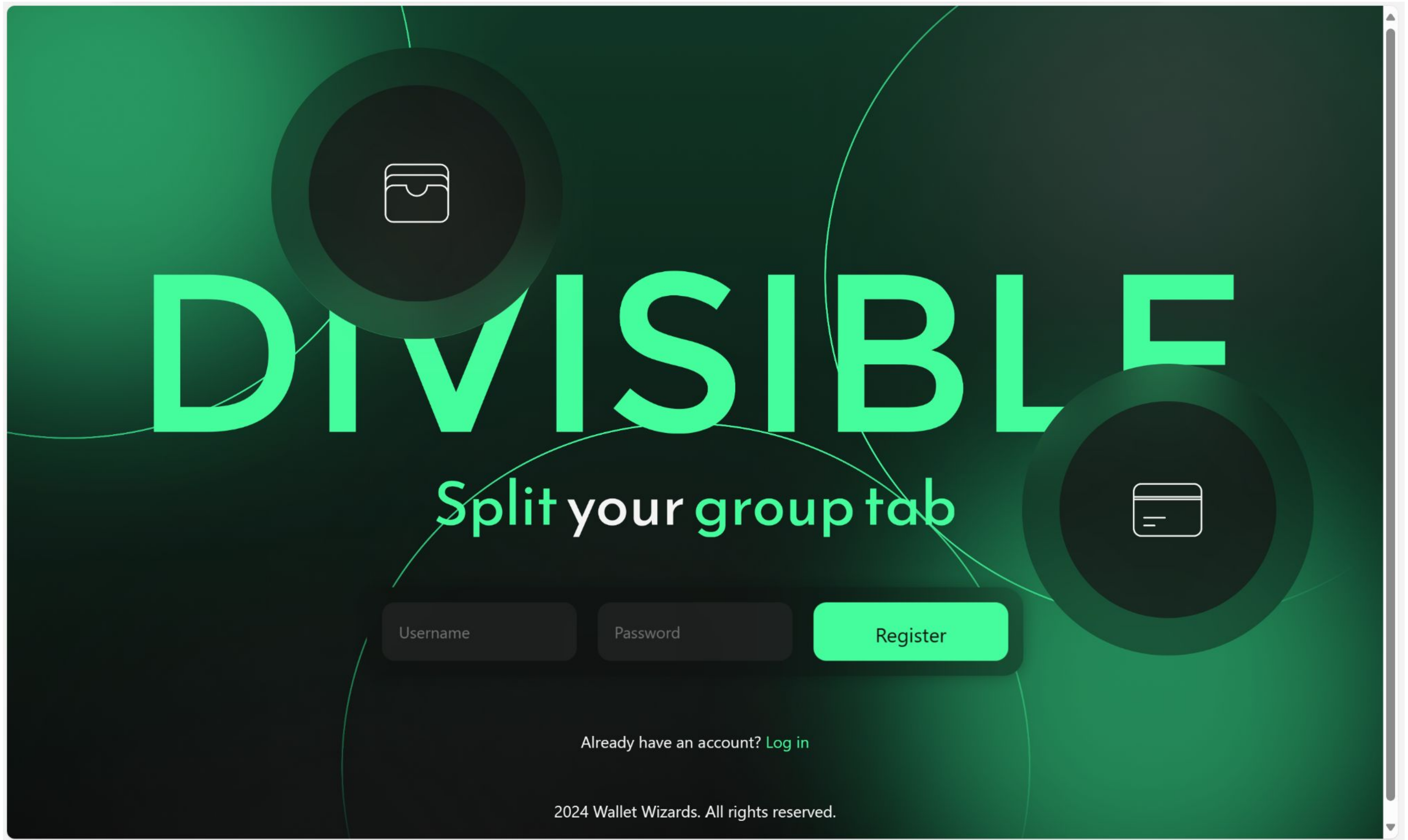
Ben Booker, Jace Simons, Noah Bays, Lucca Giannini, Logan Haase, Owen Villas


DIVISIBLE

Split your group tab

Already have an account? Log in

2024 Wallet Wizards. All rights reserved.

Project Description

Divisible is a group payment application. You can keep track of group expenses, and even them up with venmo-like payment functionality. If you need to just pay individual friends, you can do that too, by clicking the Pay Individuals tab. But the focus is on group payment. Add friends in our application, and create groups. Then, when you log in to Divisible, you can see all your groups, their members, what you owe, and a transaction history for that group. Simplify splitting your tab with Divisible.

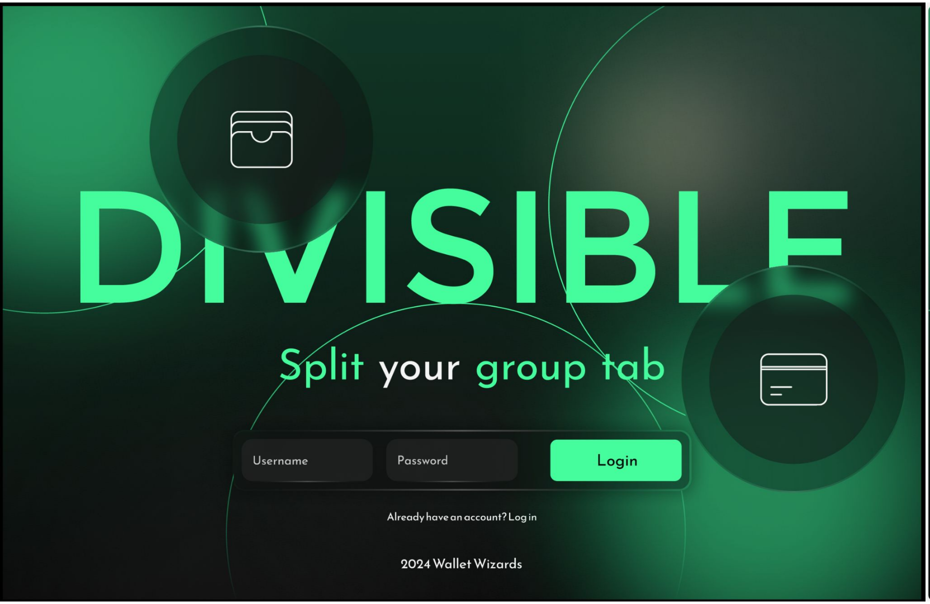
Resources

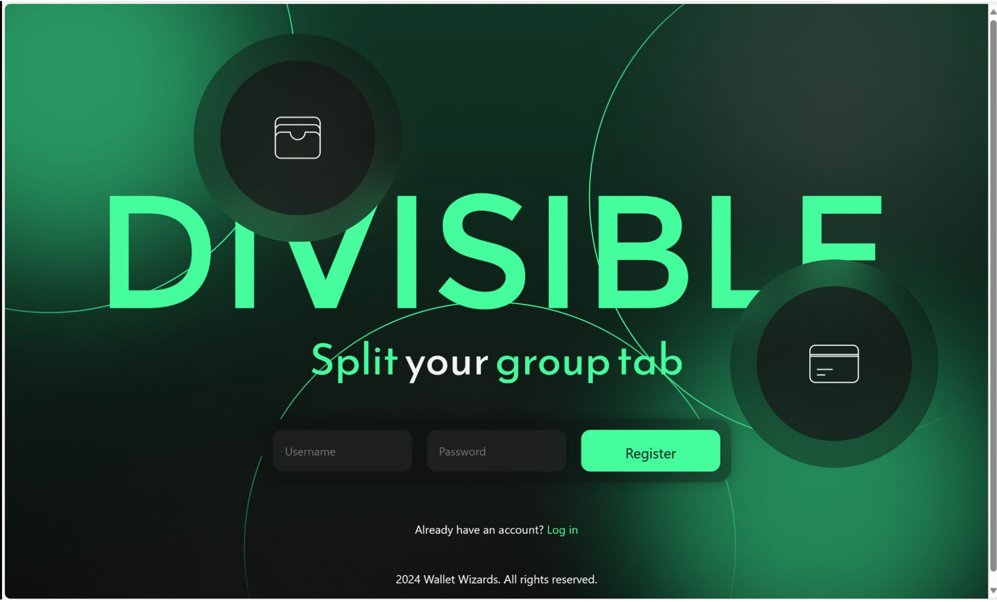
Github: Divisible: Automated expense sharing for groups (github.com)

Github Project Board: Board · Recitation-016-Team-03 (github.com)

Figma mockups

Real Application




DIVISIBLE
Split your group tab
Username
Password
Register
Already have an account? Log in
2024 Wallet Wizards. All rights reserved.

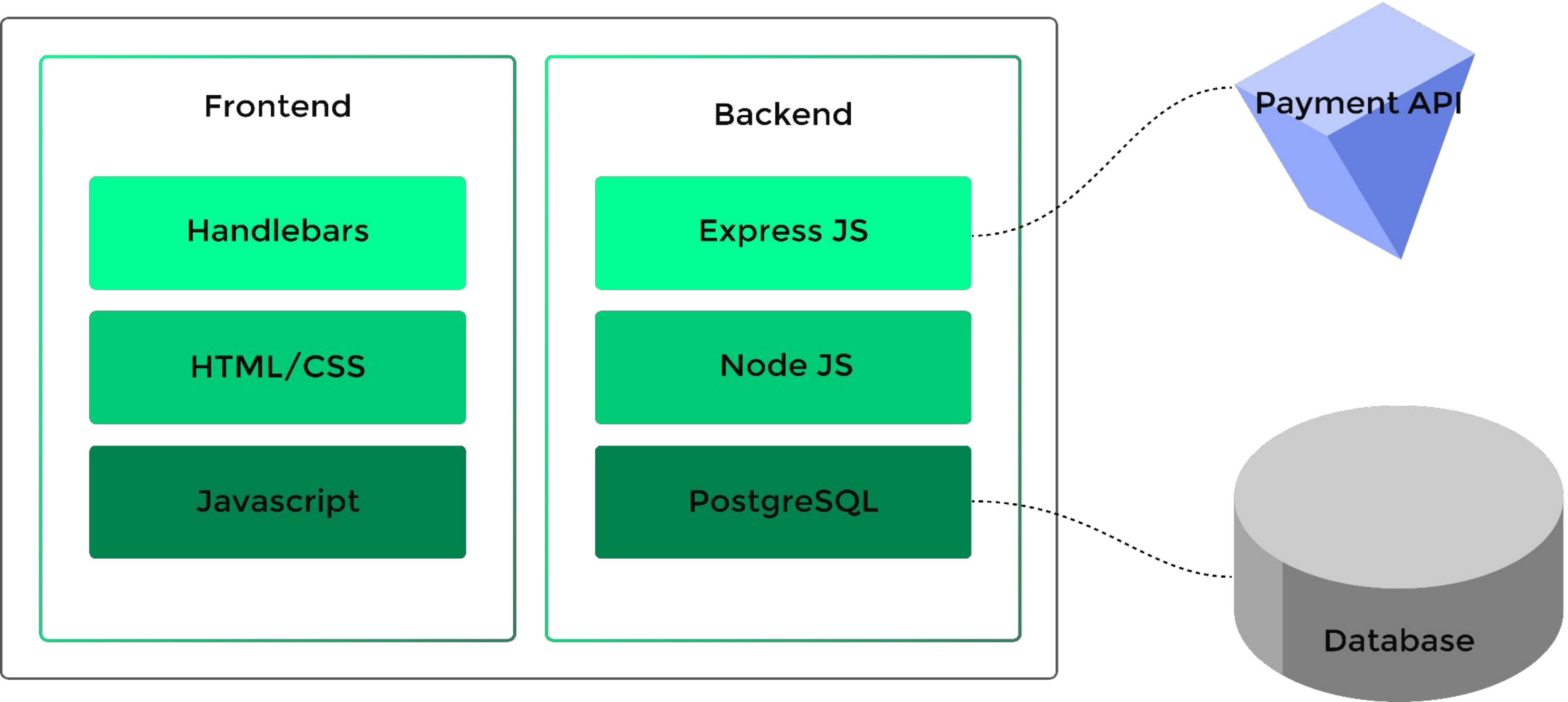
Tools

1. VCS: Github
    a. We rate this tool a ⅘ stars. It is a great place to hold our code in an easily accessible place, but some of its features are a bit complex for beginners, like pull requests, and the code review process itself.
2. Project Tracking: Github Projects
    a. We rate this tool a ⅗ stars. It gets the job done, but it’s very basic and pretty complicated to set up. It also has unnecessary steps for creating issues and transitioning them to tickets, setting up the columns, etc.
3. Database: PostgreSQL
    a. We rate this tool ⅘ stars. No complaints, but nothing special. SQL is pretty easy to use for queries and such, and the database spins up pretty fast.
4. IDE: VSCode
    a. We rate this 5/5 stars. Simple to use and familiar.
5. UI Tools: Handlebars
    a. We rate this a ⅖ stars. We understand the value of templating, but they make it overly complicated to implement some things to the point where it's borderline impossible. For example, we wanted to have a toggle at the top of the Payment page between Groups/Individuals, and it changes which partial is rendered below it, but handlebars.js maintainers have some hangup about ‘logic-less templates’, meaning you can’t easily achieve this, so we had to scratch this approach and simply make two separate pages.

Tools (cont.)

1. Application Server: Node JS
   a. We rate this 5/5 stars. It is very good at what it does, and we didn’t have any actual problems with node.js itself.
2. Deployment: localhost
   a. We rate this ⅖ stars. We had the application hosted at the point of Lab 13, but we ran into some problems and eventually decided just to continue with localhost. It allows you to see it, but of course you have to build and run the application locally to open it. If this application ever moves forward, one of the first things we would do is get it hosted.
3. Testing Tool: Mocha/Chai
   a. We rate this ⅗ stars. It runs the tests, but it is kinda complicated to write them and the syntax is confusing. They also take a comparatively long time to run.
4. Containers: Docker
   a. We rate this ⅗ stars. The containers are nice for keeping your environment clean, but it is pretty cluttered and fragile. We had multiple instances where someone would accidentally overwrite docker-comose.yaml, and its error messages weren’t helpful enough to diagnose this problem for a while.
5. Mockups: Figma
   a. We only made a few true mockups (we did hand drawn wireframes the first week), but the ones we did make are in Figma. We rate this ⅘ . We had a few people with experience using it, so they like it, but it’s a pretty steep learning curve for new users.

Divisible Application Stack



Challenges

1. Register and login endpoints
    a. For the first week, we couldn't get our api endpoints to play nicely with bcrypt and return errors or success properly. We solved this by taking a look at the problem as a group, and eventually identifying some problems with the code. We then did peer programming to get these endpoints written properly
2. Paypal/Venmo APIs
    a. Originally, we wanted to make use of either the Paypal API or the Venmo API to complete real payments. We found that the Venmo API has since been shut down. We then wanted to employ Paypal, but that would have cost real money, so we reorganized our plans for the application, and moved forward having made new plans in the face of adversity.
3. Payment api
    a. The biggest piece of our application is the custom payment api. This was a big group effort, that came with a lot of smaller challenges. We employed a lot of peer programming to write/fix code, and especially during the code review process to ensure that it was working properly

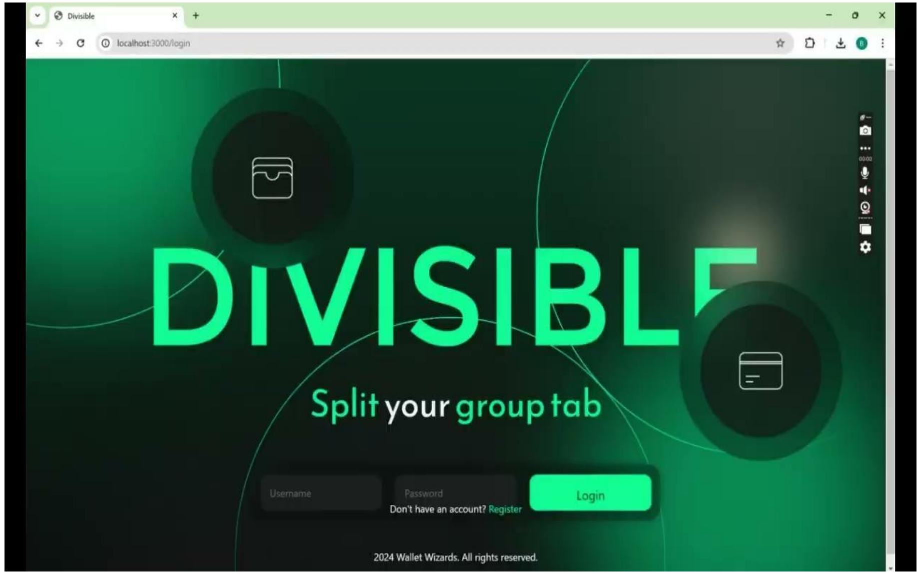
Future Enhancements

If Divisible were to move forward in the future, we have 3 main goals

1. Implement the Paypal API to make the application functional with real money
2. Host the application using Google Cloud or Azure, or something similar.
3. Lastly, we would want to strip out the Handlebars dependence, and instead write the dynamic parts of our application using a framework like Vue or React.

5 Minute Video Demo

TODO


Divisible
localhost:3000/login
DIVISIBLE
Split your group tab
Username
Password
Login
Don't have an account? Register
2024 Wallet Wizards. All rights reserved.