

# Udacity Blockchain Project 1 v2

## Private Blockchain Application

Development environment used VS Code + Thunder Client. Screen shots and copy/paste of each get/post operation included below in order provided by course documentation.

### Get Height

```
{
  "hash": "69660ab8739427caa987c4291282d71fc350dff7a33a45f99d322fe375c3f8f",
  "height": 0,
  "body": "7b2264617461223a2247656e6573697320426c6f636b227d",
  "time": "1649695555",
  "previousBlockHash": null
}
```

GET http://localhost:8000/block/height/0 Send

Query Headers 2 Auth Body 1 Tests

Query Parameters

parameter	value
-----------	-------

Status: 200 OK Size: 189 Bytes Time: 26 ms

Response Headers 6 Cookies Results Docs New

```
1 {
2   "hash": "69660ab8739427caa987c4291282d71fc350dff7a33a45f99d322fe375c3f8f",
3   "height": 0,
4   "body": "7b2264617461223a2247656e6573697320426c6f636b227d",
5   "time": "1649695555",
6   "previousBlockHash": null
7 }
```

PROBLEMS OUTPUT **DEBUG CONSOLE** TERMINAL

Filter (e.g. text, !exclude)

C:\Program Files\nodejs\node.exe .\app.js

Server Listening for port: 8000

app.js:51

## Request Validation

"mrqDJNfg8pTGuT8ZxHJXBJCSmtwsR6CvGz:1649695665:starRegistry"

The screenshot shows a web browser's developer tools interface. At the top, a POST request is shown to `http://localhost:8000/requestValidation`. The request body is a JSON object: `{ "address": "mrqDJNfg8pTGuT8ZxHJXBJCSmtwsR6CvGz" }`. The response status is 200 OK, with a size of 60 Bytes and a time of 3 ms. The response body is a JSON object: `{ "hash": "43a167bf519ed3e309256dd05efac608e1c9f9735da7ed6f4298b209e3dcd372", "height": 4, "body": "7b226f776e6572223a226d7271444a4e6667387054477554385a78484a58424a43536d74777352364376477a222c2273746172223a7b22646563223a223638c2b0203532272035362e39222c227261223a223136682032396d20342e3473222c2273746f7279223a225465737420537461722034227d7d", "time": "1649695813", "previousBlockHash": "945fe825a34177c8731c31d916fc94523ec0ab78d72854c34365254b57a497da" }`. The bottom of the screenshot shows the terminal output of a Node.js application, indicating it is listening on port 8000.

POST `http://localhost:8000/requestValidation` Send

Query Headers <sup>2</sup> Auth Body <sup>1</sup> Tests

Json Xml Text Form Form-encode GraphQL Binary

Json Content

```
1 {
2   "address": "mrqDJNfg8pTGuT8ZxHJXBJCSmtwsR6CvGz"
3 }
```

Format

Status: 200 OK Size: 60 Bytes Time: 3 ms

Response Headers <sup>6</sup> Cookies Results Docs New

```
1 { "hash": "43a167bf519ed3e309256dd05efac608e1c9f9735da7ed6f4298b209e3dcd372",
  "height": 4,
  "body": "7b226f776e6572223a226d7271444a4e6667387054477554385a78484a58424a43536d74777352364376477a222c2273746172223a7b22646563223a223638c2b0203532272035362e39222c227261223a223136682032396d20342e3473222c2273746f7279223a225465737420537461722034227d7d",
  "time": "1649695813",
  "previousBlockHash": "945fe825a34177c8731c31d916fc94523ec0ab78d72854c34365254b57a497da"
}
```

PROBLEMS OUTPUT **DEBUG CONSOLE** TERMINAL

Filter (e.g. text, !exclude)

`C:\Program Files\nodejs\node.exe .\app.js`

Server Listening for port: 8000

app.js:51

Submit 4 stars:

```
{
  "hash": "43a167bf519ed3e309256dd05efac608e1c9f9735da7ed6f4298b209e3dcd372",
  "height": 4,
  "body":
    "7b226f776e6572223a226d7271444a4e6667387054477554385a78484a58424a43536d74777352364376477a222c2273746172223a7b22646563223a223638c2b0203532272035362e39222c227261223a223136682032396d20342e3473222c2273746f7279223a225465737420537461722034227d7d",
  "time": "1649695813",
  "previousBlockHash": "945fe825a34177c8731c31d916fc94523ec0ab78d72854c34365254b57a497da"
}
```

The screenshot shows the Bitcoin Signatures application window. At the top, there's a title bar "Signatures - Sign / Verify a Message". Below it, there are two tabs: "Sign Message" (selected) and "Verify Message". A warning message states: "You can sign messages/agreements with your addresses to prove you can receive bitcoins sent to them. Be careful not to sign anything vague or random, as phishing attacks may try to trick you into signing your identity over to them. Only sign fully-detailed statements you agree to." Below the warning is a text input field containing the address "mrqDJNfg8pTGuT8ZxHJXBjCSmtwsR6CvGz". To the right of the input field are two icons: a QR code icon and a document icon. Below the input field is a large text area containing the signed message: "mrqDJNfg8pTGuT8ZxHJXBjCSmtwsR6CvGz:1649695665:starRegistry". Below the text area is a "Signature" label and a text input field containing the signature: "II7OC5XCd9nPCW8MBxcFG2Up0k8wXMm+Ng3ozpDLhKzPCox4zT8b1QdjxffAPXxgYWxaFGL3CCuFF5C4fCjTI=". To the right of the signature input field is a document icon. At the bottom, there are two buttons: "Sign Message" (with a pencil icon) and "Clear All" (with a trash icon). A green status message "Message signed." is displayed on the right side of the bottom bar.

## Get Star by Address

```
[
  {
    "owner": "mrqDJNfg8pTGuT8ZxHJXBJCSmtwsR6CvGz",
    "star": {
      "dec": "68° 52' 56.9",
      "ra": "16h 29m 1.0s",
      "story": "Test Star 1"
    }
  },
  {
    "owner": "mrqDJNfg8pTGuT8ZxHJXBJCSmtwsR6CvGz",
    "star": {
      "dec": "68° 52' 56.9",
      "ra": "16h 29m 2.2s",
      "story": "Test Star 2"
    }
  },
  {
    "owner": "mrqDJNfg8pTGuT8ZxHJXBJCSmtwsR6CvGz",
    "star": {
      "dec": "68° 52' 56.9",
      "ra": "16h 29m 3.3s",
      "story": "Test Star 3"
    }
  },
  {
    "owner": "mrqDJNfg8pTGuT8ZxHJXBJCSmtwsR6CvGz",
    "star": {
      "dec": "68° 52' 56.9",
```

```
"ra": "16h 29m 4.4s",  
  
"story": "Test Star 4"  
  
}  
  
}  
  
]
```

The screenshot displays the Thunder Client interface. At the top, a GET request is configured to `http://localhost:8000/blocks/mrqDJNfg8pTGuT8ZxHJXBjCSmtwsR6CvGz`. Below the request bar, the 'Query Parameters' section is empty. The response status is `200 OK` with a size of `489 Bytes` and a time of `4 ms`. The response body is a JSON array with two objects. The first object has an 'owner' and a 'star' object containing 'dec', 'ra', and 'story' fields. The second object is partially visible. At the bottom, the 'DEBUG CONSOLE' tab is active, showing the command `C:\Program Files\nodejs\node.exe .\app.js` and the message `Server Listening for port: 8000`. The file `app.js` is highlighted at line 51.

```
GET http://localhost:8000/blocks/mrqDJNfg8pTGuT8ZxHJXBjCSmtwsR6CvGz Send
```

Query Headers 2 Auth Body Tests

Query Parameters

parameter value

Status: 200 OK Size: 489 Bytes Time: 4 ms

Response Headers 6 Cookies Results Docs New

```
1 [
2   {
3     "owner": "mrqDJNfg8pTGuT8ZxHJXBjCSmtwsR6CvGz",
4     "star": {
5       "dec": "68Â° 52' 56.9",
6       "ra": "16h 29m 1.0s",
7       "story": "Test Star 1"
8     }
9   },
10  {
11    "owner": "mrqDJNfg8pTGuT8ZxHJXBjCSmtwsR6CvGz",
```

PROBLEMS OUTPUT **DEBUG CONSOLE** TERMINAL

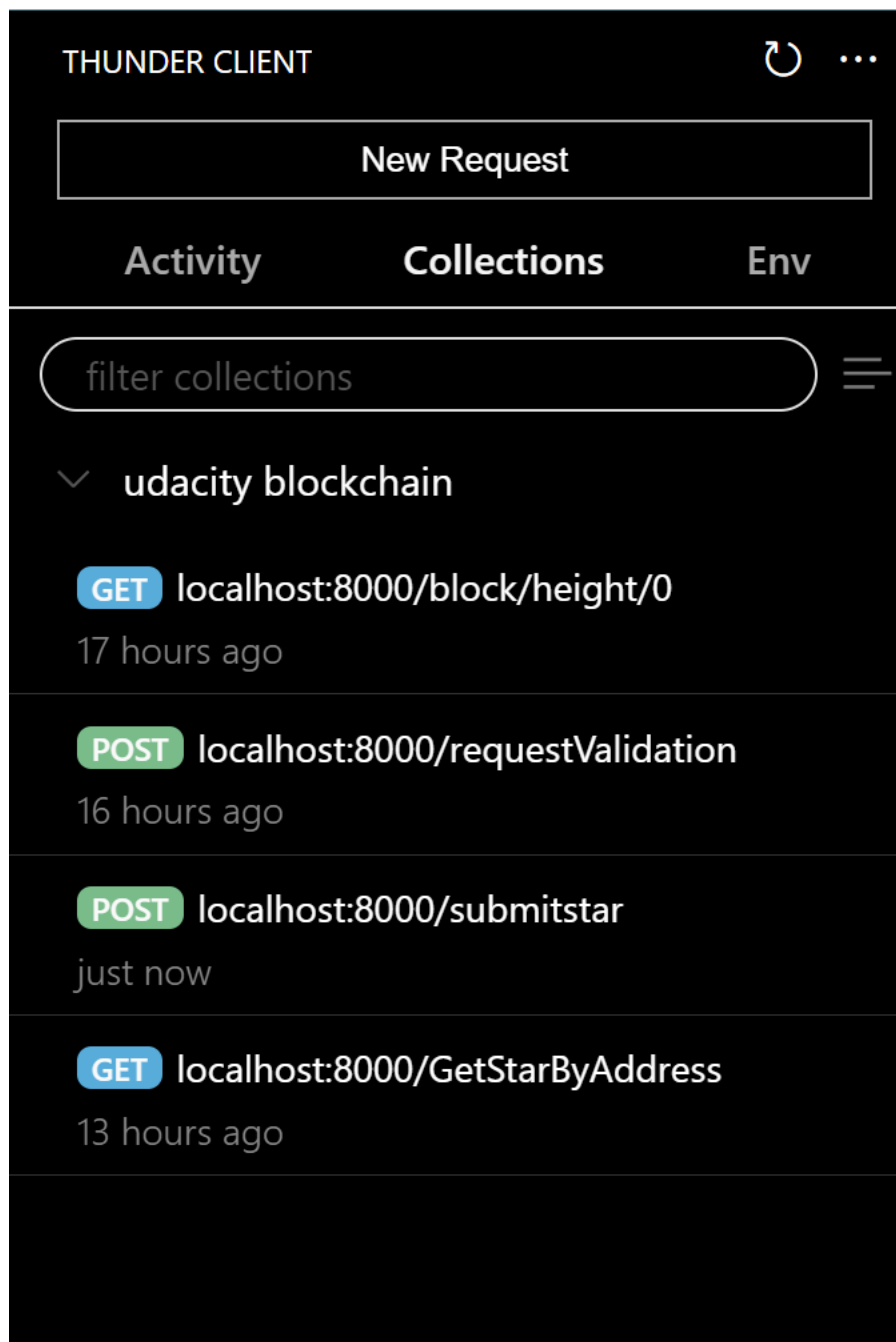
Filter (e.g. text, !exclude)

C:\Program Files\nodejs\node.exe .\app.js

Server Listening for port: 8000

app.js:51

**Thunder Client config (instead of Postman)**



Feedback:

The overall layout, source code files, readme and instructions are all excellent, only the address signing activities needs to be updated due to deficiencies in the Electrum wallet. From my limited research it appears difficult to obtain legacy addresses from Electrum and other wallets making the address invalid

when trying to sign the message for the final steps of the project post operations. There are many simplified options available, one of which is to use Bitcoin core itself. I recommend updating the content with a different mechanism as to keep the focus on the code and not on operational issues that are not as valuable.

Must use command line with legacy to get valid address for use with Bitcoin Core

```
bitcoin-cli -testnet getnewaddress "" legacy
```

