Test cases for broken-hashserve V. 0c3d817.1

Summary:

These test cases are meant to provide comprehensive test coverage for the broken-hashserve app. Both positive tests and negative tests were conducted to write these test cases and capture their intended, as well as unintended, behavior.

Test case 1: Post request to /hash endpoint with valid json

Scenario:

POST to /hash endpoint with json with the following json payload {“password”:”*your_password*”}

Prerequisites:

1. Install broken-hashserve app
2. Set PORT in the terminal by entering *export PORT=8088*
3. Start broken-hashserve app by entering *broken-hashserve* in the terminal

Steps:

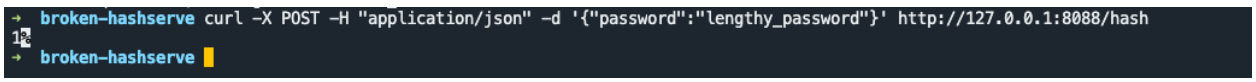
1. Enter the following command in the command line (you may choose your password to your liking):

   curl -X POST -H “application/json” -d “{“password”:”*your_password*”}” http:127.0.0.1:8088/hash

Expected Behavior:

In the terminal, a number should be displayed. This number is indicative of the job identifier followed by percent sign delimiter. This number should be displayed immediately after entering the command. The server returns a 200

Example:



Test Case 2: Post request to /hash endpoint with invalid json

Scenario:

POST to /hash endpoint with *invalid* json with the following json payload {“password”:”*your_password*”,}

Prerequisites:

1. Install broken-hashserve app
2. Set PORT in the terminal by entering *export PORT=8088*
3. Start broken-hashserve app by entering *broken-hashserve* in the terminal

Steps:

1. Enter the following command in the command line (you may choose your password to your liking but the json must be invalid. To check for validity of JSON here at https://jsonlint.com/):

   curl -X POST -H “application/json” -d “{“password”:”*your_password*”,}” http:127.0.0.1:8088/hash

   The extra comma is what makes the json listed above as invalid json

Expected Behavior:

Server responds with a 400 and with a message to the terminal “Malformed Input”

Example:

Test Case 3: Post request to /hash endpoint with empty string

Scenario:

POST to /hash endpoint with *invalid* json with the following json payload {“password”: “”}

Prerequisites:

1. Install broken-hashserve app
2. Set PORT in the terminal by entering *export PORT=8088*
3. Start broken-hashserve app by entering *broken-hashserve* in the terminal

Steps:

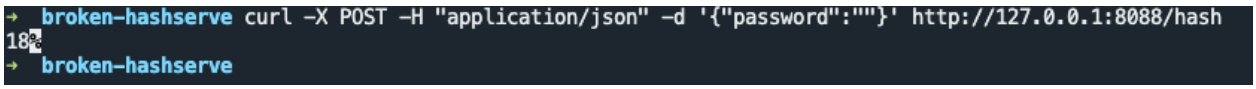
1. Enter the following command in the command line. In the json payload, the value for password must be an empty string

   curl -X POST -H “application/json” -d “{“password”: “”}” http:127.0.0.1:8088/hash

Expected Behavior:

There should be some server side validation against the password. Please note, that this test case will fail when running tests since the server returns a 200.

Example:



Test Case 4: Get request to /hash endpoint with Job Identifier

Scenario:

GET to /hash endpoint with a job identifier appended to the URL.

Prerequisites:

1. Install broken-hashserve app
2. Set PORT in the terminal by entering *export PORT=8088*
3. Start broken-hashserve app by entering *broken-hashserve* in the terminal
4. Execute the first test case, listed above, at least once, so that there is a testable response from the server

Steps:

1. Enter the following command in the command line:

   curl -H “application/json” http://127.0.0.1:8088/hash/1

Expected Behavior:

The terminal shall display the base64 encoded password delimited by the percent sign.

Example:

Test Case 5: Get request to /hash endpoint with a non existent Job Identifier

Scenario:

GET to /hash endpoint with a non-existent job identifier appended to the URL.

Prerequisites:

1. Install broken-hashserve app
2. Set PORT in the terminal by entering *export PORT=8088*
3. Start broken-hashserve app by entering *broken-hashserve* in the terminal Execute the first test case, listed above, at least once, so that there is a testable response from the server

Steps:

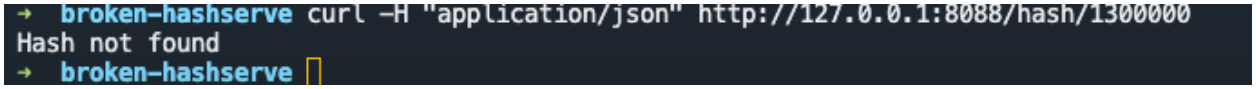
2. Enter the following command in the command line:

   curl -H “application/json” http://127.0.0.1:8088/hash/100000

Expected Behavior:

The server responds with a 400 and the message ‘Hash Not Found\n’ is displayed on the terminal

Example:



Test Case 6: Get request to /stats endpoint

Scenario:

GET to /stats endpoint shall return a payload with the number of requests made to the server since uptime as well as the average time of a hash request in millaseconds.

Prerequisites:

1. Install broken-hashserve app
2. Set PORT in the terminal by entering *export PORT=8088*
3. Start broken-hashserve app by entering *broken-hashserve* in the terminal
4. Execute the first test case, listed at the top of the first page, at least once, so that there is a testable response from the server

Steps:

1. Enter the following command in the command line:

   curl http://127.0.0.1/hash/stats

Expected Behavior:

The terminal shall display a json response in the schema of

```
{
  "$schema": "http://json-schema.org/draft-04/schema#",
  "type": "object",
  "properties": {
    "TotalRequests": {
      "type": "integer"
    },
    "AverageTime": {
      "type": "integer"
    }
  },
  "required": [
    "TotalRequests",
    "AverageTime"
  ]
}
```

The response looks like this:

{"TotalRequests":3,"AverageTime":5004625}

Example:

Test Case 7: Post request to /hash endpoint with shutdown

Scenario:

Post to /hash endpoint with data ‘shutdown’ shall shutdown with a 200 as a response

Prerequisites:

1. Install broken-hashserve app
2. Set PORT in the terminal by entering *export PORT=8088*
3. Start broken-hashserve app by entering *broken-hashserve* in the terminal
4. Execute the first test case, listed at the top of the first page, at least once, so that there is a testable response from the server

Steps:

1. Enter the following command in the terminal

   curl -X POST -d ‘shutdown’ http://127.0.0.1:8088/hash

Expected Behavior:

Example: