Want to join us?

The test is a simple CRUD Typescript Node JS application designed to give us a small glimpse into your coding abilities. **It is not essential to return a fully functional application**, although that would be amazing. **Please submit it even if you think it is not ready.**

Definition

Implement a simple money transfer interface to transfer money between two accounts using both back-end and front-end technologies.

- All calculations should be performed by the back-end.
- Front-end represents the supporting UI to interface with the back-end API.

Back-end

The backend should be developed in the RESTful way. End-points should accommodate for caching, although the caching layer itself is not required. Each endpoint must be clearly documented and have basic tests in place. It is expected that each failure will be handled gracefully and a meaningful error message along with appropriate HTTP status code is returned. API Authentication is not required.

The back-end should be implemented using the following technologies:

- Node.js (any framework of your choice)
- Typescript

Front-end

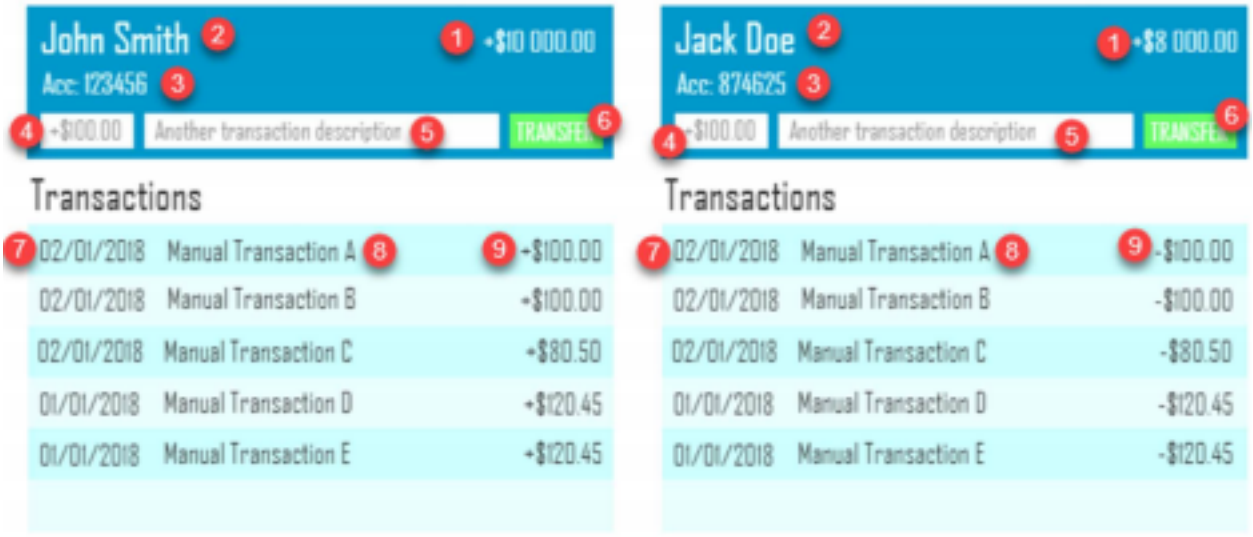
The main goal of the front-end is to represent the graphical UI of the bank account interface, facilitate money transfer capabilities and capture users input for processing by the API.

The front-end should be implemented using the following technologies:

- React.js (preferably using create-react-app)
- Typescript

Don’t be concerned if you don’t have experience with Typescript, we’re interested in how easily you adapt to new technologies and having you use a new technology gives us good insights.

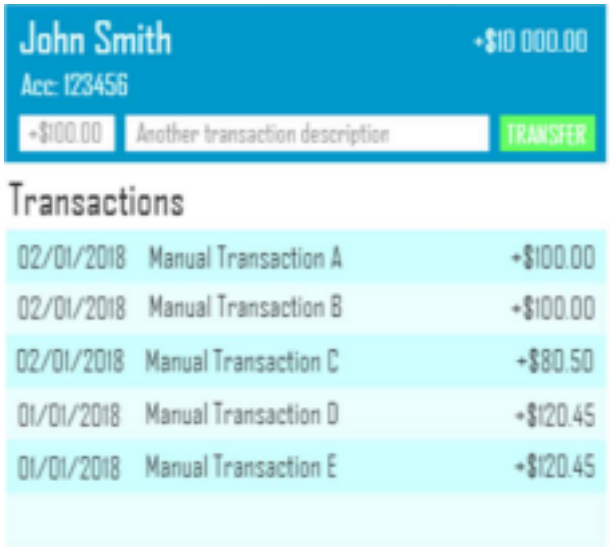
You will be building the following interface:

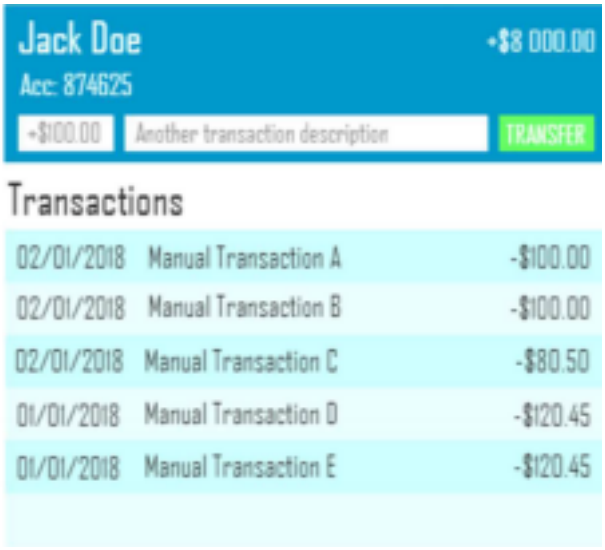


NOTE: What you see in the picture above is **one screen NOT two.** You are required to present both accounts on the same page opposite to each other.

- Both accounts must clearly show name (2), account number (3) and balance (1).
- It is possible to transfer money from one account to the other by entering transfer amount (4), transaction description (5).
- By clicking the transfer button (6) it is assumed that the transaction destination is set to the account number on the opposite side of the screen.
- Data must be sanitized and validated before the transfer can take place. ● Transaction cannot take place if the source account balance will become negative after the transaction (1).
- Once the transaction takes place, it should be reflected in the transaction area of both accounts.
- Each transaction line item has the
    - date of the transaction (7)
    - original transaction description (8) and
    - transaction amount (9) (either debit or credit based on the transaction relationship to the account).

CLEAN EXAMPLE





Some things to consider:

- **unit testing:** not essential, but the inclusion of a functional test suite would be nice to see
- **logging:** an HTTP request log, to file or console, would also be a nice touch
- **styling**: don't spend too much time on styling, as long as it's functional and correctly interfaces with the backend it's ok

Once completed upload a zip archive into the portal, along with instructions for installing and running it.
Have fun!

N.B. Please send your repository link and/or any questions to techchallenge@creditsense.com.au