Code Test Assignment Preamble

Please follow the How-To section below for how to setup your devenvironment. The following tasks assume that the Docker devnode container is already started.

If all is setup properly, the following command should return "running":

docker inspect --format='{{.State.Status}}'
dev-node

```
Apples-iMac-74:app Apple$ docker images
REPOSITORY TAG IMAGE ID CREA
TED SIZE
demo/node latest 86dab9f72ade 35 m
inutes ago 201MB
```

```
Apples-iMac-74:app Apple$ docker inspect --format='{{.State.Status}}' dev-node running Apples-iMac-74:app Apple$
```

Note: in this file all commands that start with docker should be executed from the host machine. Any other commands should be run from inside a running instance of dev-node container.

Tasks

- 1. Fix all failing unit tests by implementing the required logic in the code under test to make all tests pass successfully.
- 2. Implement an API endpoint that adds a user to the database. Please analyze the mock data included in this project to determine which fields are required, and include the necessary data validation in your code.
- 3. Generate the test reports using: docker exec dev-node nyc npm test

```
running
Apples-iMac-74:app Apple$ docker exec dev-node nyc npm test
> app@0.0.0 test /app
> jasmine
Randomized with seed 21412
Started
2 specs, 0 failures
Finished in 0.559 seconds
Randomized with seed 21412 (jasmine --random=true --seed=21412)
File
             % Stmts
                       % Branch
                                    % Funcs
                                               % Lines
                                                         Uncovered Line #s
All files
               41.67
                               0
                                      22.22
                                                 43.48
 users.js
               41.67
                               0
                                      22.22
                                                 43.48
                                                        1... 37,39,40,43,46
Apples-iMac-74:app Apple$
```

4.

- 5. Package your response using the following command, and submit the generated submission.tgz as an attachment. docker exec dev-node tar -exclude=node modules -czvf
- 6. submission.tgz /app

```
app/spec/fixtures/
app/spec/fixtures/mockdata.json
app/.DS_Store
app/package.json
app/utils/
app/utils/db.js
app/app.js
app/routes/
app/routes/mocks.js
app/routes/index.js
app/routes/users.js
app/routes/tests.js
app/views/
app/views/layout.hbs
app/views/users.hbs
app/views/error.hbs
app/views/index.hbs
app/views/status.hbs
tar: removing leading '/' from member names tar: /app/submission.tgz: file is the archive; skipping
Apples-iMac-74:app Apple$
                                Ln 22, Col 14 Spaces: 2 UTF-8 LF

    Go Live
ot selected
                                                                        JavaSo
```

9. Note: the archive should contain your updated *.js files as well as the generated test reports.

file is attached.

7.

8.

submission.tgz

Good luck!

How-ToCreate development

environment

We have tested these instructions using Docker Version 17.12 1. Build the Docker image

```
docker build -t demo/node .
```

2. Start development docker instance (in background)

docker run -d --name dev-node -p
3000:3000 -v `pwd`/app:/app
 demo/node

3. Connect to development docker instance

docker exec -it dev-node /bin/sh Now you can you execute commands inside the docker instance: E.g.

node --version

Build and test the app

The following commands should be run from inside the dev-node docker container as needed. 1. Install app dependencies

npm install

2. Run all tests

npm test

3. Run and test the app

Start the app inside the container:

DEBUG=app:* npm start

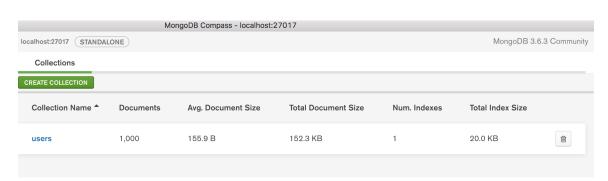
Test the app from a different terminal on the host machine: E.g.

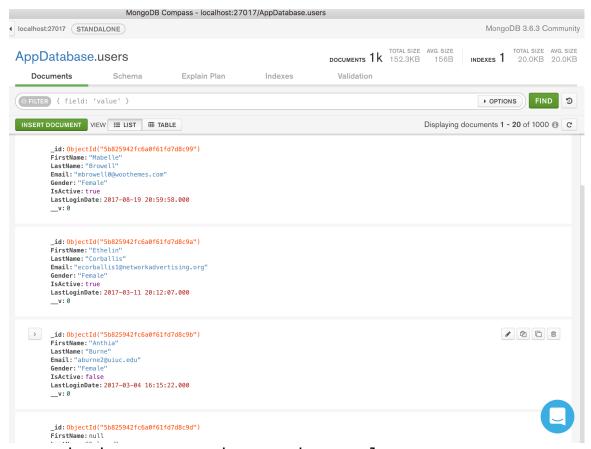
docker exec dev-node curl -v
localhost:3000/api/v1/users/
 setup-mock-data

i localhost:3000/api/v1/users/setup-mock-data

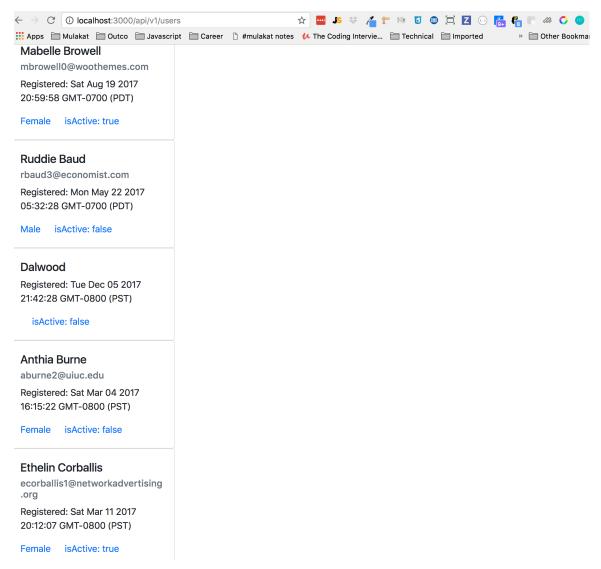


te":"2017-04-21T21:09:35.000Z","_id":"5b825943fc6a 0f61fd7d907d"} +1ms app:users Insert user into db: {"FirstName":"Flo renza", "LastName": "Cobby", "Email": "fcobbyrp@micros oft.com", "Gender": "Female", "IsActive": true, "LastLo ginDate":null,"_id":"5b825943fc6a0f61fd7d907e"} +0 app:users Insert user into db: {"FirstName":"Ced
ric","LastName":"Fried","Email":"cfriedrq@ftc.gov" ,"Gender":"Male","IsActive":false,"LastLoginDate": "2017-10-16T21:29:50.000Z","_id":"5b825943fc6a0f61 fd7d907f"} +0ms app:users Insert user into db: {"FirstName":"Gia como", "LastName": "Northen", "Email": "gnorthenr@map quest.com", "Gender": "Male", "IsActive": true, "LastLo ginDate":"2017-12-28T20:21:58.000Z","_id":"5b82594 3fc6a0f61fd7d9080"} +0ms app:users All mock data inserted. +380ms GET /api/v1/users/setup-mock-data 304 608.125 ms -





docker exec dev-node curl -v
localhost:3000/api/v1/users



or

Navigate to http://localhost:3000/api/v1/users in a browser on the host machine.