How to run a small social network site for your friends

network dandelion

Black outline of a dandelion seedhead resembling a network diagram

Run your own social

How to run a small social network site for your friends

by Darius Kazemi, July 8, 2019

Table of Contents

How to run a small social network site for your friends

- Who this guide is for
- Social solutions to social problems

What makes a small social network site different from a big one?

- You control the computer that runs the site
- You can modify the software
- You get to make the social rules and policies
- Wait, who is "you"?

Why run a small social network site?

- Software can be complicated where necessary
- You can have hyper-specific norms
- Why to NOT run a small social network site

What you can do today

- Keep it small
- You are the party host
- Provide a custom introduction to your network for every user
- Group activities
- Keep it interconnected
- Funding
- The bus problem
- Code of Conduct
  - Enforcement
- Technical recommendations
  - Examples of services you can provide

What we need to do in the future

- Places where we need better tech
  - Fluidity of identity and the ability to migrate
  - Let people keep things in the community
  - Server forking should be easy
  - Lean software that doesn't have to scale
- More on scale
- Beyond local and public: the neighborhood

Conclusion

- Hard questions
- So what do you, the reader, do next?
  - If you are comfortable with servers and hosting
  - If you are not comfortable with that but would like to learn
  - If learning the tech stuff is realistically not going to happen
- Running a small social network site is completely possible to do on the side

Acknowledgments

About the author

License

Older Versions

How to run a small social network site for your friends

Since August 2018 I have run a social network site called Friend Camp for about 50 of my friends. I think Friend Camp is a really nice place, and my friends seem to agree that it has enriched our lives. I'd like to see more places like Friend Camp on the internet, and this document is my attempt to provide some practical guidance as to how you might run a social network site like this.

Friend Camp is an intentionally small social network so the home page doesn't look like much if you don't have a login. You can read some of our policies if you like.

Who this guide is for

If you're tired of Facebook or Twitter or wherever else and have thought that there's got to be a better way, this is for you.

If you currently run a social network server for people besides just you, using software Mastodon or Pleroma or whatever else, this is for you.

If you have some programming experience, this is for you.

If you have no programming experience, this is for you.

Social solutions to social problems

This document exists to lay out some general principles of running a small social network site that have worked for me. These principles are related to community building more than they are related to specific technologies. This is because the big problems with social network sites are not technical: the problems are social problems related to things like policy, values, and power.

There are still some areas where technical progress is needed, and one section of this document discusses some of those areas.

Running a social network site is community building first and a technical task second.

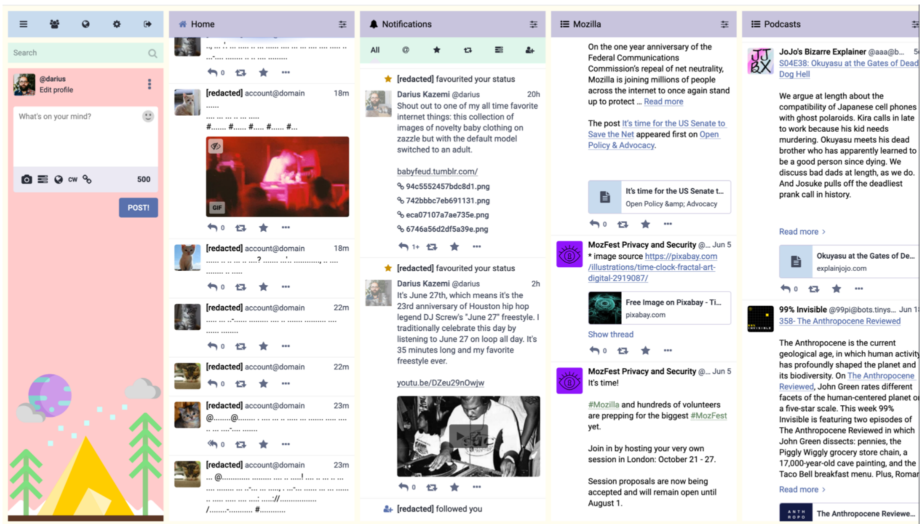
And while community building is hard work, it's often worth it.

This is my pitch to you: using big social media sites is easy, but you pay a steep price for it. You should consider running your own site, which is harder, but can be extremely rewarding.

For those of you familiar with the term, I'm discussing federated social media. I find that term nearly useless though and have tried to keep discussion of federation itself to asides like this one.

What makes a small social network site different from a big one?

Friend Camp looks like this on a desktop web browser:



[redacted] account@domain 18m

[redacted] account@domain 18m

[redacted] account@domain 22m

[redacted] account@domain 22m

[redacted] account@domain 23m

[redacted] account@domain 23m

[redacted] favourited your status

Darius Kazemi @darius 20h
Shout out to one of my all time favorite internet things: this collection of images of novelty baby clothing on zazzle but with the default model switched to an adult.

babyfeud.tumblr.com/
94c5552457bdc8d1.png
742bbbc7eb691131.png
eca07107a7ae735e.png
6746a56d2df5a39e.png

[redacted] favourited your status

Darius Kazemi @darius 2h
It's June 27th, which means it's the 23rd anniversary of Houston hip hop legend DJ Screw's "June 27" freestyle. I traditionally celebrate this day by listening to June 27 on loop all day. It's 35 minutes long and my favorite freestyle ever.

youtu.be/DZeu29nOwjw

[redacted] followed you

On the one year anniversary of the Federal Communications Commission's repeal of net neutrality, Mozilla is joining millions of people across the internet to once again stand up to protect … Read more

The post It's time for the US Senate to Save the Net appeared first on Open Policy & Advocacy.

It's time for the US Senate t…
Open Policy &amp; Advocacy

MozFest Privacy and Security @… Jun 5
* image source https://pixabay.com/illustrations/time-clock-fractal-art-digital-2919087/

Free Image on Pixabay - Ti…
pixabay.com

Show thread

MozFest Privacy and Security @… Jun 5
It's time!

#Mozilla and hundreds of volunteers are prepping for the biggest #MozFest yet.

Join in by hosting your very own session in London: October 21 - 27.

Session proposals are now being accepted and will remain open until August 1.

JoJo's Bizarre Explainer @aaa@b… 5
S04E38: Okuyasu at the Gates of Dead Dog Hell

We argue at length about the compatibility of Japanese cell phones with ghost polaroids. Kira calls in late to work because his kid needs murdering. Okuyasu meets his dead brother who has apparently learned to be a good person since dying. We discuss bad dads at length, as we do. And Josuke pulls off the deadliest prank call in history.

Read more ›

Okuyasu at the Gates of De…
explainjojo.com

99% Invisible @99pi@bots.tinys… Jun 1
358- The Anthropocene Reviewed

The Anthropocene is the current geological age, in which human activity has profoundly shaped the planet and its biodiversity. On The Anthropocene Reviewed, John Green rates different facets of the human-centered planet o a five-star scale. This week 99% Invisible is featuring two episodes of The Anthropocene Reviewed in which John Green dissects: pennies, the Piggly Wiggly grocery store chain, a 17,000-year-old cave painting, and the Taco Bell breakfast menu. Plus, Roman

Read more ›

The Anthropocene Reviewe…

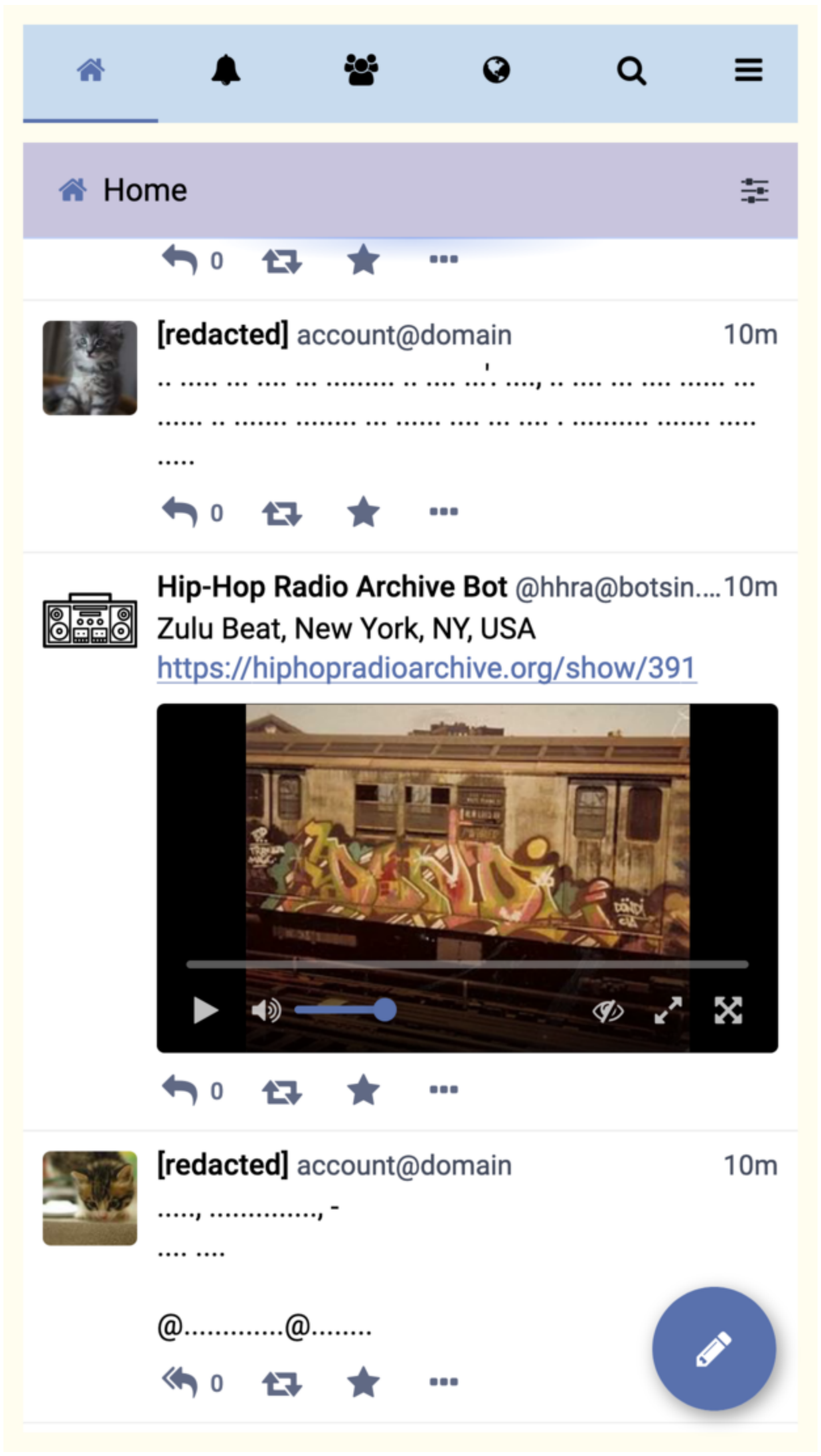
Figure 1: Text obscured to protect the innocent.

And like this on phones:

It's fairly Twitter-like at first blush: you post short messages, you can "@" individual users, you can follow and be followed, you can use hashtags, and you can boost things that you want your followers to see. While Friend Camp is small, the universe of content we have access to is big. We are connected to about 5,000 other communities on the web, allowing us to interact with hundreds of thousands of users if we want.

Friend Camp is a modification of a program called Mastodon—I'll get into the specifics later, but I want to say right away that I did not create 99.999% of the software and I'm grateful for the work of the Mastodon authors and contributors.

So far, this seems like an underpowered Twitter, but there are some key differences from Twitter:



[redacted] account@domain 10m

.. ..'. ...., ...
....
.....

Hip-Hop Radio Archive Bot @hhra@botsin.... 10m

Zulu Beat, New York, NY, USA
https://hiphopradioarchive.org/show/391

[redacted] account@domain 10m

....., ..., -
....

@............@........

Figure 2: The phone view of Friend Camp.

- you control the computer that runs the site
- you can modify the software that powers the site
- you get to make the rules and policies

Next I'll explain each of these in detail.

You control the computer that runs the site

Big social media sites all run on computers, usually a big complex network of computers. When you type "facebook.com" into your browser or load up your Facebook app, what happens is that your phone sends a message to a computer owned by Facebook, which then coordinates with a bunch of other computers owned by Facebook to collect all the information that is then displayed to you.

In the context of Facebook, this means that everything you do on Facebook passes through computers that they control. And then they take the information about what you do and sell it to advertisers, directly or indirectly.

But a small social network site doesn't need a huge complex network of computers. One computer can be enough. Often it's the kind of thing you can rent for $10 a month, or even run at home on an old computer you have lying around if you want.

For example, Friend Camp costs about $30 a month to run, it experiences about 10 minutes of down time each week, and it takes about 2 hours a week of my own time to maintain. I maintain the computer that Friend Camp's software runs on. I'm training another camper to be an administrator as well, so if something happens to me there is still someone who can make sure the site keeps going. The site is funded by campers who can afford to throw a few bucks a month at a Patreon that I run just for us. It's a small price to pay for a nice place to talk to friends on the internet.

As a result of these economic realities, Friend Camp does not have the annoyances of big social network sites. We sell no data, we collect no extraneous data, there are no advertisements at all, and no major features get changed unless I talk to the campers about it first.

Another result of these economic realities is that we're never going to grow much bigger than 50 active users. But that's okay, because it keeps costs low and we can still talk to people outside of our server.

You might ask: how can one small computer replace something as huge as Facebook in my life? Well, the reason is that you control this one computer for your site, and then that computer talks to thousands of other computers, which gives you access to literally millions of people and organizations and videos and all sorts of other things. So what this does is replace thousands of computers owned by a single company with thousands of computers owned by thousands of different people.

You can modify the software

You don't have access to the code and the algorithms that Twitter uses to run their service. If I wanted Twitter to make it so that the word "tomato" always appears in bright red, I would have to submit a request to Twitter, and then they would laugh at me, and that would be the end of it. But on Friend Camp, I have the ability to change our code to do exactly that.

You can also hire someone to change your code if you can't do it yourself. Or encourage someone in your community to learn to do it. It involves work, but the point is that it's possible, whereas it's impossible to have access to the Twitter code unless you are an employee of Twitter.

A silly example of this kind of rule in action is Dolphin Town, a fully functioning social network site that I started a couple years ago. Like Friend Camp, it's a modification of Mastodon. It behaves just like Mastodon with one important change: the only letter you are allowed to use is "E".

Dolphin Town is based on the great oulipo.social, a social network where you can't say the letter "e".

The reason I can make these modifications is that Mastodon is *open source*. There are a lot of things that open source software enables, but for our purposes the important thing is that you can open up the software, tinker with it, and put it back together in its newly modified state.

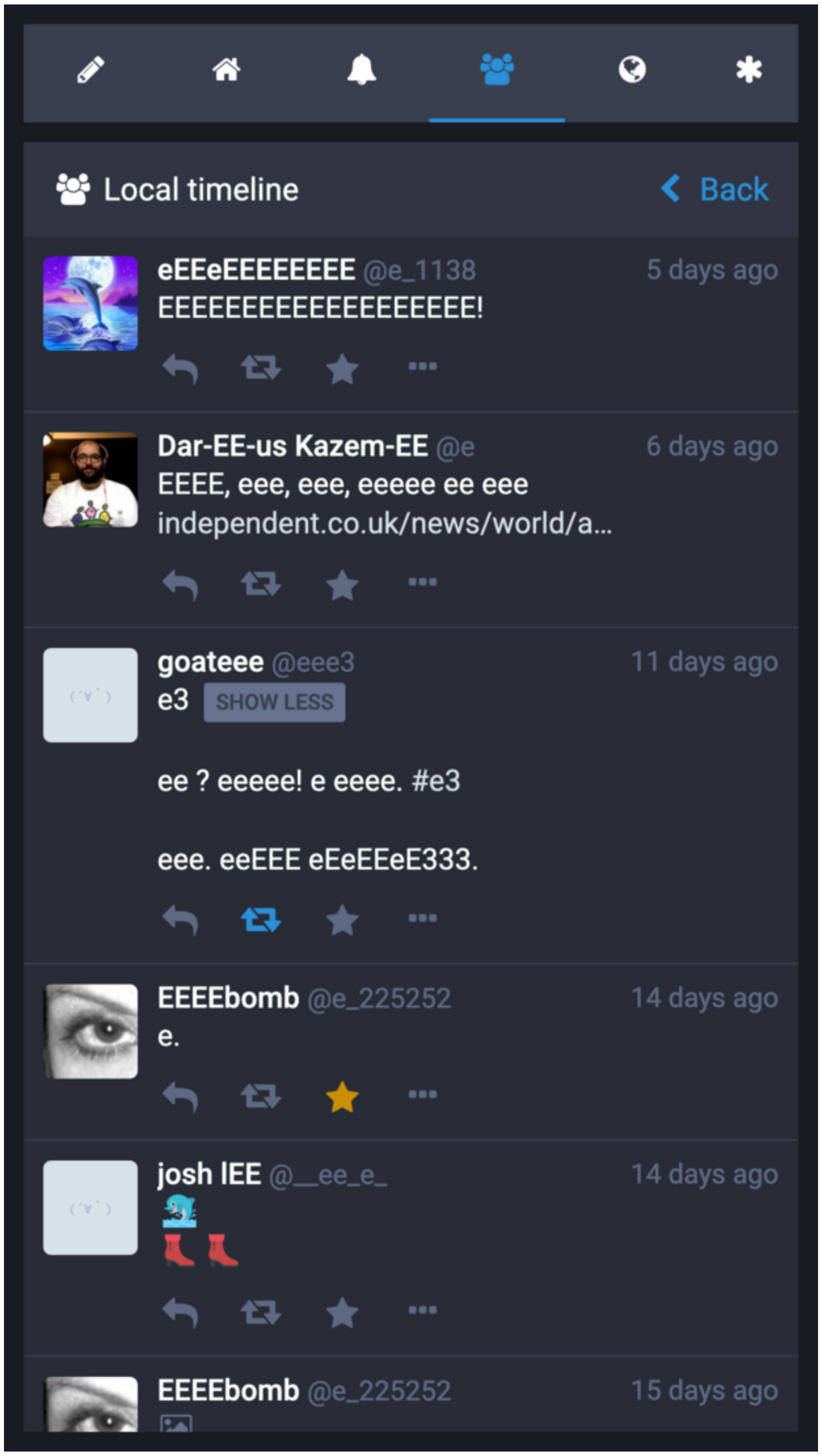
I made Dolphin Town to draw attention to something that I think a lot of people miss: if you run a social network based on open source software, you can fundamentally alter how it works to your liking, at least for the version that runs on the computer you own.

You get to make the social rules and policies

On a big site like Facebook, if you encounter content that you think shouldn't be there, you need to ask Facebook to take it down.

On a small site serving a small community, if the content comes from inside your community, then you can navigate it the way you navigate problems in small communities in the real world. You bring up with the rest of your community that someone is behaving in a way that is counter to the values of the community. It's unpleasant, but at least you have some control over that process, and it's something tangible that you can navigate.

It's more work than simply letting an army of underpaid and traumatized moderators handle all of that for you. But it's worth it. (You might sense "it's hard work but it's worth it" as an ongoing theme.)


Local timeline

Back

eEEeEEEEEEEEE @e_1138 5 days ago

EEEEEEEEEEEEEEEEEEEE!

Dar-EE-us Kazem-EE @e 6 days ago

EEEE, eee, eee, eeeee ee eee independent.co.uk/news/world/a...

goateee @eee3 11 days ago

e3 SHOW LESS

ee ? eeeee! e eeee. #e3

eee. eeEEE eEeEEeE333.

EEEEbomb @e_225252 14 days ago

e.

josh IEE @__ee_e_ 14 days ago

🐬
👢👢

EEEEbomb @e_225252 15 days ago

Figure 3: Dolphin Town in action.

Wait, who is "you"?

When I refer to "you" here I mean "your community" and "your decision making process". Some communities may have a single person who is kind of the "benevolent dictator". Other communities may run based on consensus or even more formal processes.

Some servers are formal cooperatives that use cooperative decision making tools like Loomio to organize their node, which has its own advantages and drawbacks that I won't go into the details of here. Tools like Loomio are important when your group gets to a size that is too big to manage informally. This is why I am very much in favor of keeping these sites really small, like way less than 100 people. It allows decision making to stay informal and means that running the site can be a hobby or a low-intensity community project instead of a full time job.

And of course nothing is going to stop a corporation from running one of these sites either! But I don't recommend joining a corporate site; these things will eventually be driven by profit motive to replicate the same bad experience you get on the current social network sites that you hate.

Why run a small social network site?

The main reason to run a small social network site is that you can create an online environment tailored to the needs of your community in a way that a big corporation like Facebook or Twitter never could. Yes, you can always start a Facebook Group for your community and moderate that how you like, but only within certain bounds set by Facebook. If you (or your community) run the whole site, then you are ultimately the boss of what goes on. It is harder work than letting Facebook or Twitter or Slack or Basecamp or whoever else take care of everything, but I believe it's worth it.

Let's go back to Friend Camp. While there are a hundred thousand people we can talk to from Friend Camp, there are only about 50 people with an active Friend Camp login. We call ourselves "campers" because we are corny like that. And campers have a special communication channel that lets us post messages that only other campers can see.

If I make software that makes the lives of 50 people much nicer, and it makes 0 people more miserable, then on the balance I think I'm doing better than a lot of programmers in the world.

Because we're mostly all friends with each other, this extra communications mode is kind of like a group chat on steroids. For our community it ends up being a sort of hybrid between Twitter and a group chat. As a result of having a community layer alongside a more public layer, we have a movie night, a book club, and a postcard club. Campers visit each other when we travel, even if we've never met in person before. We correspond with each other about what we're making for dinner and trade recipes. They're the kind of mundane interactions that you probably don't want to have with perfect strangers but you cherish in a group of people you care about.

We are also able to have moderation rules that are hyper-specific to our own values as a community. It lets us maintain an environment that's far more pleasant than you find on most social media sites.

And I can make custom features at the request of the people on the site. So if someone wants a particular kind of visual theme, or content filter, or a button to do something that the software can't do right now, I can make it happen, at least within the limits of my own ability.

Software can be complicated where necessary

There are certain things that are notoriously subtle and complex to communicate to users passively via a user interface. A classic example is security settings: Facebook, for example, offers a lot of decent security options but it can be overwhelming to many users. Of course, by having reasonable defaults you can get past this problem, but any default setting is at best going to work for some but not all users.

For example, Mastodon has four levels of privacy available for your posts, ranging from "a private message to another person" to "anyone on the internet can see this". According to some discussions I've seen, one reason that Mastodon has been reluctant to add a "local-only" posting option to its service is that it multiplies that to a combination of eight total privacy settings. The idea is that it would confuse people and they would potentially use it wrong, which could lead to bad security breaches due to user error in dealing with a complex design.

I agree with this in a situation where a user tries out Mastodon and has to learn to use it themselves. But in a case like Friend Camp, I can personally walk the user through what these complex features mean. I can tailor the instruction to their personal learning style and to their needs. Plus, I'm available to answer questions about it whenever they come up.

I'll discuss this more below in the section "What you can do today".

All that said, there's no need for overly complicated software. It should be as simple as possible. But in cases like security and privacy I think the ability to granularly control your settings outweighs factors like ease of use.

You can have hyper-specific norms

Because of their need to have as many users as possible, big social network sites have to try to be everything to everyone. In practice this means they need to limit the number of people they alienate, which means they have to be very careful about what kinds of actions and speech they ban on their network. Every person they ban from their network is another set of eyeballs that could otherwise be looking at advertisements and making them money.

In the end, this means everyone is going to be unhappy with the policies of a big site. You can see it right now on Twitter and Facebook: people on the left complain that these sites support right wing extremism, and people on the right complain that these sites support left wing extremism. And for a website with thousands of posts a second from hundreds of millions of potential sources that's aiming to please as many people as possible, both of these viewpoints are absolutely going to be true. There is no way to moderate this effectively, and you can never come up with rules for behavior that make millions of people happy.

But on a small social network site, you can set hyper-specific norms. For example, Mastodon allows for "content warnings", where you hide content behind a text tag that you define for your post. It's often things like "nsfw" or "mental health discussion". This acts as a warning so people who don't want to read about that kind of stuff won't click through to see it. On Friend Camp, we generally tag anything related to United States politics as "uspol" — we didn't invent this tag, and lots of other servers use it too. I couldn't ever imagine people tagging politics tweets on Twitter, since Twitter styles itself as a news network, for the broadest possible interpretation of the word "news".

You also don't need to implement some kind of draconian filtering system to have norms: you simply model the behavior you want to see, and remind people to act a certain way when they post outside of those norms. This kind of thing does not work well in a forum of even 200 people, but in the 50-ish range it is entirely doable.

Your small social network site can have its own rules about, for example:

- what speech is acceptable
- what actions are considered violence
- what actions are considered protected speech or expression

This will be very different for every site and I think that's part of the beauty of this whole thing. I talk more about the specifics of this in the "What you can do today" section below.

Why to NOT run a small social network site

There are many reasons not to run a small social network site. As I've mentioned before and will continue to mention, it's really hard work. You need to at least have time for a hobby or figure out a way to get community support for doing it (maybe people chip in to keep things running; maybe you can trade in-kind services with your community members; etc.).

If you choose to run a site like this, it means that people will now depend on you for something that's important to them. People on Friend Camp are very forgiving when we have down time, and sometimes that down time can last hours if I'm asleep or at a movie or otherwise busy. Later in this document I discuss some cases where people had to abandon a site because it got too big and unwieldy to run.

Friend Campers are, of course, incredibly patient and gracious and understanding that I'm a person with a life.

But none of this is unique to running a social network site. I think all of this stuff applies if you take away the internet and computers entirely. This is exactly the cost of starting a community theater group, activist group, social group, sports club, book club, and so on. Even the bit about "down time" applies if you run a physical space like a theater. You can't be there for every performance but if something goes wrong with the building, you'd better have a plan to deal with it.

In other words, the costs of running a small social network site end up being more or less the costs of starting, well, anything involving other people.

What you can do today

This section describes practices that anyone running Mastodon or similar software can do today to make for a cohesive, rewarding, and caring community. This section lays out some rules that have helped make Friend Camp a good place. The high level overview is:

- keep the number of users on your server very small
- remember that your job is social first and technical second
- provide custom onboarding
- provide group activities
- provide custom features if you can
- enforce your code of conduct

Keep it small

It is absolutely necessary to have no more than 50 to 100 active users who log in to your online community.

This is a number I am pulling out of my ass. And eagle-eyed readers will notice that I'm not even settling on a single number here. I know that 50 is a number that works as a cap, and I think anything more than 100 definitely does not work as a cap.

In my admittedly limited experience as a human being on this planet, you are not going to be able to form and maintain a cohesive group of more than about

50 to 100 people who all basically agree on values and moderation rules and that sort of stuff. Even as a group where people are mutually looking out for each other, you are not going to be able to meaningfully make sure everyone in that group's wellbeing is maintained. I don't have science to back this up. I have seen this happen in startups, in activist groups, and in internet forums.

In the world of startups this is sometimes referred to as the problem of keeping your culture as you grow. I often hear people lament, "I miss when our company was just ten/twenty/fifty people."

I can't stress enough how important it is to keep your numbers small. To draw from recent history, witches.town was one of the more popular Mastodon servers in 2017 and 2018. In mid-2018 there were a series of disagreements between the primary administrator and the users. I can't speak to the nature of these disagreements as I wasn't there but there were multiple disagreements about moderation policy, and also the primary administrator (who I believe actually owned the server and its domain) said they were burned out on running the server. In the end, the site was shut down. According to archive.org it had about 2400 registered accounts in April 2018, shortly before the instance was deleted. If we look at instances of similar size today we can extrapolate that there were perhaps 500 active users on the server at the time it went dark. KNZK is another network site that shut down on June 30th 2019 and it had about 3100 registered accounts and 560 active users. It shut down due to maintenance tasks demanding too much time and effort from administrators.

I posit that 500 active, invested community members will not be able to achieve a values-based harmony or consensus. It's simply too big to be possible. My assertion is not backed up by any studies I have read but rather my personal experience in online and offline groups of all kinds. You cannot wrangle consensus from 500 people. With that many active, committed community members you will necessarily have at least a few dedicated members who feel investment and ownership in the community who are also extremely unhappy with the direction of the community.

People like to bring up Dunbar's number when I talk about this stuff, and I have no idea if that concept is a crock of bullshit or what. For starters the number itself, 150 people, seems a bit on the high side to me intuitively. But if you believe in Dunbar's number then that does lend some credence to this idea.

In addition to promoting group cohesion, having a small number of active users means moderation is completely achievable by a single person. With 50 active users I field, on average, one moderation request a month. This is in addition to me being vigilant and blocking bad actors as I see them. Now, this wouldn't be true if I had 50 random people who all signed up. But these are 50 users who are screened and onboarded into our community.

Even tricky matters like delineating what is and is not acceptable speech become easier when you're dealing with a small community. For example, there may be some inalienable right for expression of disagreeable political speech, but if it is speech that all the people on your corner of the network agree you don't want to see, you can simply ban it internally and filter or block it externally.

And when your community is small, you can use powerful tools that would be irresponsible to wield in a larger community. For example, when I discover a user on someone else's site engaging in speech that violates our local norms, I will often mute or ban the entire server that hosts the user in question on behalf of all the people on Friend Camp. This would be inadvisable on a huge network like Twitter, but on Friend Camp, we have all agreed we don't want to see certain things and we don't want to engage with other servers that allow for those types of speech.

Of course, the question is how does this scale beyond those 50 people? Well, that is why I'm writing this guide. I think there should be thousands of these small servers talking to each other.

You are the party host

Running a small social network is like hosting a party. It requires social intelligence, empathy, and yes, technical skills.

You might wonder what aspect of hosting a party is technical. I mean "technical" in the sense of "requiring technique and skill". Party hosts need all sorts of technical skills: managing RSVPs and calendars, planning and purchasing supplies, cooking and serving, wrangling a music playlist, etc. Not everyone is comfortable hosting a party because these are technical barriers to entry. Similarly, not everyone is comfortable running social network software because of technical barriers to entry.

But as with hosting a party, technical skills aren't enough. You need to have social skills, or have someone assisting you who does.

For example, I need to maintain an understanding of everyone on the server. I have to remember what their individual needs are, which means that I need to be able to hold all of that information in my head. It's not a task that everyone is suited to, but it's critically important.

Similarly, programming computers is an important task that not everyone is suited to do. The big difference is that when it comes to anything dealing with software we tend to overestimate the importance of programmers and underestimate the importance of everyone else.

Among other things, my duty is to read the local timeline and catch up on everyone's posts. I need to take the temperature of the network and also provide social lubrication where necessary. ("Hey, you just mentioned you're traveling to Montreal! So-and-so lives there, maybe you should message them and see if they want to meet up.") Catching up on all local posts would take forever with hundreds of active users, but with dozens, I only have to read the timeline for 20 minutes a day.

Do not fool yourself into thinking that your job as an administrator is primarily technical. It's social first, and technical second. If you want to only focus on the tech, then please find someone who cares deeply about the social organizing side and recruit them as a co-administrator.

Provide a custom introduction to your network for every user

Because you only have 50 people in your community, you are able to do all sorts of things that would never be possible in a larger group. One really important place where this comes into play is in new user onboarding.

When a new user joins Friend Camp, I now schedule a video chat with them (or an in-person chat if they are local). For one to three hours, I introduce them to Friend Camp and walk them through our basic features. I customize the content of the chat with them depending on their background. Generally we discuss the following topics:

This does not have to be a video or in person thing. I think video works for many, but there are certainly all sorts of people who can't do that kind of interaction for all sorts of reasons. My recommendation ultimately comes down to: pick a form of onboarding with a personal touch, where you can convey a lot of information and answer questions in a way where the person you're onboarding feels comfortable asking questions.

- the basics of using Mastodon, tailored to the individual's needs
- how to find further help if they need it
- who *everyone* on the server is (I go down the list and explain in one sentence who every non-private user is, kind of like a party host might introduce guests to one another)
- our custom software features
- our group traditions and how to participate in them (movie night, for example)

If I know them to be social media power users, I'll spend more time on those kinds of features. If they are not very technical and need more basic training, then I spend more time on that. I'll ask them what kind of phone they use, and give them specific information about iOS or Android as needed.

In addition to explaining who everyone else on the server is, I point out specific people who I think they might get along with. Again, it's like being a party host. If you know people in common or have common interests, I will mention that.

This is one of those things that a small social network site can provide that a big site could *never* provide, because it just doesn't scale.

Group activities

It's important to do things together as a group. Organize whatever makes sense for your community, and allow your community to organize as well.

For example, on Friend Camp we have: movie night, book club, postcard club. Not everyone participates in these things, and that's okay too. But the option is there. The people on your site will probably want different activites than these specific ones, but the important thing about this is we are connecting through some kind of shared activity.

Keep it interconnected

The ultimate goal of the last three sections (hosting a party, custom onboarding, group activities) is to socialize everyone on your server with everyone else on your server. Not everyone has to get along with or even interact with everyone else, but the more positive connections you build between people on your server the better a place it will be. Having a tight-knit group where lots of people know each other is key to the principle of maintaining hyper-specific norms.

An interconnected network also helps with conflict resolution, so when things go "wrong" it's hardly noticeable. As one camper put it to me:

> "Tiny scale conflict resolves between people without formal moderation. Alice says something that Bob didn't like, Bob points it out, Alice agrees and apologizes, and everyone moves on."

That kind of conflict resolution doesn't easily happen between strangers, even in a group where people share values with one another. But when the chance of any two people being complete strangers is close to zero, this happens way more frequently.

Funding

Unfortunately, running a social network node requires you to move electrons around on pieces of metal, which means you need metal and electricity. And to have access to those resources, you need some kind of funding. Either you have a benefactor, you get together as a group and buy some equipment and internet service and pay for repairs, or pay a company like Linode some money to host a server for you.

We have a Patreon for Friend Camp simply because it's the easiest way for me to collect monthly donations. Campers contribute what they want or what they can. I needed to upgrade our server at some point for various reasons and I made an appeal to the community. I ended up getting the monthly pledges to cover the new server costs for everyone. The monetary costs of running Friend Camp are laid out in this spreadsheet. At the moment, Friend Camp costs $31.00 a month to run. This is not a small cost for a single person, but spread between 50 people it can be very affordable, more like $1 per month per person even if half the active users can't afford to contribute anything.

The bus problem

For almost a year, Friend Camp had one person with the keys to the kingdom: me. In this first year, if I got hit by a bus, the server would probably run until its first crash and then never come back again.

I decided to put out the word to see if a member of the community would be able to volunteer to be co-administrator. Thankfully, one of our own decided to take on the responsibility. I was lucky that there was a person in the community willing to do this and with development experience and experience running a server. It took just a handful of video calls to get them up to speed and within a month they had pushed their first production upgrade to Friend Camp.

Not every community is going to have people who are willing and able to take on these duties. But if you have someone willing, there could be a longer mentoring process to get them to the point where they are able to do at least minimal work to maintain the server in the absence of the main administrator. If someone is at least able to reboot the server if things go weird, that's a huge step up from nothing.

Ideally you can train a willing person to the point where they can do more complicated administrative tasks. If you have a small social network community that actually really means something to you, it's work that helps keep a thing alive that people deeply care about.

Code of Conduct

A code of conduct (CoC) is about what actions are acceptable on a network, which is absolutely necessary but not sufficient on its own for establishing a culture. A CoC perhaps implies certain values but it mostly states what is NOT acceptable rather than what IS encouraged. Some codes of conduct do state things like "please engage with people courteously" etc.

I think a good CoC should be specific enough that it is actively repulsive to some people. And frankly, the more people who are repulsed by it the better. Perhaps simple anti-racist sentiment is enough to repulse a small number of explicitly white supremacist people, but if you add in sentiment like "no TERFs allowed", suddenly a lot of people who consider themselves leftists will also decide not to join. As an example, Friend Camp is anti-free-speech, at least in the sense that freedom of speech is commonly understood as a value. This is repulsive to some people on both the left and the right, and it's important that people with that core value find somewhere other than Friend Camp to set up their online home.

For those wondering, the RationalWiki definition of freedom of speech is exactly the kind of freedom of speech we take issue with on our server. Again, you may find this abhorrent, but you don't have to join our server, and we're never going to house more than a few dozen people.

Again this is a *good thing.* I remember one code of conduct I read for a chat forum where they stated that "before posting here, ask yourself if what you are saying is necessary for the health of the discourse". And I saw that and thought, "Wow, I would never feel comfortable posting on that forum because I am always doubting the necessity of what I have to say." This is good, though. The forum clearly signaled that they wanted a certain kind of person, one who is confident in the necessity of their speech. And that is not me, and I was able to steer clear. Did I feel excluded? Sure. But ultimately that feeling of pre-emptive exclusion is better than joining a community and THEN realizing that I don't belong there.

I believe this rule was based on "The Three Gates of Speech", which is an idea that has been around for a long time that I think has some serious problems in practice as rules for engagement.

[Enforcement]Enforcement

Valerie Aurora and Mary Gardiner have published a free book called *How to Respond to Code of Conduct Reports* that I think is required reading for anyone with a code of conduct on their site. The book explains what a CoC is *and what it is not* and how to make yours actually effective. Some of their recommendations make more sense for very large communities (like conferences of thousands of people) than the small ones I'm advocating for, but most of their recommendations do apply, and the book comes with many real world examples that you can learn from.

One reason I'm a big fan of adding a local-only communication layer to these social network sites is that it helps with enforcement. The reason is that it creates serious social repercussions for violating rules to the point where you're kicked off of a server. This is because the person who was kicked out cannot just join a new server and re-follow everyone and continue to participate in or monitor conversations. It's more like being kicked from a group chat. It can be socially devastating, which means there are real consequences to violating group policy.

At the moment, Mastodon does not support a local-only communication layer, though modifications like Friend Camp do.

I'm sure there will be friction on Friend Camp but nobody has, to my knowledge, broken our code of conduct or otherwise required admonishment or disciplinary action. When that does happen and it is beyond a simple warning followed by an apology, I will individually solicit opinions from literally every active user on what should be done. Because we keep the server small, I could in theory host video calls with every single user where we discuss what ought to be done. I'm not sure if that last one is a good idea but I have that option, which you can't say for Facebook. This is all possible because we are a small group of people and we will never be a big group of people.

Technical recommendations

This section might not make a lot of sense if you're not a programmer. Feel free to skip it and move to "What we need to do in the future" if that's not you.

Ideally you will pick social media software that has features which allow for communication that only your users on your server can see. In the case of Friend Camp, we maintain Friend Camp as a fork of Mastodon and pull in Renato Lond's great work that allows for posts that only the 50 people with logins to Friend Camp can see. I would go so far as to say this is a necessary feature for group cohesion, and my hope is that implementors of decentralized social media software come to understand it's important.

If you are a programmer, I suggest that you maintain a fork of whatever open source social network software you end up using so that you can implement custom features for your users. You might end up making purely aesthetic changes, small browser hacks, or complicated new features that you need to maintain version to version. If you implement custom features that your users ask for, it gives them a stronger sense of ownership over the site even if they are not the ones writing the software. Make sure to credit them somewhere too! I try to credit individual users for each suggested feature in the commit messages and also on our server's About page.

Provide multiple services for your community members! Probably not email because that is a nightmare but there are other possibilities. I use Mastodon's login to authenticate small helper applications that I write for my users. I just put in a hard coded gateway so that only people from my server have access to them (so their OAuth login account has to originate from the server friend.camp). This means the basic social media login for your server is also a login for many other services; kind of like a default LDAP but not.

[Examples of services you can provide]Examples of services you can provide

I only have one auxilary service I provide my users that gets active use but I think it's a big one. Bugle is a bot that any Friend Camper can use that sends a direct message to *every other person on the server* (unless they have blocked it or have DMs otherwise disabled). This would normally be a horrible idea on a site like Twitter. But because we're a small community and we all know each other, I trust that my users will use it well and sparingly. If it is misused I will give the user a stern talking-to.

A stern talking-to is a laughable deterrent for most sites but it works for us because we are so tight knit.

What we need to do in the future

The current situation for small social network sites is not even close to perfect. There is work that needs to be done to make this idea better and more viable for more kinds of people.

Places where we need better tech

Most of the problems that exist are social problems with social solutions, and I've tried to lay some of those out above. But there are still unsolved problems where better tech could really help, so I'll try to enumerate those here.

[Fluidity of identity and the ability to migrate]Fluidity of identity and the ability to migrate

The existence of a server and an administrator implies some local form of centralization. I think this is necessary because most people don't want to run their own network node, and there are fantastic benefits to having a trusted local administrator. That said, the drawbacks are also great and we need to be able to mitigate the drawbacks.

What happens if I, as the administrator, violate the social norms of my own community? In other words, *what happens if the person with all the real power is the problem*?

People *need* to be able to jump ship and migrate their accounts, seamlessly and wholly, to other servers. We do not have a good technical solution for this yet. In my opinion it is the one big area regarding federated social networks where we need to work on technical solutions.

If I decide that my values no longer align with a server, I need to be able to take that account to a new server. If the server decides to kick me off, that's fine, but I should *in theory* be able to set up shop elsewhere.

I know that people have been working on this issue, but as far as I'm aware it's nowhere near resolved. As it stands right now, the fact that you can't uproot and take your stuff with you from one site to another means there is a very real kind of lock-in. This helps with enforcement (as I said above) but is a huge liability when it's simply the choice of the individual to move.

[Let people keep things in the community]Let people keep things in the community

Most open source social network software right now is designed to get your messages out to as many people as possible. We need support for private communities. For federated social networks this means support for messages that don't federate and can only be viewed by people with access to the server on which the message was posted.

Unlike the identity problem, this is a very easy thing to implement technically. It already exists on a minority of open source social network servers. But the big players don't support it and that's causing more harm than good.

There is currently a pull request open on Mastodon by Renato Lond that implements exactly this feature. It's derived from glitch-soc, a Mastodon fork.

[Server forking should be easy]Server forking should be easy

It should be easy for, say, 25 members of Friend Camp to pick up and start "Friend Camp 2". This could be because Friend Camp is getting too big, or it could be because these people don't like what Friend Camp has become and would like to move en masse.

As far as I'm aware there is no work at all being done on this issue, but I suppose a prerequisite to this is a solution to the identity migration problem above.

[Lean software that doesn't have to scale]Lean software that doesn't have to scale

What if we built software to run on very low spec, low power machines, that was a federated social media server for 50 people but could never grow to support more than that? You could use something like SQLite instead of a "real" database because you'd never realistically have to support a lot of writes to the DB. You could run on a raspberry pi.

There are some federated servers that fit this bill already. Pleroma is a Twitter style server that extremely light on its use of resources. And Write Freely, a federated Medium-style blogging server, is incredibly lean as well.

But even Pleroma and Write Freely are built with thousands of users in mind. What kind of low tech solutions can we enable if we keep our communities intentionally small? This may open up more paths to equitable access by communities with different resources available to them.

More on scale

Any time I propose a new piece of software to a group of software engineers I'm asked the same question: how will it scale? We are trained as a group to ask this question. I think it's the software equivalent of in manufacturing when someone asks "What will it cost to produce?" Since the marginal cost of producing software is effectively zero, it's the scale, the ability for the software to be used by millions or billions of people, that becomes the limiting factor that everyone brings up.

Imagine two different software developers. One person writes a piece of software that makes the lives of one million people slightly easier. Maybe it's better routing for navigation software and it shaves 30 seconds off the commute of a million people. Another person writes a piece of software that only ten people ever use, but it tangibly changes their lives for the better in very material ways; maybe they learn a trade that becomes a career.

One of these outcomes is not necessarily better than the other, and yet due to myriad factors, only the software with a million users is likely to get funding from entities—whether the context is for profit or not for profit.

I'd like to advance the notion that software does not have to scale, and in fact software can be better if it is not built to scale. I hope some of the examples I've given above have illustrated what is possible when software is used by a small number of people instead of a large number of people.

Beyond local and public: the neighborhood

Right now on federated social networks you have a concept of people on your own "home" server (which I'll call local) and people on every other server in the world (which I'll call public). But we need concepts that are more fine grained than local and public.

I would like to see groups of servers that band together through a kind of mutual approval system. The group of people on Server A decide that Server B is to be trusted, and vice versa. They approve each other, in a manner similar to friending someone on Facebook, but for a whole server instead of an individual user. Now the 50 people on server A and the 50 people on server B are in a "neighborhood" together.

Servers that belong to the same neighborhood could share things with each other. For example, the servers could share access to posts that are tagged on either server as "available to the neighborhood". Servers could share block lists, since mutual trust between servers probably implies some level of shared values between the people on both servers.

A neighborhood would necessarily consist of two or more servers, and could in theory grow to be very large.

However, I don't think neighborhoods should be very big themselves. I think there should be a kind of mutual decision-making, so perhaps now that A and B are connected, both A and B have to agree that C is worth connecting to, and C has to agree that A and B are worth it. This makes every extra node you add more difficult, which is the point. Big "private" communities should be very very hard to come by — and "public" should be where the "broad" conversations happen.

I also think there shouldn't be concepts of overlapping Venn diagrams of multiple neighborhoods. I *loved* the concept from Google Plus of circles. The idea was that as a person you have friends, coworkers, college friends, family, IRL neighbors, etc. People you "know" on Google Plus could belong to none, one, or more than one of these groups. A childhood friend who you also work with would probably belong to "friends", "childhood friends", and "coworkers". This system sounded *great* to me… until I tried it in practice. And in practice it was so much tedious digital paperwork to keep it all fresh and updated. I didn't want to manage people into one or more of dozens of categories and in the end I just went for the default "mutual friends on the social network" and "public" circles.

I think many of us can hold three of these groups in our head though, especially if it's not the job of the individual user to constantly maintain it. I'm sure lots of people can hold more than three groups in their head at a time, but I'm picking the smallest useful number in order to reduce the confusion I experienced with Google's "circles" feature.

So for me an ideal network would be partitioned into three basic levels for posting:

- local - just the people on my physical server
- neighborhood - the concept I described above of people who belong to servers that have a mutual trust agreement
- public - everyone in the world

This would be in addition to the usual layers of "followers only" and that kind of thing.

One way I see this breaking down is:

- local - tin foil hat stuff, "I do not want this message to leave this piece of metal", or perhaps the *very* intimate like "only local mutual followers get to see my nudes"
- neighborhood - most of your friendly chatter and ideological debate happens here
- public - the place for things that are inconsequential (cute cats and jokes), or require signal boosting, or where you really want to put an idea in front of the public

Conclusion

There are a lot of unanswered questions here, and in a lot of ways I am still learning as I go. I don't want to give the impression that I have the answer to everything so I'd like to address some questions that I don't have good anwers to at all.

Hard questions

How do people join one of these networks if they don't have friends who are already on such a network? This may be where sites with thousands of users come into play. Perhaps people join a giant server but then do it with the understanding that they are there to make friends on smaller servers with the idea of eventually moving there or even starting their own server. This is not very feasible until the account migration stuff I mention in the "What we need to do in the future" section is solved.

What happens to people who suck? Like what about people who are actively opposed to community cohesion? Will they find a sort of anti-community where they are accepted? Will they simply migrate to the giant networks of thousands or hundreds of thousands of people? Will they set up their own single-person networks? Will this kind of network even be the place for them at all? I don't know.

What about equity? For example, how do we get these servers run and operated by people in poor communities? I look to work by people like Bruna Zanolli, going in to Amazon indigenous communities and training the women to run the networks. If you have the privilege to run one of these sites then you might also have the ability to teach people how to run them for themselves. If that's you, then it's really your duty to get other people up and running. The more communities you help get started, the more options your own community will have for people to connect to. I also would love to see granting organizations support this kind of work.

Do the technical administrators have to be the same people doing the social organizing? I think the answer as of June 2019 is, sadly, yes. If you have 2 people with root access to the server and 2 people managing the community aspects, you'll end up with imbalances in that group of 4. You will end up with technical administrators who feel like code monkeys who never get the gratitude that the community organizers get, or you'll end up with community organizers who feel like glorified babysitters while the techies have all the real power. You might even end up with a situation where both are true. I think that if you're dedicated to this sort of project though, you could start with something like that 2 and 2, and then the techies could teach the organizers the technical skills, and the organizers could teach the techies the organizing skills.

However, if more paid hosting services would support a wider variety of software and forks and features, this reliance on technical people like me will hopefully become lessened and an actual community organizer, or some kind of community organization structure, can be in charge.

So what do you, the reader, do next?

What you can do next depends on who you are.

[If you are comfortable with servers and hosting]If you are comfortable with servers and hosting

If you've done the system administrator thing, you have many options here.

Ask your friends if they'd like to try out a social network run by you. If you can find 5 people who would be interested that's enough. Then get going! There are plenty of technical guides out there:

- Setting up a Mastodon server for a Twitter-like experience
  - This also works with the various forks, like Glitch Social and Florence
- Setting up a Pleroma server for another Twitter-like experience
- Setting up Pixelfed for an Instagram-style experience
- Setting up PeerTube for a YouTube-style experience

There are many other servers and services out there as well, but these are the ones that I personally have used and can recommend.

Keep in mind that **none of these services support any of the "future" concepts like neighborhoods**. Some of the forks like Glitch Social do allow for local-only posting.

[If learning the tech stuff is realistically not going to happen]If learning the tech stuff is realistically not going to happen

Do you have any technical friends? If you do, ask them if they might be willing to do this sort of thing.

You can also pay a company like Masto.host to run all the technical stuff for you. If you take my advice and only have a small number of users, then you can probably get away with paying less than $10/month for your hosting. Paying a company necessarily means that you have less control over your system. You won't be able to provide custom features for your users, but if you are not a technical person you probably aren't going to be writing custom features anyway.

I have no business relationship with Masto.host but I have spoken to them about the possibility of getting modified versions of Mastodon that include things like local-only posting onto their hosting service. I can't promise anything on this front except that I am working toward it.

Running a small social network site is completely possible to do on the side

I suppose I'll repeat what I said multiple times in this document, which is that running a small social network site for your friends is hard work, but it's worth it. It is first and foremost the work of community building, and only secondarily is it a technical endeavor. And it's completely possible to do, today, though depending on who you are and what your resources are it's going to be difficult in different ways.

Acknowledgments

First and foremost I'd like to thank the Mozilla Foundation and the Ford Foundation for the Mozilla Fellowship that's kept me funded for the last ten months. I would not have been able to write this document without dedicated time to think about these problems.

I especially want to thank the Mozilla Fellows in my cohort, who have been excellent listeners and collaborators for these past ten months. Code for Science & Society, who are the best possible organization to be embedded with for the work I'm doing. The people at ScuttleCamp and Eyeo Festival, who provided me with thought-provoking conversation and gave me a semi-public forum to test out some of these ideas. The various fediverse and ActivityPub developers who have given their time to answer all sorts of questions I've had. And everyone who's met me in the last ten months and let me talk their ear off about this stuff.

I also want to thank the folks at Ink & Switch for publishing their local-first software article, which got me thinking, "hey, I could write an article too."

Thank you to Emma Winston for designing the HTML and CSS so that it actually looks good.

A special thank you to my husband for being incredibly generous and patient as I've flown around the world talking to people about this stuff.

And last but not least, thank you to the Friend Campers, without whom none of this would be happening. I couldn't ask for a better petri dish.

About the author

Darius Kazemi is a 2018-2019 Mozilla Fellow who lives in Portland, Oregon, USA. He cofounded Feel Train, a creative technology cooperative. He makes weird internet art, writes open source software, gives talks, wrote a book about niche videogame history, is very serious about karaoke, and is blogging about the first 365 IETF Request For Comments (RFC) documents.

License

Older versions

You are currently viewing version 1.1 of this page, last updated July 9 2019.

- Version 1.1: added an aside about video onboarding. July 9, 2019.
- Version 1.0: original publication. July 8, 2019.

Back to top