TODO

MVP

- [ ] implement the OMAP for the user store
  - https://francismurillo.github.io/2019-07-31-Understanding-Rust-Through-AVL-Trees/
  - [1]
  - [2]
  - maybe just do a trivial avl implementation and slap in oram to access nodes that way?
    - $O(\log^3(N))$ time complexity
- [ ] user server implementation to create new user
- [ ] have a simple client side impl, preferably on pc / cli
- [ ] Nice documentation for everything
- [ ] figure out how adding a user works (OMAP semantics)
- [ ] End 2 End encryption

Post MVP

- [ ] multi-device support via support of proxy
  - inspired by how groovy had the provider system
  - to make sparta-ll into a provider based system, its fairly cheap to have a small embedded device to act as a proxy for each user its not super unfeasable
- [ ] client side implementation with sqlite?
- [ ] figure out how to get this building on fortranix sgx
  - If I dont reach this, could sell this as not feasable in such a short amount of time but looking forward to do it in the future.

Project Structure

Hermes

- proxy? or could be the name for the entire thing

Athens

- Client Cli
- Tauri mobile app

Sparta

- Sparta LL implementation

Questions for Kyle

1. Does sparta support users with multiple devices?
2. What sort of E2E encryption scheme can be added onto sparta?
3. How does authentication work with oblivious systems?

Qucklinks

oram library:

- https://github.com/facebook/oram?tab=readme-ov-file
  - only secure inside of an enclave with memory encryption

enclave framework:

- https://github.com/fortanix/rust-sgx

intel-sgx?

- https://github.com/intel/linux-sgx-driver

- i dont have the hardware

Bibliography

[1] P. Mishra, R. Poddar, J. Chen, A. Chiesa, and R. A. Popa, "Oblix: An Efficient Oblivious Search Index," in *2018 IEEE Symposium on Security and Privacy (SP)*, 2018, pp. 279–296. doi: 10.1109/SP.2018.00045.

[2] X. S. Wang *et al.*, "Oblivious Data Structures," in *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, in CCS '14. Scottsdale, Arizona, USA: Association for Computing Machinery, 2014, pp. 215–226. doi: 10.1145/2660267.2660314.