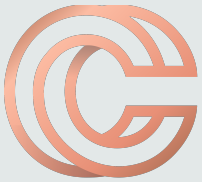


copper.co

Technical Assessment Test iOS Developer

Private & Confidential

Technical assessment test

We would like you to build a simple application that consists of one screen only. The main functionality of the app is to fetch orders from our test API, save them to persistent storage on the device and display them to the user. The tricky part is that you will receive a list with more than 10,000 items, ought to this you should carefully organise the local storage and multithreading processes so that UI will not be affected while objects will be saving. Also, it is very important to follow all designs guidelines while making the interface of the app.

Requirements:

1. You must use CoreData as persistent storage. It is prohibited to use any external libraries to work with the local storage.
2. You can use any tooling (including external open-source libraries) that you prefer for managing architecture, networking, animations and etc.
3. While the data will be downloading and saving, UI must not be freeze, but stay responsive and smooth.
4. All design guidelines must be appropriately met.

Deadline:

You should make an app within a week after you received an assessment.

Submitting:

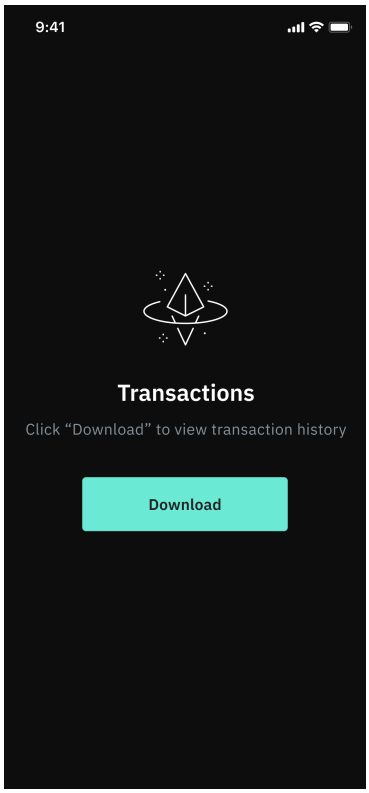
Upload your code to a source control system that suits you. The repository with the code should be accessible for everyone to view.

Resources:

1. The link to fetch the test data is
GET https://assessments.stage.copper.co/ios/orders

2. Design guidelines can be found here:
https://www.figma.com/file/KooWcNZDz90ZhR2dAsc0se/Test
(you should have a Figma account to be able to export files)

Steps:



1. When a user opens the application, he sees the screen with download button.

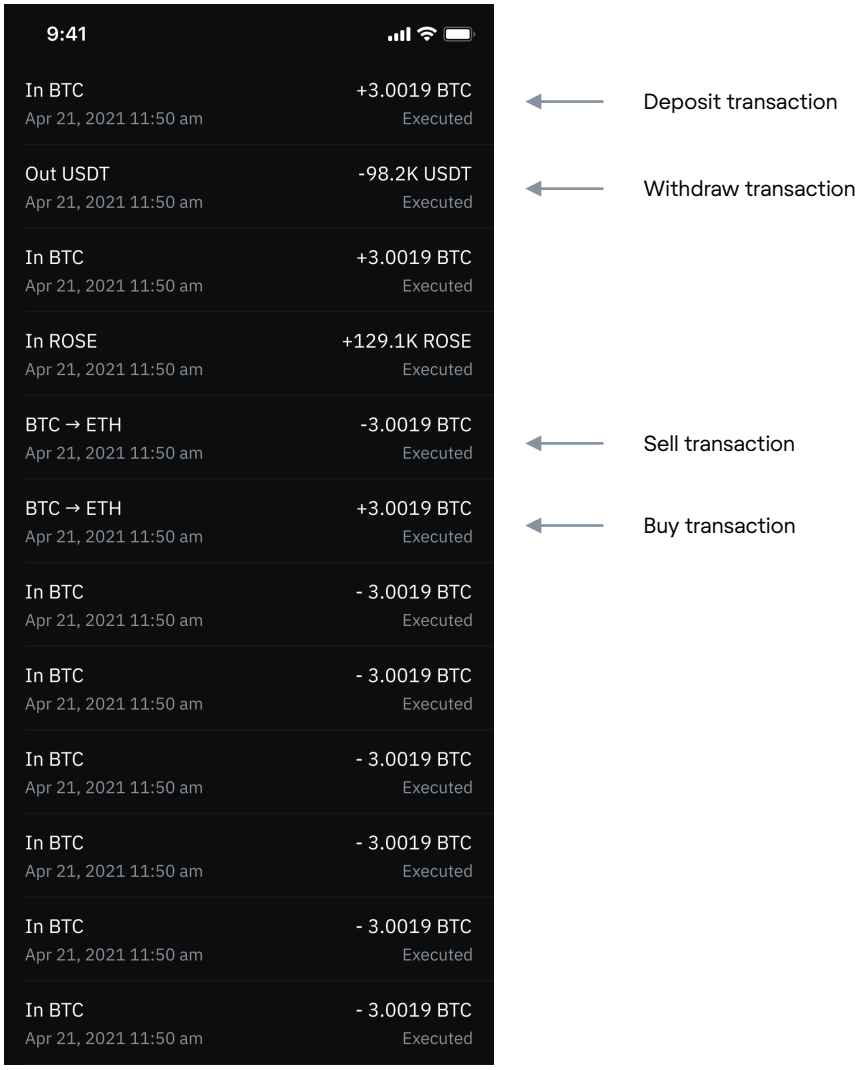
2. After the “Download” button is clicked, an activity indicator should appear while fetching data from the server.

3. You will receive a response similar to this:


```
{
    "orders": [
        {
            "orderId": "3a16aef1d2afe8af1ad52fd4ec374fae",
            "currency": "BTC",
            "amount": "748.279727546401",
            "orderType": "deposit",
            "orderStatus": "approved",
            "createdAt": "1595770212105"
        },…
    ]
}
```


orderId	identifier of the order
currency	currency identifier
amount	amount of the order
orderType	type of the order. Can be one of the following list: **deposit, withdraw, buy, sell**
orderStatus	status of the order. Can be one of the following list: **executed, canceled, approved, processing**
createdAt	timestamp of the date when the order was created in milliseconds

4. After you receive the mock data, it should be saved to the local storage and displayed to the user following next rules:



5. When the application opens after the orders have been already fetched, the user should see the screen with fetched orders.

6. Good luck!