Financy

$

Smarter Money, Smarter You

Financy API

API Routes for Financy, Written by Nathan Lee for TamuHack 2024

Nathan Lee

Table of contents

API Routes

> ⚠ Replace `financy_route` with the actual route

How to Login

1. Submit a request to get a key
2. Use the key as the bearer for other routes

Routes

> ⚠ GET `http://financy_route/key`

Headers:

- None

  Body:

- Include password (as an SHA256 hash), username, and expiry as a unix timestamp.
- Formatted like: `user;hash;expiry`

  Returns

- Bearer key in form `bearer_[number]`
- Both parts of the bearer key are required.

> ⚠ POST `http://financy_route/transaction`

Headers:

- Header `x-bearer`, containing the bearer key returned by the above route.

  Body:

- Include the name given for the transaction, and then the amount as a float.
- Formatted like: `name;amount`

> ⚠ GET `http://financy_route/balance/<bearer>`

<bearer>: The API token Returns: The balance for the user.

> ⚠ GET `http://financy_route/transactions/<bearer>`

<bearer>: The API token Returns: The total transactions for the user, separated by newlines. Transactions are formatted like `unix timestamp:name:amount`

Financy

> ⚠ ***todo***

Inspiration

> ⚠ todo

What It Does

> ⚠ todo

How We Built It

We used Swift UI to build a simple yet powerful frontend for iOS apps. We also wrote a backend in Rust to deploy a RESTful API quickly and efficiently using Rocket.

Challenges We Ran Into

We were very limited in the number of features we could do because in addition to time constraints, we also only had 2 people on our team. In addition to that, our frontend developer didn't know Rust and our backend developer didn't know Swift UI. We overcame these challenges by efficiently communicating the specifics about the API routes, what we wanted for the backend, and how the UI should be designed in the frontend.

Accomplishments That We're Proud Of

This was our first hackathon, and we had a lot of challenges to overcome with the amount of work to be split among 2 people. We were both equally proud of:

- How quick the UI design was made
- How speedy all the routes for the API were written Overall, we like to see how efficiently our code was written and fully completed.

What We Learned

We learned that we need to have more concurrency in the programming languages that we know, so that in the future, when someone is done, or not doing anything, they can help program and finish it instead of doing busywork that is usually reserved for the end of the hackathon, like presentations and Dev Post submissions.

What's Next?

> ⚠ todo

How to Run

Requires:

- MacOS
- `rustc` version >=1.74.1
- `cargo` version >=1.74.1

Install and run the XCode project. Then, in a separate folder, start the Rocket server.