

Tng Liang Yi, Ailanthus A0190024U

Submission for Task B1, B2 and B3

Task B1

Link to github: <https://github.com/ailanthustng/taskb>

How to run the API locally:

- 1) Clone the repository using `git clone`.
- 2) Go into the directory using `cd taskb`.
- 3) Run `npm install`.
- 4) Run the application using `nodemon index`.

Available endpoints:

- 1) GET /api/contacts
- 2) POST /api/contacts
- 3) GET /api/contacts/:id
- 4) PUT /api/contacts/:id
- 5) DELETE /api/contacts/:id

Steps before using Postman on **Localhost**:

1. Select the Request you want (e.g PUT, GET).
2. For PUT & POST, select 'Body', then choose 'x-www-form-urlencoded'.

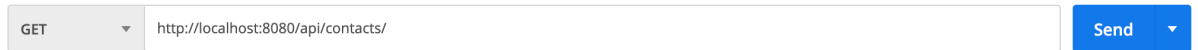
E.g:

The screenshot shows the Postman interface for a POST request to `http://localhost:8080/api/contacts/`. The 'Body' tab is selected, and the 'x-www-form-urlencoded' format is chosen. The body is a form with the following data:

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> name	bobby	
<input checked="" type="checkbox"/> email	bob@email.com	
<input checked="" type="checkbox"/> phone	98765432	
<input checked="" type="checkbox"/> gender	Male	

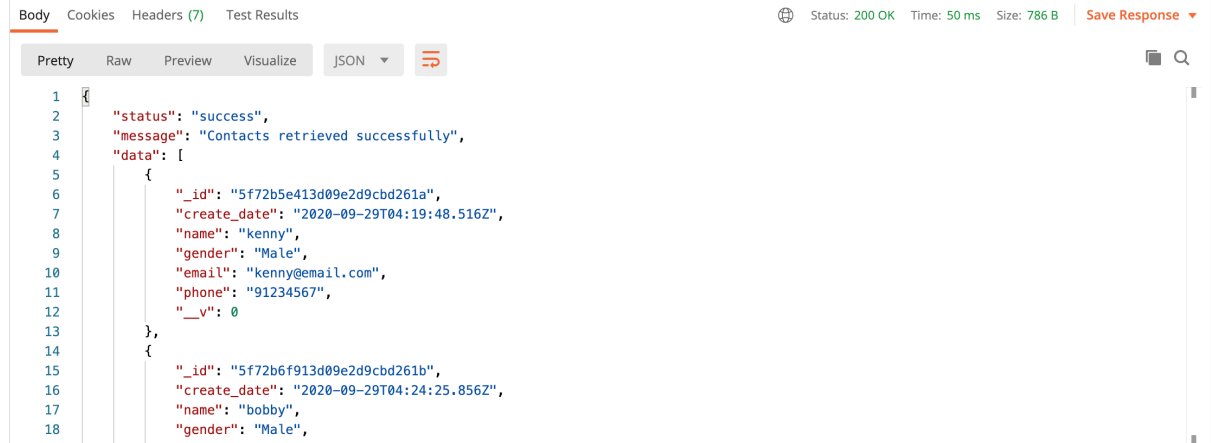
GET /api/contacts (to get **ALL** contacts)

- 1) Using Postman, enter <http://localhost:8080/api/contacts/> in the request URL box.

A screenshot of the Postman interface showing a GET request to the URL 'http://localhost:8080/api/contacts/'. The 'Send' button is visible on the right.

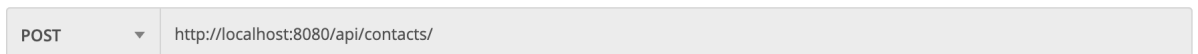
- 2) Press “Send”.
- 3) You will receive the response body with a success message.

E.g:

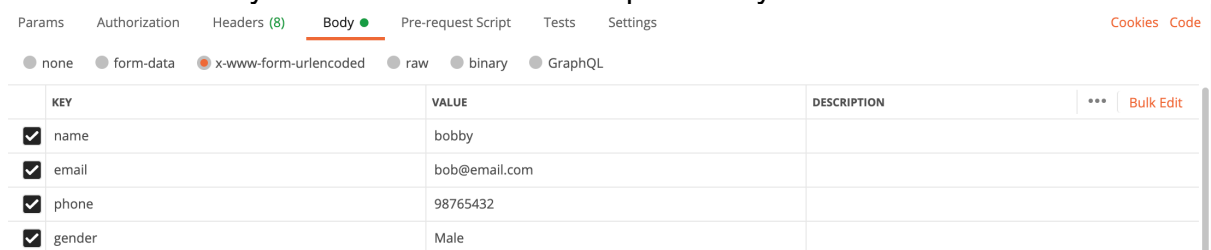
A screenshot of the Postman response body for the GET request. The status is 200 OK, time is 50 ms, and size is 786 B. The response is in JSON format, showing a success message and a list of two contacts: 'kenny' and 'bobby'.

POST /api/contacts

- 1) Using Postman, enter <http://localhost:8080/api/contacts/> in the request URL box.

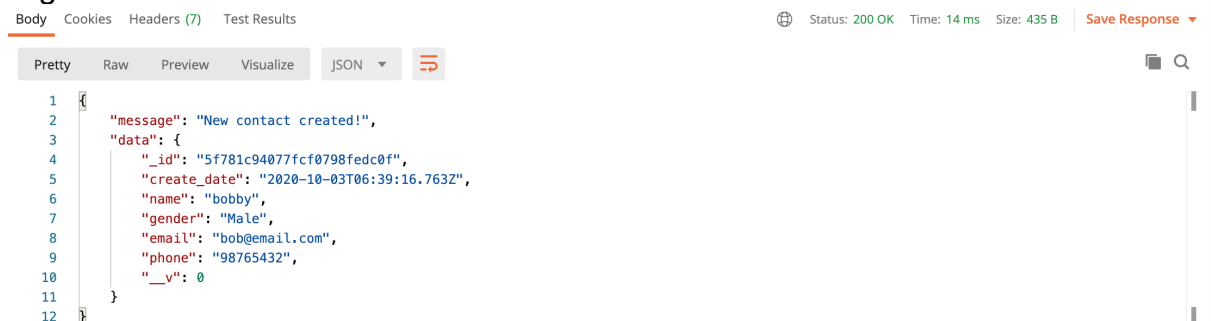
A screenshot of the Postman interface showing a POST request to the URL 'http://localhost:8080/api/contacts/'.

- 2) Enter the details you want to send in the request body.

A screenshot of the Postman request body for the POST request. The 'Body' tab is selected, and the 'x-www-form-urlencoded' radio button is chosen. A table shows the key-value pairs for the request body: name (bobby), email (bob@email.com), phone (98765432), and gender (Male).

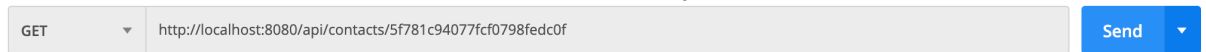
- 3) Press “Send”.
- 4) You will receive the response body with a success message.

E.g:

A screenshot of the Postman response body for the POST request. The status is 200 OK, time is 14 ms, and size is 435 B. The response is in JSON format, showing a success message and the details of the newly created contact: 'bobby'.

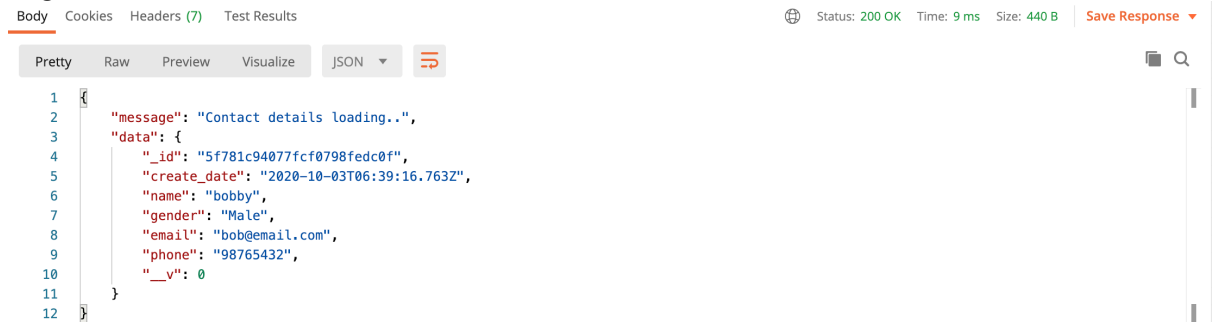
GET /api/contacts/:id (to get **individual** contacts)

- 1) Using Postman, enter <http://localhost:8080/api/contacts/:id> in the request URL box, where id can be retrieved from the GET request.



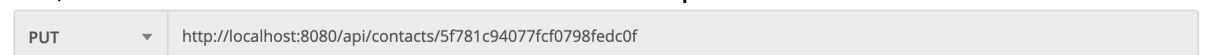
- 2) Press "Send".
- 3) You will receive the response body with a message.

E.g:

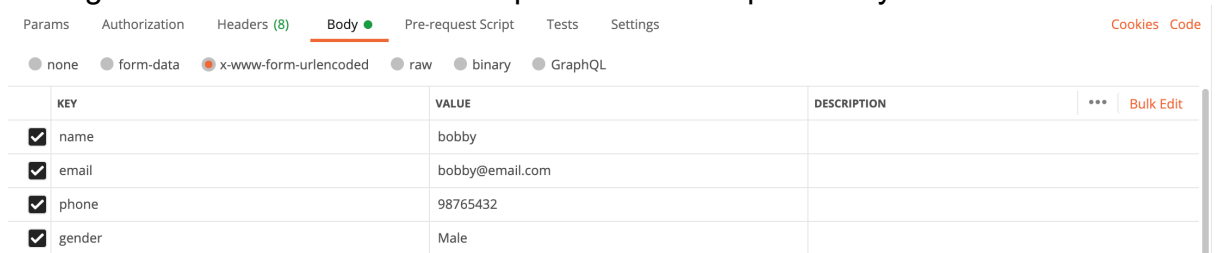


PUT /api/contacts/:id (to **update** individual contacts)

- 1) Using Postman, enter <http://localhost:8080/api/contacts/:id> in the request URL box, where id can be retrieved from the GET request.

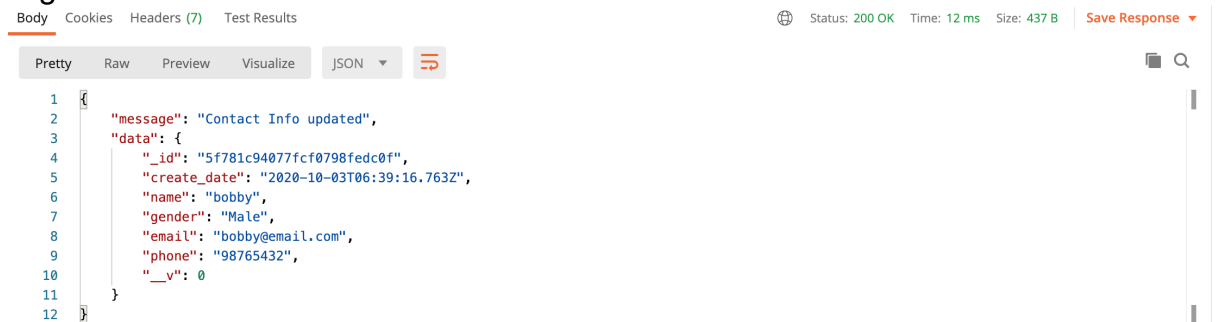


- 2) Change whatever is needed to be updated in the request body.



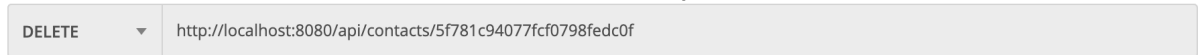
- 3) Press "Send".
- 4) You will receive the response body with a success message.

E.g:



DELETE /api/contacts/:id

- 1) Using Postman, enter <http://localhost:8080/api/contacts/:id> in the request URL box, where id can be retrieved from the GET request.



- 2) Press "Send".
- 3) You will receive the response body with a success message.

E.g:



Task B2

Task B2 reference: <https://www.digitalocean.com/community/tutorials/test-a-node-restful-api-with-mocha-and-chai>

In this task, I have utilized Mocha and Chai to write the test cases (under 'test/test.js').

I have also utilized Travis CI to automate the testing. Each time the code is pushed onto github, Travis is ran and the testing is done.

The testing can also be done manually by running `npm run test` in the command line.

My .travis.yml file is as such:

```
language: node_js
node_js:
  - stable

services:
  - mongodb

before_script:
  - npm install

script:
  - echo "Running tests"
  - mocha --exit
```

The `mocha --exit` line under the script portion is the command that runs the test file. Mocha automatically searches in its directory for a `test` folder and runs every test file in that folder.

The output in Travis is as such:

```
225 $ mocha --exit 0.61s
226 Db connected successfully
227 (node:5718) DeprecationWarning: current Server Discovery and Monitoring engine is deprecated, and will be removed in a future version.
    To use the new Server Discover and Monitoring engine, pass option { useUnifiedTopology: true } to the MongoClient constructor.
228 (Use `node --trace-deprecation ...` to show where the warning was created)
229 Running TaskB on port 8080
230
231
232 Contacts
233   /POST contact
234   ✓ it should POST a contact (42ms)
235   /GET contact
236   ✓ it should GET no contacts at the beginning
237   /GET at least 1 contact
238   ✓ it should GET all (>1) the contacts
239   /GET/:id contact
240   ✓ it should GET a contact by the given id
241   /PUT/:id contact
242   ✓ it should UPDATE a contact by the given id
243   /DELETE/:id contact
244 (node:5718) DeprecationWarning: collection.remove is deprecated. Use deleteOne, deleteMany, or bulkWrite instead.
245   ✓ it should DELETE a contact by the given id
246
247
248 6 passing (112ms)
249
250 The command "mocha --exit" exited with 0.
251 store build cache cache.2
252
253
254
255
256
257 Done. Your build exited with 0.
```

Task B3

This task communicates with the API that is deployed on AWS Lambda via Postman.

The steps are similar to those in Task B1 except that:

- 1) Use <https://utx8rwr4e4.execute-api.ap-southeast-1.amazonaws.com/dev/api/contacts> rather than <http://localhost:8080/api/contacts/>.
- 2) Use <https://utx8rwr4e4.execute-api.ap-southeast-1.amazonaws.com/dev/api/contacts/:id> rather than <http://localhost:8080/api/contacts/:id>.

My updated .travis.yml file is as such:

```
language: node_js
node_js:
  - stable

services:
  - mongodb

before_script:
  - npm install
  - npm install -g serverless
  - serverless config credentials --provider aws -k $AWS_ACCESS_KEY_ID -s $AWS_SECRET_ACCESS_KEY

script:
  - echo "Running tests"
  - mocha --exit
  - sls deploy

env:
  global:
    - secure: wVYxFd8RhvLFNtq/W68YGFub5j46aPAE7XcB4N7ntdi5n15zJn1Kkkm1wDPxTaCgwFkj9gUY5sebtPw7CnBXv/R5A+pu1I2hXIebi
    - secure: rQdb4awyug+mt3u2NVNruGRwtiXA7QWnwv9XXD/1gnnwKbkgX0mYPS2h2h5uLKwP826IX555T5BQlEi60geYMeSygePC3gaHr2b/e
```

The `serverless config` line configures the AWS credentials, and subsequently, `sls deploy` automatically deploys the API to AWS Lambda.