

# The Network Times

Ian Turner

June 27, 2024

# 1 Full Stack Rust Rewrite Loading

i love leptos so this app will be my way to become cracked at rust. it's in different [git repo](#) btw (too lazy to update readme in other repo)

## 2 Bounty Submission for Balajis on Farcaster

balaji put out a [cast](#) on wednesday the 6th challenging anyone to create an AI NYT. he provided a nice example of something working [18 months ago](#) with inferior LLM models. i've been looking for a reason to stay up all night coding, so i figured i'd give it a shot.

### Why $\text{\LaTeX}$ Ian?

great question anon! i write these notes everyday at work and on other projects since pmarca gave me the idea with his anti-todo list concept. blog is no longer on the internet or else i would link it here. this may be completely useless to people, but i don't really put much detail in my git commits (i should start) so maybe this will help if you end up contributing.

**p.s.**

weeks are in reverse chronological order btw (and the weeks are not sequential, nor are they full weeks). i'll add a TOC eventually...

### **3 Week 10?**

**Monday, 06/24/2024**

- working on hubble endpoint replication for casts.

**Tuesday, 06/25/2024**

- fixed dates on this document (not really but listed this month as july for a while, weeks still way off).
- riced my color scheme mostly yesterday (it fixed me), so will just continue with casts endpoint.

**Wednesday, 06/26/2024**

- i am nvim maxing rn, playing around with octa to create and merge prs from nvim. it is so slick and updates on save in real time since it uses gh cli.
- really just procrastinating, but at least prs will be fun now.

Thursday, 06/27/2024

- i realized that i should have deployed this project from the beginning. to help with this blunder i've decided to rewrite my fork of [ishan's engblogs.dev](#) to leptos.
  - this way i can break this other project since it's much simpler for now and doesn't involve farcaster at all.
  - we are back (Figure 1)

- we are back (Figure 1)

**Figure 1:** we back

- i had python woes last i checked on this project so we'll see how this goes when trying to update things haha
  - we are so back (Figure 2

**Figure 2:** we so back

- wow it worked first try, let's gooooo. okay this isn't deployed but I'll do that next. feels good to revive this project as well in like a few hours (i can't search or anything yet, but basic stuff works)

## 4 Week 9?

### Monday, 06/17/2024

- man i can not count, but we are back. leptos rewrite going well, realized i need to use something like **deadpool-diesel** since i already have some diesel endpoints i want to repurpose and apparently this deadpool crate integrates well with tokio (at least better than plain r2d2 from diesel)
- i will play around with this for article storage and more
- looks like deadpool is the move, will explore latest changes to crate and try to implement this over r2d2
- channels still sus as hell the way i use create\_effect, but we will send it for now (i don't know how to use loom crate or similar yet to test async / concurrent stuff)

### Tuesday, 06/18/2024

- i am getting skill issued when trying to convert to deadpool from vanilla postgres r2d2 setup haha. might just stick with what i had before and if the stream fails halfway by user refresh or anything else, that's just tough buddy<sup>1</sup>
- wait, this *is* possible with vanilla diesel, i was doing this months ago since it was somehow easier than keeping track of each chunk in a vector until the end haha
- ...

### Wednesday, 06/19/2024

- finally got deadpool-diesel working and made a ding Figure 3

The terminal window displays the following content:

**deadpool-diesel**  
Dead simple async pool for diesel  
by Alice Ryhl, Michael P. Jung  
<https://lib.rs> Rust pack...  
blocking waiting for file lock on package cache  
Finished 'dev' profile [unoptimized + debuginfo] target(s) in 0.36s  
Compiling nwtr v0.1.0 (/home/eze/dev/nwtr)  
Cargo finished cargo build --package=nwtr --lib --target-dir=/home/eze/dev/nwtr/targe...  
front --target=wasm32-unknown-unknown --no-default-features --features=hydrate  
front running WASM  
Tailwind finished and...  
Failed to parse the request body as JSON: active\_lab: expected value at line 6 column 26eze  
~ \$ curl -X POST http://localhost:3000/api/create\_message -H "Content-type: application/json"  
{"thread\_id": "63429007",  
 "content": "oooo im deadpoolinggg",  
 "role": "user",  
 "active\_model": "gpt-3.5-turbo-0125",  
 "active\_lab": "openai"}  
nwtr# select \* from threads;  
id | created\_at | updated\_at  
-----+-----+-----  
63429007 | 2024-06-20 05:09:37.049123 |  
(1 row)  
nwtr#

Figure 3: *fearless concurrency!*

<sup>1</sup>this will destroy me inside

**Thursday, 06/20/2024**

- added url encoding to allow user to upload code snippets or special characters basically. this should prevent bugs where a cast can have some strange character which breaks the request to create an article.
- we will see what the limits of what i can pass into the url for a get request can be. if this becomes an issue i can always change it up or even experiment with web sockets rather than sse if the complexity is not too crazy.

## Saturday, 06/22/2024

- testing ffmpeg vid and audio
- it works lets go
- will continue reworking the hubble endpoints from react project in leptos components
- might even rework the chat stream component to use server functions properly
- i think in order to get the chat working where messages are actually chained together, i need to rework my `conversations` module.
- since i want to use some structs on the client side, i need create new structs for client only (without any server crates in the ssr block of the `cfg_if` macro), then new structs which share these common types for the server with the database functionality.
- this way the new structs `ThreadView`, `MessageView`, and `NewMessageView` are safe to use client and server side, while diesel specific structs `Thread`, `Message`, and `NewMessage` are only compiled for ssr.

## Sunday, 06/23/2024

- getting absolutely cucked by a simple server function implementation. i straight up can't get them to work dawg, but i want to be fancy and use them so i will keep trying.
- i think i should for sure abstract away the thread fetching logic to its own component, this may be the number one source of my woes recently.
- we are back (Figure 4)

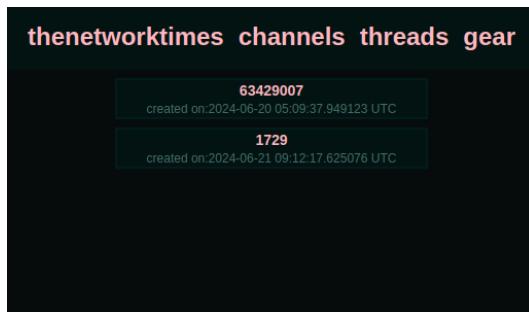


Figure 4: init threads component

- essentially, i forgot that i could import the `AppState` struct like i do in main function to acquire a db connection. the more annoying thing was the error formatting. i needed to create a custom `ThreadError` enum to categorize different types of errors (`PoolError`, `DatabaseError`, `InteractionError`) but they could not be type cast properly. this didn't properly convert them all the `ServerFnError` which leptos server functions must return. unlucky, so i needed to create a `to_server_error()` function which explicitly converts `ThreadError` to `ServerFnError`.
- these problems easier to tackle once i stopped trying to do everything in the `chat.rs` component.

## 5 Week 8?

### Sunday, 06/10/2024

- we might be back (leptos rewrite). notes from testing existing hubble node routes with leptos components.
- this is not live yet since it has very limited functionality.
- `fetch_username` checks if `lead_usernames` already has the `fid`. if not, it fetches the username and updates the state.
- added `ongoing_requests` to track fetches and avoid multiple requests for the same `fid`.
- Used `create_effect` to manage side effects, ensuring usernames are fetched only once.
- Used `spawn_local` for async tasks to keep the main thread non-blocking.
- `Signal` and `set_signal` handle reactive state in the `Channels` component, making sure the UI updates when data changes.
- `HashSet` tracks `ongoing_requests`, preventing duplicate fetches and redundant state updates.
- The component dynamically displays usernames from the updated state.
- asdfasdf

### Monday, 06/11/2024

- removed the `lead_username` variable and its assignment using `unwrap_or_else` from the `view!` macro.
- added closure inside the `view!` macro that matches on `lead_usernames.get().get(<ref>fid)`.
- if the lead username is available show it, else show "chill" in div where username would be
- closure is defined using `move ||` to capture the `lead_usernames` and `fid` variables.

Tuesday, 06/12/2024

- begin rewrite of cast and cast list pages, need to think about using `create_effect` in the complicated way that i did above.
- using this `create_effect` in the way that i am to sync the reactive system is *officially* (Figure 5) discouraged since you might shoot off your foot with an infinite loop if you don't create something to keep track of the ongoing\_requests. i like the chill message though and how it could show up at different times for different channels in some cases (probably won't be used, but we will see).
- i'll do the cast stuff without `create_effect` to see if it's less complicated and achieves virtually the same thing.

```
let (a, set_a) = create_signal(0);
let (b, set_b) = create_signal(0);

// ✅ use effects to interact between reactive state and the outside world
create_effect(move |_| {
    // immediately prints "Value: 0" and subscribes to `a`
    log::debug!("Value: {}", a.get());
});

set_a.set(1);
// ✅ because it's subscribed to `a`, the effect reruns and prints "Value: 1"

// ❌ don't use effects to synchronize state within the reactive system
create_effect(move |_| {
    // this technically works but can cause unnecessary re-renders
    // and easily lead to problems like infinite loops
    set_b.set(a.get() + 1);
});
```

Figure 5: *create effect bad*

## 6 Week 7?

Saturday, 04/20/2024

- Q is helping me with ai nyt server.
- I am hard stuck on messages so perhaps we will crack it together.

Sunday, 04/21/2024

- Got diesel initialized in the project for storing articles, have not yet tested the ORM yet, I'll do this soon.
- I was distracted by the reactions feature for the network times. This was really easy to add, and I styled it indigo-300 or something like this. I've never seen a social site have this color like, so it's interesting at least.
- You can't yet press the like, this will work once I finally figure out how to make the app a valid signer for users.
- I added a profile page as shown in Figure 6.



Figure 6: Profile Page

- I added hover effects (you can't see it here but matt's pfp background has a slight black circle with 10% opacity applied here (Figure 7).



Figure 7: Pfp Hover Effect

## 7 Week 5

### I can't be asked

- I am too lazy to summarize my shitty git commits from the 13 til now (the 31st).
- Enjoy Mr. Incredible Uncanny 4 instead:

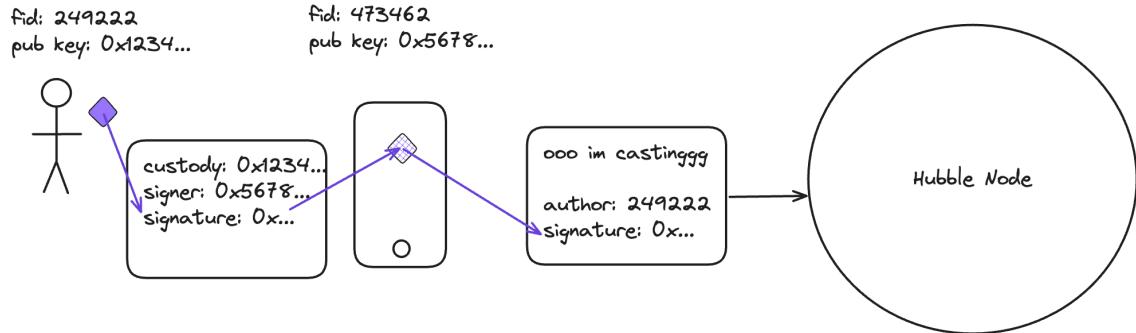


**Figure 8:** *oooooooooooooooooooo*

- Right, now I've added a few things:
  - custom link style and formatting in casts
  - render images in casts
  - sign in with farcaster (doesn't do much else yet)
  - link to view cast on warpcast
- I suppose I should add reaction data and stuff now.
- I'm avoiding the actual hard tech of having the LLM not hallucinate. I'm thinking of having a writers room where my proprietary spaghetti code strips out the things that don't make logical sense from the summary from the LLM.

## I really can't be asked

- I suck at updating this, but I'm focused on a native farcon app at the moment. you can check it out [here](#).
- I'm trying to get messaging working by handcrafting endpoints in rust (this will be my downfall, I should just use Neynar) so I'll use them in both apps anyways.
- Figure 9 is a drawing (from farcaster YT tutorial) which shows how to create a message with a custom client application.



**Figure 9:** Farcaster Signer Concept

- pinata, farcasterindex, things to thing about using recommended by a friend.

**8 Week 4 SEE WEEK 5**

## 9 Week 3 - AHHHHH

Monday, 3/11/2024

- See Figure 10.



Figure 10: *AHHHH*

Tuesday, 3/12/2024

- See Figure 11.



Figure 11: *AHHHHHHHHHHHHHHHH*

Wednesday, 3/13/2024

- Store articles in local storage to prevent multiple API calls per channel per session. Still very clunky behavior with NavBar versus logos on ArticleList.

## 10 Week 2 - Feature Maxing

**Monday, 3/11/2024**

- Settle on OVH server to host and migrate away from AWS.

**Tuesday, 3/12/2024**

- Get Claude 3 Opus API working for article gen on photo click.

**Wednesday, 3/13/2024**

- Redesign by another 10X designer, Elvia Franco.

**Thursday, 3/14/2024**

- Skim through [1] for the 4th time.
- Initialize this very document (meta).
- Start to implement redesign.
- redesign nearly complete, need to style tweets still.

**Friday, 3/15/2024**

- Adjust system prompt, use haiku to save money rofl.
- Add LATEX pdf for change log button.

**Saturday, 3/16/2024**

- See Figure 12.



**Figure 12:** *ahhh*

**Sunday, 3/16/2024**

- See Figure 12.

## 11 Week 1 - Ship Website ASAP

**Friday, 3/8/2024**

- Recruit 10x designer, Michael Raisch.
- Set ship or die goal to Sunday (Didn't know the deadline at the time).
- Install hubble and run node.
- Get basic views built.
- Rust server which makes API calls to various LLM models.

**Saturday, 3/9/2024**

- Connect rust server api calls to client components.
- Struggle with hosting server.

**Sunday, 3/10/2024**

- Launch at 6AM rofl.

## References

- [1] C. Huyen, Designing Machine Learning Systems. USA: O'Reilly, 2022.