
9.	Create a private event	CO, PA, S	A	y
10.	Users can change profile settings	CO, PA, S, T	A	y
11.	Join private event(password)	PA, T, S	A	y
12.	Public chat in an event	PA, T, S	B	y
13.	Paid users can put in more information regarding the event	CO, PA, T, S	B	n - cancelled
14.	Report system in events	CO, PA, S, T	B	y
15.	Event host moderation and event settings	CO, PA, S, T	B	y
16.	Send private messages in an event	PA, T, S	C	n - cancelled

17.	Search for public events on map	PA, T, S	C	n - cancelled
18.	Notification to join event when in radius	PA, T, S	C	n - cancelled

13. User stories

What the user can do in Hitta

These stories describe what users can do and what their goal is by doing these things.

User stories general setup:

As a <type of user>, I want to be able to <some goal> so I <some reason>

13.1. User

As a user I want to be able to...

- sign up so I can create an account.
- sign in so I can enter the application.
- join an event so I can interact with other people near me.
- chat with other people in the same event as me so I can let them know what I'm thinking.
- post pictures in the feed so I can let everyone know what I'm doing and/or where I'm at.
- read the description of the event so I know what it is about exactly.
- modify my profile settings so I can change my account information.
- create an event with fitting information so I can let other users join.
- report another user so I can get him banned from the event for sharing inappropriate content.

13.2. Host

As a host I want to be able to...

- modify my event settings to change the color theme of the event or other settings related to my event

- unban users that have been banned for reasons that I don't agree with or to give them a second chance
- delete my event since it's not going to be in use anymore

14. Use cases

Describe different processes when using the system, how the ideal process is and alternatives

Use cases describe different scenarios that the user can be at, at any point in our application. The use cases have a name, a number which defines the number of that use case and a priority from A - B which describes their importance. Precondition describes what requirements are needed to get to that use case. The Main success scenario describes how the ideal flow should be and the extensions describe what can go wrong and how that is dealt with. Post condition describes what the user has succeeded in doing after finishing the use case. And finally the actor describes whether or not the user is a Host ("owner" of event) or a User. User and Host are inherently the same, but are used in different context.

Name	User signs up
Number	1
Priority	A
Precondition	No precondition
Main success scenario	<ol style="list-style-type: none"> 1. User inputs username 2. User inputs email 3. User inputs password
Extensions	<ol style="list-style-type: none"> 2. User inputs invalid email <ul style="list-style-type: none"> - User has to type a valid email 3. User inputs invalid password <ul style="list-style-type: none"> - User has to choose another password
Post condition	An account is created for the user
Actors	User

Name	User logs in
-------------	--------------

Number	2
Priority	A
Precondition	User has an account
Main success scenario	<ol style="list-style-type: none"> 1. User inputs his email address 2. User inputs his password
Extensions	Information is incorrect, retype email and password
Post condition	A user has logged in
Actors	User

Name	User joins an event
Number	3
Priority	A
Precondition	User is logged into his account
Main success scenario	<ol style="list-style-type: none"> 1. User clicks on event and joins it
Extensions	<ol style="list-style-type: none"> 1. <ol style="list-style-type: none"> a. Event is private - User types in event password b. Event is full - User can't join event c. User is banned from event - User can't join event unless unbanned from host
Post condition	The user has joined an event
Actors	User

Name	User posts in an event
Number	4
Priority	A
Precondition	User is in an event
Main success scenario	<ol style="list-style-type: none"> 1. User clicks plus (+) sign in event feed 2. User decides if he/she wants to post a picture with the post 3. Choses picture and/or write text 4. Post to feed
Extensions	<ol style="list-style-type: none"> 2. User has not allowed access to photo gallery and can't post a picture User is banned while writing a post so he is kicked out of the event
Post condition	The user has posted in the event
Actors	User

Name	User provides feedback on a post in an event
Number	5
Priority	A
Precondition	User is in an event
Main success scenario	<ol style="list-style-type: none"> 1. User can... <ol style="list-style-type: none"> a. Like a post b. Comment on a post c. Like a comment d. Reply to a comment e. Like a reply f. Reply on a reply
Extensions	<ol style="list-style-type: none"> 1. <ol style="list-style-type: none"> a) No post in feed or post already liked b) No post in feed c) No comment on post or comment already liked d) No comment on post e) No reply on post or reply already liked f) No reply on post
Post condition	The user has provided feedback to a post
Actors	User

Name	User chats in live chat
Number	6
Priority	A
Precondition	User is in an event
Main success scenario	<ol style="list-style-type: none"> 1. User navigates to "livechat" in event navigation 2. User types in message at bottom of screen 3. User sends message to livechat
Extensions	User is banned while writing a message so he is kicked out of the event
Post condition	The user has chatted in an event
Actors	User

Name	User reads about event
Number	7
Priority	A
Precondition	User is in an event

Main success scenario	<ol style="list-style-type: none"> 1. User navigates to "about" in event navigation 2. User reads description of event
Extensions	2. There is no description of event
Post condition	The user has read about the event
Actors	User

Name	User reports another user for inappropriate content
Number	8
Priority	B
Precondition	User is in an event
Main success scenario	<ol style="list-style-type: none"> 1. User clicks on another username in an event, either on live chat or in feed 2. User clicks "Report" in popup modal
Extensions	2. User has already been reported by this user
Post condition	Other user is reported
Actors	User

Name	User changes profile settings
Number	9
Priority	A
Precondition	User is logged in
Main success scenario	<ol style="list-style-type: none"> 1. User navigates to profile page 2. User selects to modify the following: <ol style="list-style-type: none"> a. Nickname b. Email c. Password 3. User types in new information
Extensions	3. User types in invalid information - retype information
Post condition	The user has changed profile settings
Actors	User

Name	User logs out
Number	10
Priority	A

Precondition	User is logged in
Main success scenario	<ol style="list-style-type: none"> 1. User navigates to profile page 2. User clicks "Log Out"
Extensions	None
Post condition	User is logged out
Actors	User

Name	User creates an event
Number	11
Priority	A
Precondition	User is logged in
Main success scenario	<ol style="list-style-type: none"> 1. User navigates to "create event" page 2. Input name for event 3. Choose if private or public - password if private 4. Input description for event 5. Choose image banner for event 6. Choose color theme for event 7. Choose duration of event 8. Choose size of event 9. Event is launched and created
Extensions	<ol style="list-style-type: none"> 3. Invalid password length - change password 5. User has not allowed photo gallery access
Post condition	The user has created an event
Actors	User

Name	Host modifies event settings
Number	12
Priority	B
Precondition	User is a host of an event, host is in his event
Main success scenario	<ol style="list-style-type: none"> 1. Host clicks settings icon at top right 2. Host selects to modify one of the following: <ol style="list-style-type: none"> a. Event name b. Description c. Banner d. Color theme e. Private/Public
Extensions	2. e) If changed from public to private - type in password

Post condition	The host has changed the event settings
Actors	Host

Name	Host moderates event
Number	13
Priority	B
Precondition	User is a host of an event, host is in his event
Main success scenario	<ol style="list-style-type: none">1. Host clicks settings icon at top right2. Host selects the option to unban users3. Host selects users to unban4. Selected users are unbanned
Extensions	3. a) No banned users in event
Post condition	The host has moderated the event
Actors	Host

15. System requirements

Requirements that Hitta needs to be able to work properly.

Name	Description	Priority
Networking	Important to have proper networking, for real time chat, feed and connection to database.	A
Real time chat	That people are able to chat together in real time without data being lost.	A
Database management	Connection to database and proper database host. For logging, user info and event info.	A
GPS location	We can store and use the GPS location of events and users. It's sensitivity matches what works for our app.	A

16. System tests

How we approached and did our tests to check how the application handles increasing amount of users

In the last sprint; sprint 14, we decided to limit test our application in regards to user count. Like stated earlier we are using Firebase to store our data and on a free plan which restricts the number of reads/writes (bandwidth) per day both in Firebase Firestore and Firebase Storage, where event pictures from posts and event banners are kept.

We've been close to the limit on our Firebase Storage while testing our code on the daily, so allowing more users to simultaneously use our app we knew that this limit would be reached and we could not store any more pictures in Firebase Storage. Regardless of the pictures, we

wanted to limit test the app in regards to all other data stored and on how well the application would handle it in terms of speed.

16.1. Transactions test

Transactions in Firebase are supposed to ensure correct increment or decrement in regards to different variables when multiple users interact with it at the same time according to the documentation from Firebase. Transactions are used for the variables that have to be updated correctly in order to display correct numbers in real time and to make the application work as planned.

We are using transactions for:

- User count in event
- Likes on posts
- Likes on comments
- Likes on replies
- Reports that user has received

We tested the functionality of transactions for each variable mentioned above. We approached all tests in a really similar way: All 3 members spammed the different transaction operations at the same time for the same event.

For example: We constantly joined an event and then exited it again. The transactions updated correctly but when spammed it took up to a maximum of 1 second~ to update for the user to see. We are okay with that result because most of the time people will not be likely to do this, but this was mainly to mimic large amounts of users joining and exiting an event.

16.2. Database limit test 1

There are 3 members in our group, so we started out by joining an event all together that Matthías created. Since we are in different locations we “hacked” the event location to allow us all to join. We then all took turns posting in the feed, which worked out fine. Likes updated correctly and the comments and replies displayed correctly when another user gave feedback. We tried to chat simultaneously in the livechat section by spamming messages and Firebase Firestore could handle that perfectly. Lastly Matthías tested out the event settings. He changed

the name, description, color theme, banner image and changed the event from public to private. The other users; Davíð and Stefán saw the changes in real time.

This test went well, we didn't cross our limit in regards to data and everything ran smooth as expected.

16.3. Database limit test 2

Load 40 "fake" users into an event and likewise make posts from them, comments, replies, likes and chat in livechat. Then the 3 of us joined the event and we properly loaded in all the data from the users. Following that we did the same as in test 1, we posted, commented, liked and chatted together and everything worked as planned. The main difference from test 1 is the added data that is loaded upon joining an event which was handled well.

16.4. Database limit test 3

All of the group members tried to get friends and/or family to join us in the same event and to interact with each other in it. The 3 group members got 6 other people to join them. We made all participants join an event that had previously been made with "fake" users like in test 2, with data already in the event. When there were 9 people interacting with each other at the same time it often depended on what type of iPhone users were using, older iPhones were on average a little slower in terms of speed. But the data itself was handled well in terms of write/read from our database which was the main goal of the test. But in this test we went over our limit of 1 GB bandwidth for storing post images and event banners in Firebase Storage which was expected. We managed to post and offer feedback and chat to each other in livechat. We then tested out our banning feature properly. We tried to ban 3 members from the event. Then the event host unbanned them from the event settings; he selected all 3 users from the banlist and unbanned them. The users were then able to join the event again.

16.5. Thoughts about scalability for the future

The first obstacle in our way is the allowed bandwidth in Firebase, we would need to upgrade our plan to a paid one. Proper tests were not written in terms of max user population because of the bandwidth restriction in Firebase. The queries made to write and read data were optimized throughout the development, but we believe they could always be made better. If we plan to continue working on Hitta throughout the summer we believe we have made a good foundation that we can build on.

17. Progress summary

What we have implemented into our app Hitta, categorized

Functionality

- The home page displays all events that the user is in radius with.
- Users can tap on public events and navigate to that unique event page.
- Users can read the event description in the about page.
- Users can post images or text to the feed.
- Users can comment on posts or reply to comments.
- Users can like posts, comments and replies.
- Users can join the live chat and interact with other users in real time.
- Private events are displayed with a lock symbol in the home page and are restricted. The user cannot join private events unless they have the password.
- Users can report other users inside the event for inappropriate content.
- User gets kicked and banned from the event when he gets reported 10 times.
- Users can create events which are then stored in our database.
- Events and all their content is deleted from the database when the event lifetime expires.
- Event hosts can modify event settings or delete the event completely.
- Event hosts can moderate their event by unbanning people.
- Users can sign up by using an email address, username and password.
- Users can sign in by using their email address and password.

- Users stay logged in after restarting the application.
- Users can log out.
- Users can change their email, username and password.
- Users are kicked out of the event when they go out of the event range.
- Hosts can see their own events in my events page.

Database

- Events in the database are queried by their regions, allowing users to only query events that are in their region which reduces reading from the database drastically.
- Posts are stored in the database, where they possess the event id they belong to.
- Images for event banners and posts are stored in our database, provided by Firebase storage.
- Users are authenticated by Firebase user authentication.

UI and styling

- The app UI is complete and styling is consistent throughout the application.
- Roboto is used as the default font for the application.
- A logo has been designed for the app header.

18. Sprint plan

All sprints planned throughout the semester with tasks in each sprint

Tasks that are complete are marked with **green**.

Tasks that are ongoing are marked with **yellow**.

Tasks that have not yet been started are marked with **black**.

All sprints have a weight of 20, which we then shared between tasks prioritized by importance which you can see marked after each requirement. The importance represents the time spent on each task roughly.

From start of this semester until 2. March we had two week long sprints. We then changed to one week long sprints at 3. March. This was done to better keep track of our progress and worked out nicely.

Sprint setup:

- **Sprint 0:** 27. Jan - 2. Feb
- **Sprint 1:** 10. Feb - 26. Feb
- **Sprint 2:** 17. Feb - 23. Feb
- **Sprint 3:** 24. Feb - 2. March
- **Sprint 4:** 3. March - 9. March (Change from 2 week sprints to 1 week sprints)
- **Sprint 5:** 10. March - 16. March
- **Sprint 6:** 17. March - 23. March
- **Sprint 7:** 24. March - 30. March
- **Sprint 8:** 31. March - 6. April
- **Sprint 9:** 7. April - 13. April
- **Sprint 10:** 14. April - 20. April
- **Sprint 11:** 21. April - 27. April
- **Sprint 12:** 28. April - 4. May
- **Sprint 13:** 5. May - 13 May

Sprint 0	27.jan – 2.feb	Status: Done
<ul style="list-style-type: none">• Create user requirements - 4• Identify user groups - 2• Vision - 2• Work agreement - 3• Risk analysis - 3• Diagrams<ul style="list-style-type: none">○ Class - 2○ Flowchart - 2• Ideas regarding what features we can possibly add - 1• Timetable - 1		
<p>Conclusion: Everything went well this sprint except grasping the idea of risk analysis and what is expected there, we managed to make the required documents but most of them are kept alive and updated through the semester</p>		

Sprint 1	3. - 9. feb	Status: Done
<ul style="list-style-type: none">• Presentation for 8th of february - 5• Figma framework of potential design - 5• User testing of Figma design - 5<ul style="list-style-type: none">○ Questions○ Tasks• Redesign figma based on feedback - 3• Reflect on user answers from questions - 2		

Conclusion: Everything went well this sprint, we got good feedback from our instructors regarding the next steps

Sprint 2	10. - 16.feb	Status: Done
<ul style="list-style-type: none">• Coding semi rough interface based on Figma sketches - 8<ul style="list-style-type: none">○ Home page○ Create event page• Navigation for existing interfaces - 12		
Conclusion: Initializing the project and setting up the absolute basic structure worked well as well as the navigation		

Sprint 3	17. - 23.feb	Status: Done
<ul style="list-style-type: none">• Get user gps location, latitude and longitude - 9• Coding interface<ul style="list-style-type: none">○ Event page - 4• Navigation for existing interfaces - 3		

- **Prepare presentation for Stöðufundur 1 - 4**

Conclusion: GPS location was implemented, styling for existing pages and better navigation made. GPS location will be optimized better throughout the semester

Sprint 4	24.feb - 2. march	Status: Done
<ul style="list-style-type: none">• Styling interfaces properly - start - 4• Navigation for create event process - 4• Navigation inside event, about, feed, livechat - 5• Integrate map to use in ui from react-native-mapview - 7		
<p>Conclusion: Navigation and styling went well but integrating the map took some time, since we needed the map to zoom out when the user changes the size of the event.</p>		

Sprint 5	3. - 9.march	Status: Done
<ul style="list-style-type: none">• Validation of user input in event creation. - 3• Finish all screens in event creation and get all inputs for the event object. - 6• Create redux global state to keep track of user gps and other info. - 6• Get events from the database for display in home page. - 5		
<p>Conclusion: User input validation went well and the basic flow of event creation went good, but styling, information regarding the event and layout is going to be worked on throughout the semester. We started to implement redux global state in a way that didn't work, but we then found a good solution which worked</p>		

Sprint 6	10. - 16.march	Status: Done
<ul style="list-style-type: none">• Query only events that the user is inside. - 8• Send created events by the user to the database. - 6• Delete events after they expire from the database. - 6		
<p>Conclusion: We managed to connect our app to Firebase and properly figured out how to read and write data regarding events and displaying them. Likewise we managed to delete them when expired date is due</p>		

Sprint 7	17. - 23. march	Status: Done
<ul style="list-style-type: none">• Create feed component for event<ul style="list-style-type: none">◦ Users are able to post text - 6• Create data storage for photos for event creation and posts. - 5• Logo design and color scheme. - 4• Undirbúningur fyrir Stöðufund 2. - 5		
<p>Conclusion: The feed component took a lot of time and in this sprint we only implemented the feature to post text without an image. Creating a data storage was fairly easy since firebase offers that service. The photos stored in the storage are named after the object (event/post) they belong to.</p>		

Sprint 8	24. - 30. march	Status: Done
<ul style="list-style-type: none">• Kick users out events if they walk out of event radius and notify them of it with a modal. - 2• Create a sign in screen component. - 7<ul style="list-style-type: none">◦ User authentication from firebase. <p>//Scrapped phone number authentication doesn't work well enough and can be expensive</p> <p>//Implementing email authentication next week.</p> <ul style="list-style-type: none">• Allow users to post in event feed, pictures and text. - 4• Properly store pictures from posts in storage. - 2		

- **Get event banners from storage/Banners. - 2**
- **Fix navigation so bottom tabar does not show inside event page - 1**

Conclusion: We started out by implementing authentication via phone number which didn't work out as well as we planned, so we plan to move onto email/password authentication next week. Storing pictures is not supported in Firebase Firestore so we store our pictures in a separate database; Firebase Storage which we then linked to each event respectively. We implemented a feature where you can post in event feed, but styling was not done in this sprint.

Sprint 9	31. - 6. apríl	Status: Done
<ul style="list-style-type: none">• Create a modal for password login to private events and authenticate to the database if the password is correct. - 2• Modify phone authentication to email authentication. - 9• Create livechat component for event - 9<ul style="list-style-type: none">○ Users are able to message each other in livechat.		
<p>Conclusion: The password modal functionality is complete but lacks styling. Authentication for users was changed from a phone number to an email and password authentication. Users are able to send messages to each other in livechat. Livechat lacks the report modal for users.</p>		

Sprint 10	7. - 13. apríl	Status: Done
<ul style="list-style-type: none">• Exam period, team members were busy studying for exams. - 20		

Sprint 11	14. - 20. april	Status: Done
<ul style="list-style-type: none">• Display events to users created by him on the “my events” page. - 4<ul style="list-style-type: none">○ Redesign firebase model for events• Livechat component styling - 5• Automatic login for users that have already logged in before - 4• Password authentication for private events - 3• Display dynamic map when choosing radius for event - 4		
<p>Conclusion: Dynamic map fully working inside event creation when the user is selecting the size of the event. Users that have logged in before are automatically logged in. The firebase model was restructured, events are not split into different collections by their region anymore, but are instead all in the same collection. Events have their region as a value instead so the user can therefore just query all events that are in his region.</p>		

Sprint 12	21. - 27. april	Status: Done
<ul style="list-style-type: none">• Allow users to like and comment on posts in event feed. - 6• Display a settings icon for event hosts where they can modify info and moderate the event. - 4• Database query optimization - 2• Report system - 2• Fixing login issues - 6		
<ul style="list-style-type: none">• Create database triggers that delete events instead of the client performing the delete query. Needs a premium version of firebase. <p>CANCELLED, since firebase is having problems with accepting payment from Iceland, may be implemented if we can get it to work.</p>		
<p>Conclusion: Implemented comments and like into the event feed as well as moderation for the host to manage the event, along with reporting users to be banned from the event. We ran into a bug when implementing logging out that crashed the app when trying to relog back in. Implementing that fix delayed finishing user settings this week.</p>		

Sprint 13	28. - 4. maí	Status: Done
<ul style="list-style-type: none">• Splash screen for loading the app, make sure all fonts are loaded and user location has been set in the global state. - 2• Decrease userCount for event if user exits app - 2• Completely finish styling and functionality of the event creation process. - 8<ul style="list-style-type: none">○ Includes styling.• Create settings page for user, where he can modify: - 8<ul style="list-style-type: none">○ Nickname○ Image○ More(?)		
<p>Conclusion: We finished the user settings page where he can modify his nickname, email and password. User count for events works and is properly displayed in the event banner from the homepage. A couple of iterations were done when choosing the best splash screen and we landed on a good solid solution which fit.</p>		

Sprint 14	5. - 13. maí (Last sprint extended to deadline)	Status: Ongoing
<ul style="list-style-type: none">• Finish final lokaskýrsla, notendaleiðbeiningar and rekstrarhandbók.• Pop up notification indicators• Styling and simplifying user settings modals• Styling create post modal• Styling notifications• Styling permission screen• Styling event page header• Redesign figma UI.• UI styling on existing screens for user testing.• Allow users to like comments, replies and posts• Handling image library permissions• User testing• Delete all content related to event if it gets deleted• Fixing memory leaks in event settings• Fixing database listeners• Styling feed event page and post page• Create a settings screen for the event host to modify about page and other values.• Style about page• Finish functionality in event settings.• Finish styling on password and report modals• Fixing like transaction throttle, wait for user id to be uploaded to likedBy array in post/comment/reply.• Commenting code and removing redundant functions, imports and components• System testing• Users can report other users by clicking on their name wherever in the event		
<p>Conclusion: We finished all the remaining tasks for the application, did some limit testing on the system, user testing and finished all reports for the final turn in of the project.</p>		

19. The future of Hitta

The future of Hitta is yet unknown, since the possibilities of its applications are endless. You might be asking yourself, how does Hitta sustain itself and make money?

The answer is simple. Hitta's end goal is to offer a premium service to verified and large events. We would cap the event radius, max guest population and other features for free hosts, while offering boundaryless customization to events for paid hosts. But that's just the tip of the iceberg.

Imagine hosting a large event, e.g. a music festival and getting live analytics of how your guests are interacting and moving inside the event. What stage are they at? When did they arrive and leave? How many people were watching a particular artist at a particular time? These analytics allow event hosts to analyze the behaviour of their guests in real time and make future changes according to the data.

This incentivises hosts of larger events to get as many of their guests on the application to make their data more insightful. More users, more accurate data and this will directly drive the adoption of Hitta. As a guest, if you're worried about your personal data, you are anonymous on Hitta so none of your data will be linked to your name.

That's our current end goal for Hitta and we are excited for what the future brings.

20. Final words

This final project has been tough but fun at the same time. In the beginning we were not sure where to start, what to do and how to do it. Since we were not working with a company there were no clear standards or procedures to follow. This is our first time developing a large project and we were therefore not too familiar with how we should present and track our sprint progress.

We created different reports, graphs, wrote code and did other relevant tasks to the best of our ability. We got tips from our instructor on what could be done better in our reports and we followed these instructions as well as we could. React-native is a fun environment to work in and developing an application for iOS that we can interact with and show other people is very exciting.

We have tons of ideas that we can implement into Hitta if we continue development but that has not been decided yet. We want to give thanks to friends and family for their patience while we were working on this project and to everyone who helped in one way or another; whether it was testing, coming up with ideas, pointing out mistakes or anything else related to this project. We've learned a whole lot throughout this process which we can take with us into the future.

Hitta