# Full Stack Developer Test

In this test you will be required to write a simple end-to-end application comprising of:

1. Vue.js front end. Use any CSS or other framework/component you are familiar with, there is no limitation.
2. Node.js back end that will call to an external API.

What we are trying to build is an application similar to an existing one available on the web. For example, use this URL:

<https://www.blockchain.com/btc/address/1DBrYbe5U7LGDcHA5tiLCxivZ7JZAGqGhJ>

what we see are transfers made two and from a bitcoin account. Incoming 50 coins in 2009 and later in 2011 transfer of the same 50 into another account, but together with many more 50 coin transactions.

# Nodejs Backend

Write a nodejs application to create a proxy service for encapsulating following service, from api.cryptoapi.io.

The documentation for the API is here:

<https://docs.cryptoapis.io/rest-apis/blockchain-as-a-service-apis/btc/index#btc-basic-address-transactions-endpoint>

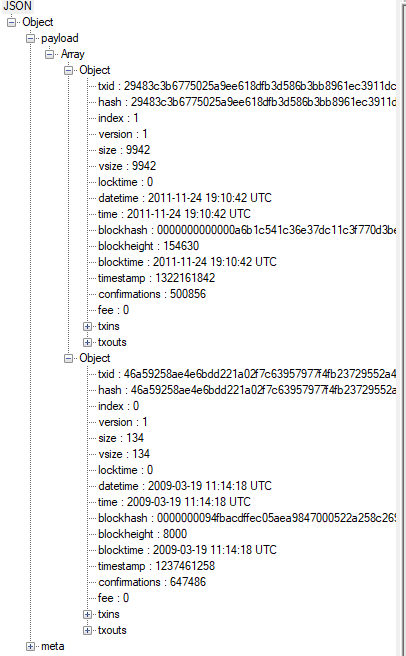
* Web Service method: GET
* Url: GET /v1/bc/btc/${NETWORK}/address/${ADDRESS}/transactions?index=0&limit=50

A sample call is below (using cURL on Windows). We are using the API key to be used in your test, which is **d45bcc3bb736ddd5d0bd7fac45149f24cf4491a4** , and we have a specific Bitcoin address used **1DBrYbe5U7LGDcHA5tiLCxivZ7JZAGqGhJ**.

curl -X GET "https://api.cryptoapis.io/v1/bc/btc/mainnet/address/1DBrYbe5U7LGDcHA5tiLCxivZ7JZAGqGhJ/transactions" -H "Content-Type: application/json" -H "X-API-Key: d45bcc3bb736ddd5d0bd7fac45149f24cf4491a4"

Your encapsulation of this service should be able to handle a case where the API key is invalid.

The service will return a payload where each array item is a transaction. In the sample below, there are two (2) transactions:



Each transaction will have txid, and each will have nested inside the incoming and outgoing transactions, in two arrays:

txins – incoming transactions

for example:

"txins": [

{

"txout": "02c4c9c002e2da17d9edb0144a876d7bf92be0afdb2373a4d1d25f96fee5c826",

"vout": 1,

"amount": "0.00075333",

"addresses": [

"mtFYoSowT3i649wnBDYjCjewenh8AuofQb"

],

"script": {

"asm": "3044022072ffc0d6ba2acbf11de33d7bf2723d667a166eeadad1920d98732ce29593e5310220605de47003debe7580ee85b3c877f04b311b391ee3381bb0a0c71ed43c023c06[ALL] 02275753690ab58df3c923001e94d407e30b03e60b1f2461729a1dd4f37ebe2469",

"hex": "473044022072ffc0d6ba2acbf11de33d7bf2723d667a166eeadad1920d98732ce29593e5310220605de47003debe7580ee85b3c877f04b311b391ee3381bb0a0c71ed43c023c06012102275753690ab58df3c923001e94d407e30b03e60b1f2461729a1dd4f37ebe2469"

},

"votype": "pubkeyhash"

},

...

So here the txout is the ID of first incoming transaction, amount is amount of coins, addresses is the address from which it came.

txouts – outgoing, similar idea (look at amounts, and addresses to see what amount sent to where).

"txouts": [

{

"amount": "0.00018",

"type": "pubkeyhash",

"spent": false,

"addresses": [

"mmskWH7hG9CJNzb16JaVFJyWdgAwcVEAkz"

],

"script": {

"asm": "OP\_DUP OP\_HASH160 45bfa6143f69be3475e17724397f1146a60a81f3 OP\_EQUALVERIFY OP\_CHECKSIG",

"hex": "76a91445bfa6143f69be3475e17724397f1146a60a81f388ac",

"reqsigs": 1

}

},

# JavaScript Front-End

Write a single page web application using Vuejs for following requirements. You can use any front end css framework for tile view. You can use css grid and css flexbox.

You are encouraged to write UI components in the chosen framework. You are supposed to provide instructions in “README.md” on how to build and run your application.

1. Create a user interface where the user to enter a bitcoin address
2. The results from the crypoapi service (encapsulated by your node.js service) returns a list transactions. Each transaction is either incoming or outgoing and will have a list of transfers.

# Final Test

We will provide another bitcoin address and see the results for that address.