# Employee REST API

## Environment set up

Open a terminal and navigate to project folder to run the following

1. Install all the npm modules from package.json
   1. npm i
2. Set up environment variable
   1. Set port to 3000
      1. Set PORT=3000
   2. Set separate debuggers for levels debug and error in the app
      1. Set DEBUG=app:\*
   3. Set NODE\_ENV to development to look into the Morgan enabled cross cutting logging. Set it to production if you don’t wish to see the logging.
      1. Set NODE\_ENV=development

## Project info: <http://localhost:3000/readme.pdf>

1. Git hub project location for
   1. nodejs backend
      1. <https://github.com/smukherjee16/nodejs>
   2. Standalone frontend
      1. <https://github.com/smukherjee16/nodejs/tree/master/employee>
2. Project file
   1. Index.js – main file with all the employees endpoint implementations
   2. validations/validations.js – validations rules for request params
   3. tests/index.test.js – unit tests for mostly validations
   4. routes/emp.js – Tried to set up a route for the employee’s endpoints but running into issues. So left the implementations in index.js for now.
   5. Routes/home.js – this is a basic root endpoint route set up that works
   6. Employee/ - All the front end files for the stand alone ui.
      1. References:
         1. <https://www.tutorialrepublic.com/codelab.php?topic=bootstrap&file=crud-data-table-for-database-with-modal-form>
         2. <https://github.com/uxsolutions/bootstrap-datepicker>

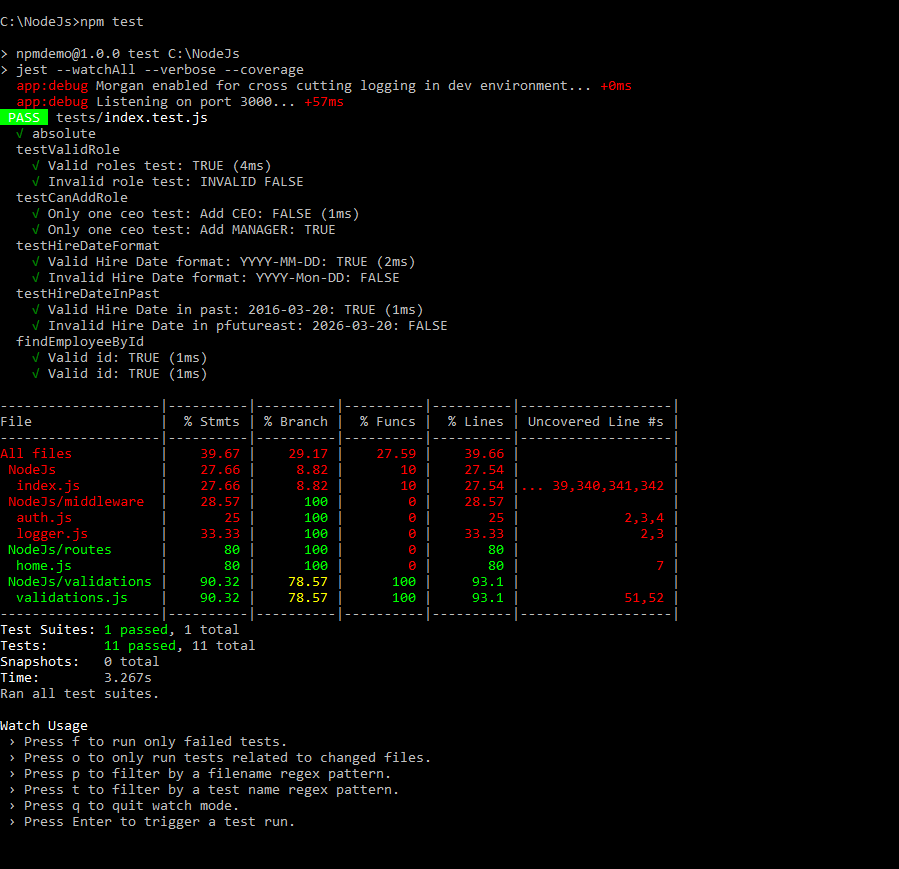
## Technologies/Tools

**Tools**: Visual Studio Code, Postman

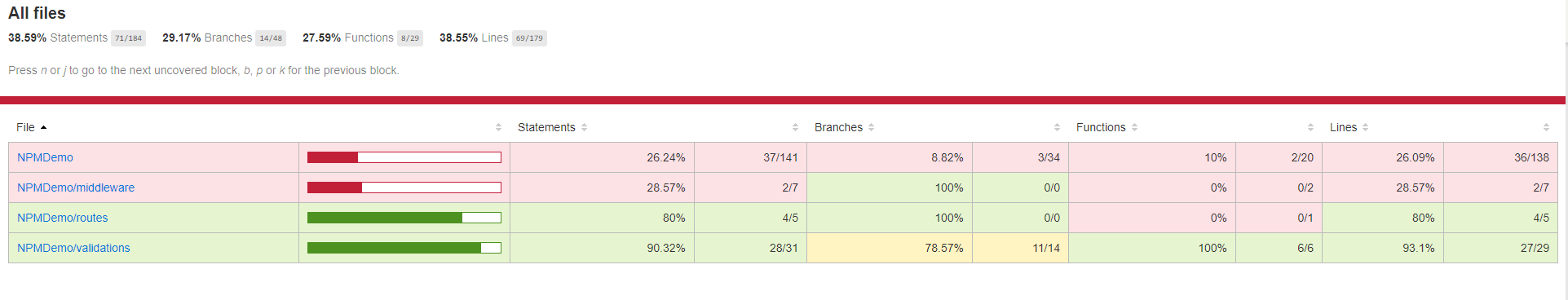
**Technologies**: Node.js, HTML, JavaScript, CSS, RESTful Webservices

## Testing Stats and Code Coverage

Run tests: npm test



Code Coverage



## Employee Endpoints

Navigate to project folder in Terminal and run nodemon index.js

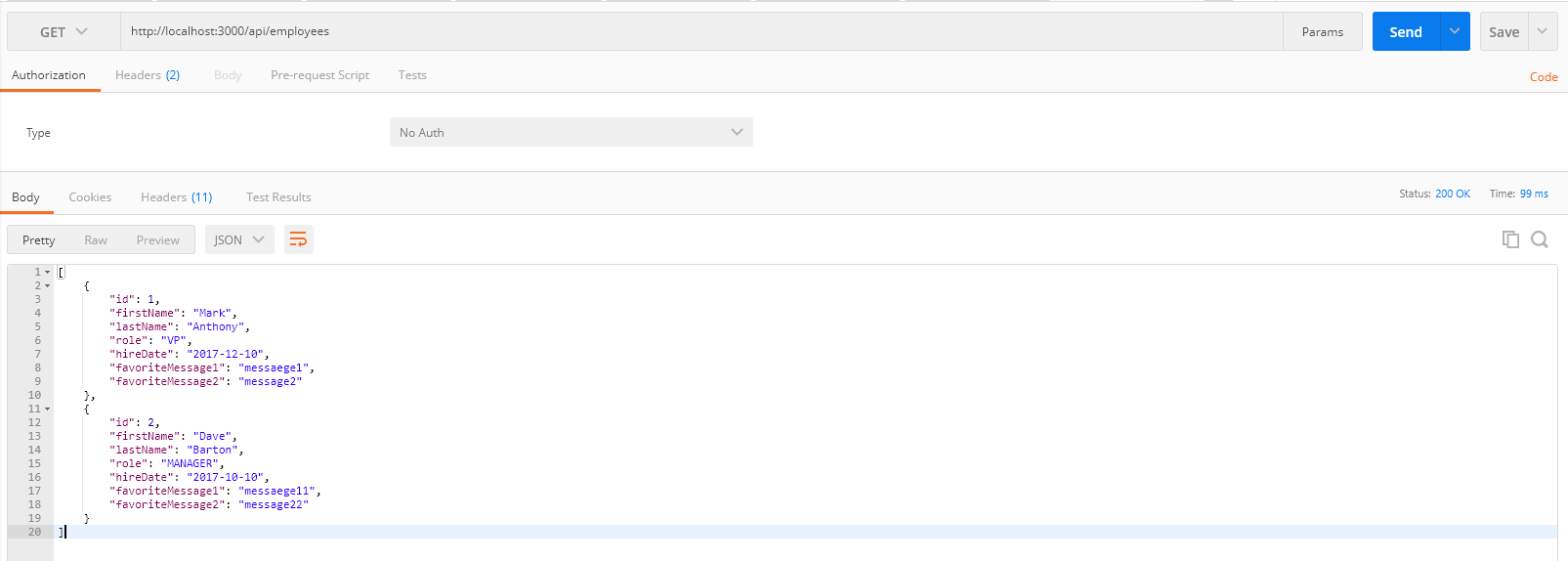
Task: Get all employee

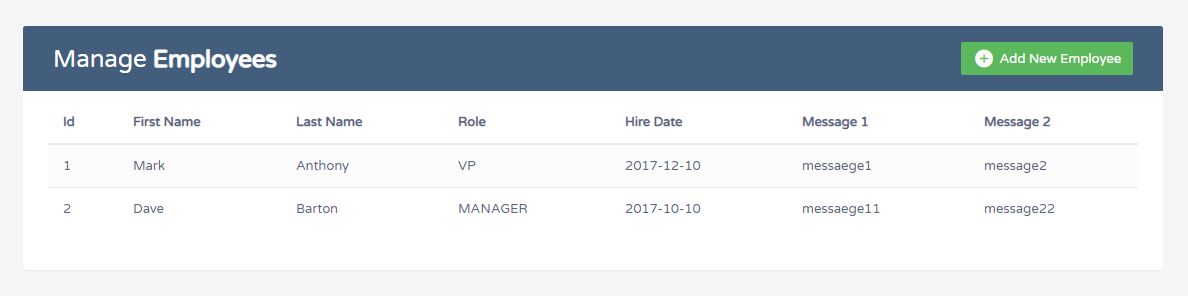
Endpoint: http://localhost:3000/api/employees

Http Method: GET

Responses:

Success: 200





Task: Get employee by id

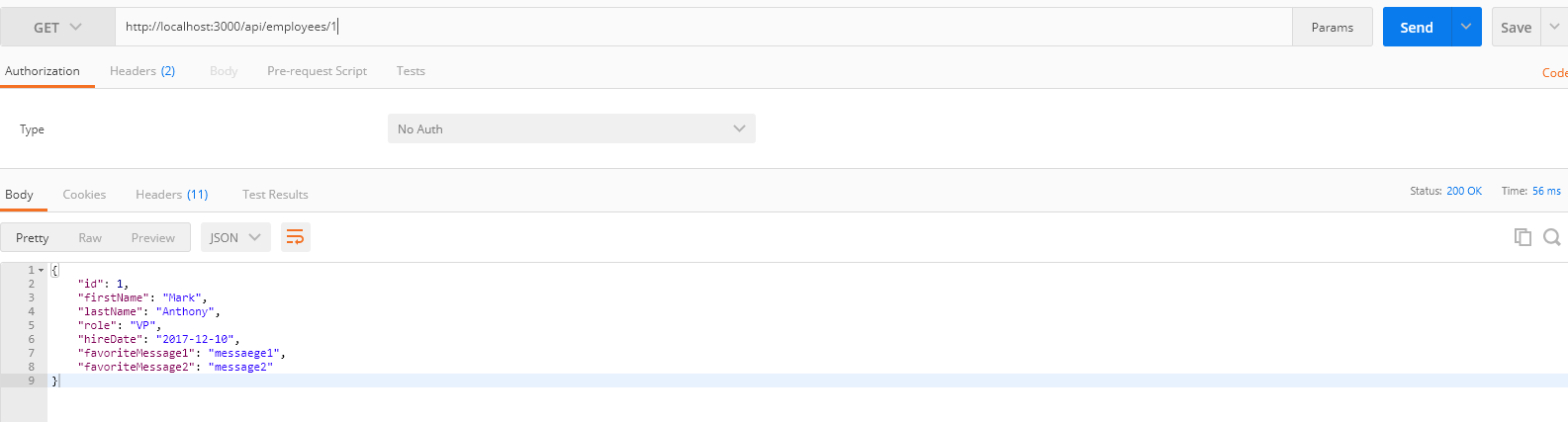
Endpoint: http://localhost:3000/api/employees/:id

Http Method: GET

Responses:

Id not found: 404

Success: 200



Task: Add new employee

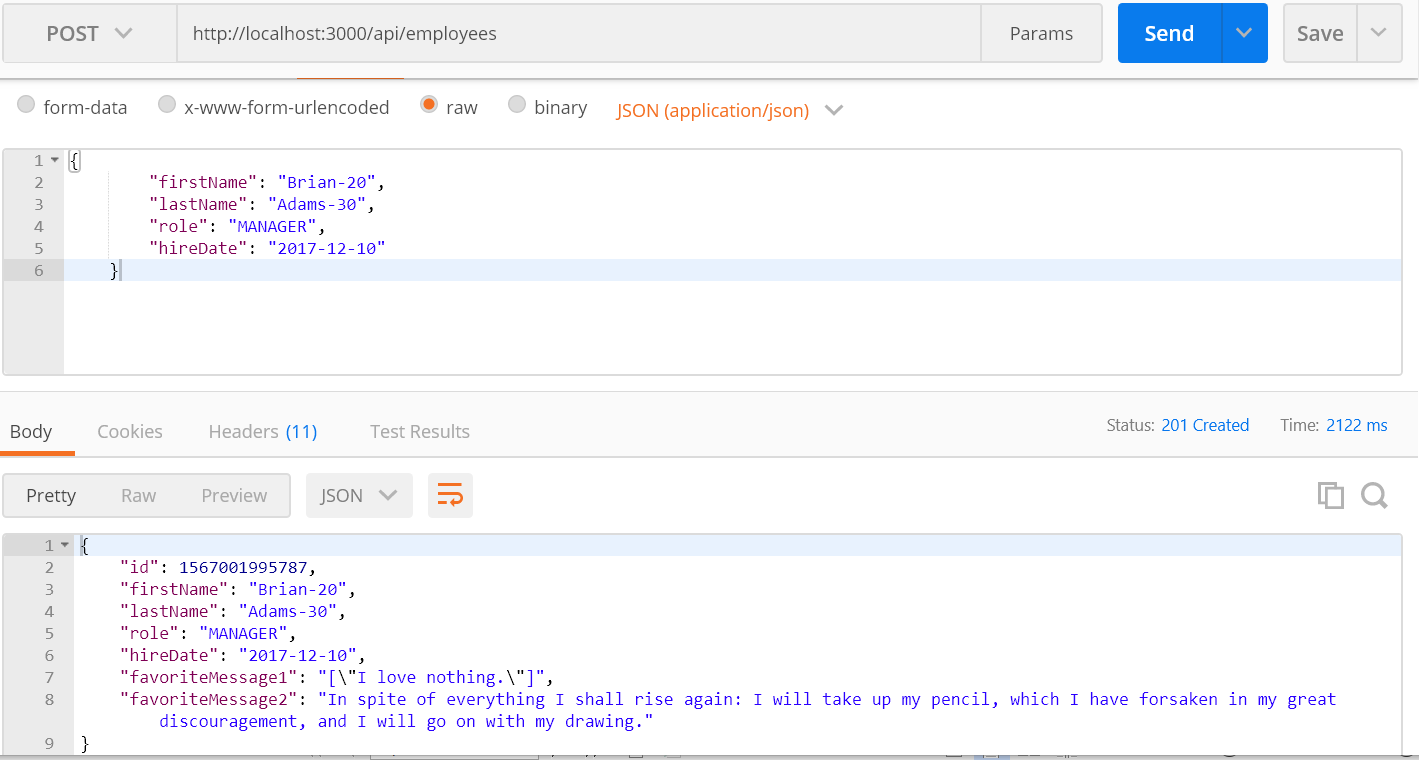
Endpoint: http://localhost:3000/api/employees

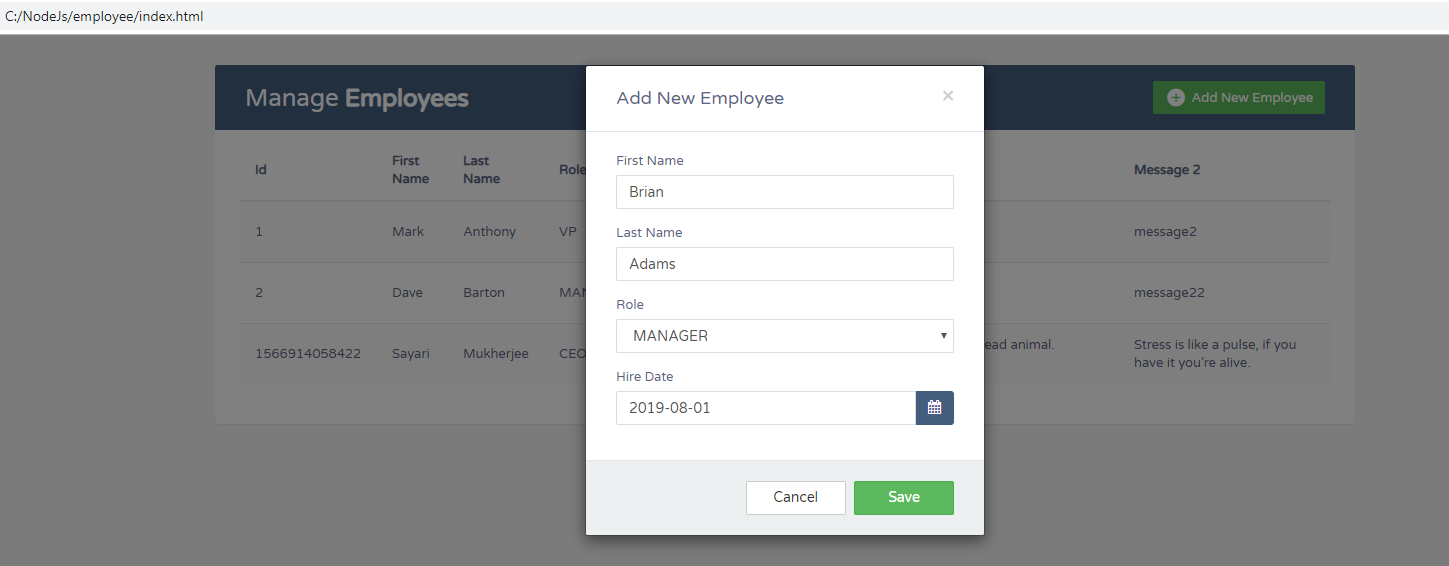
Http Method: POST

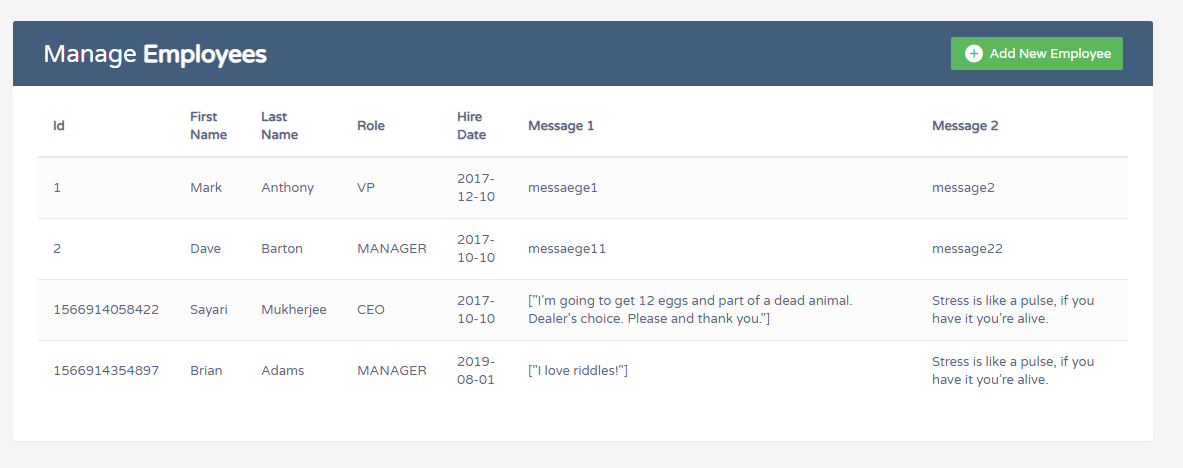
Responses:

Validation failure: 400

Success: 201







BAD REQUEST and ERROR HANDLING

Create a new record using a randomly generated value as the unique identifier (i.e. \_id field). Validate that the following fields are included in the POST body and have the right type/format as posted below:

- firstName (String)

- lastName (String)

- hireDate (YYYY-MM-DD format must be in the past)

- role (String) - must be one of the following (case-insensitive):

- CEO (can only be one of these)

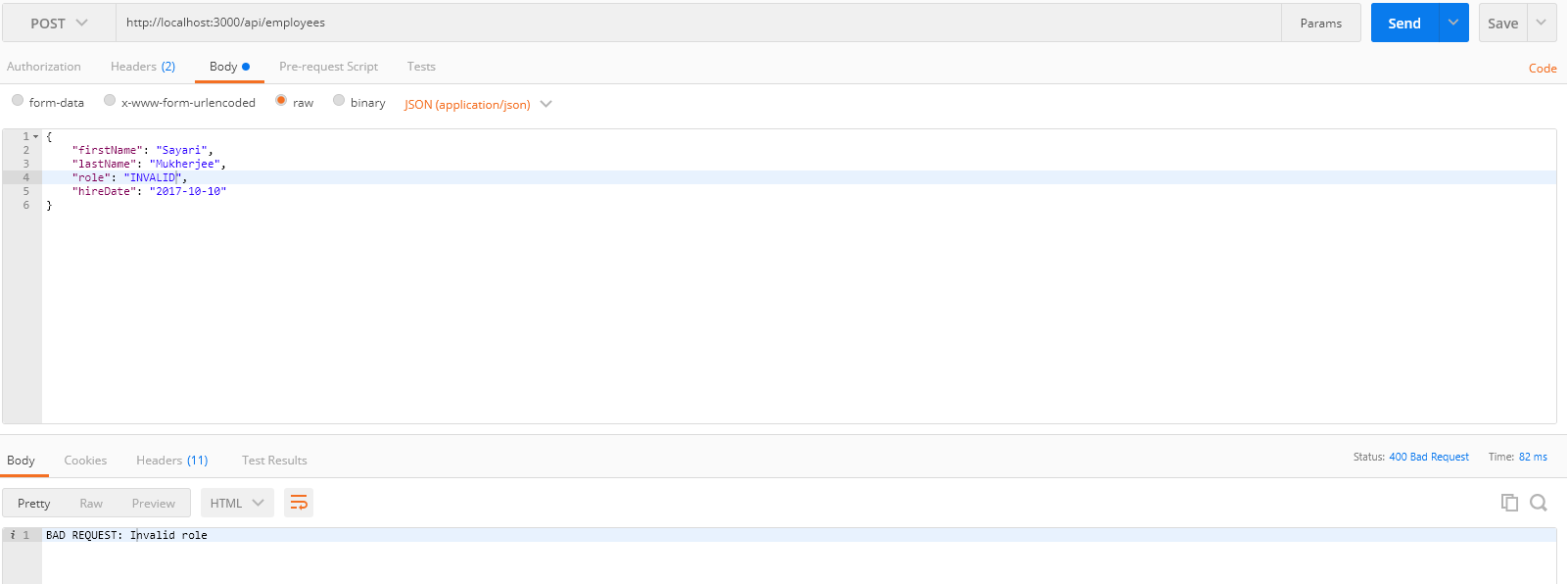
- VP

- MANAGER

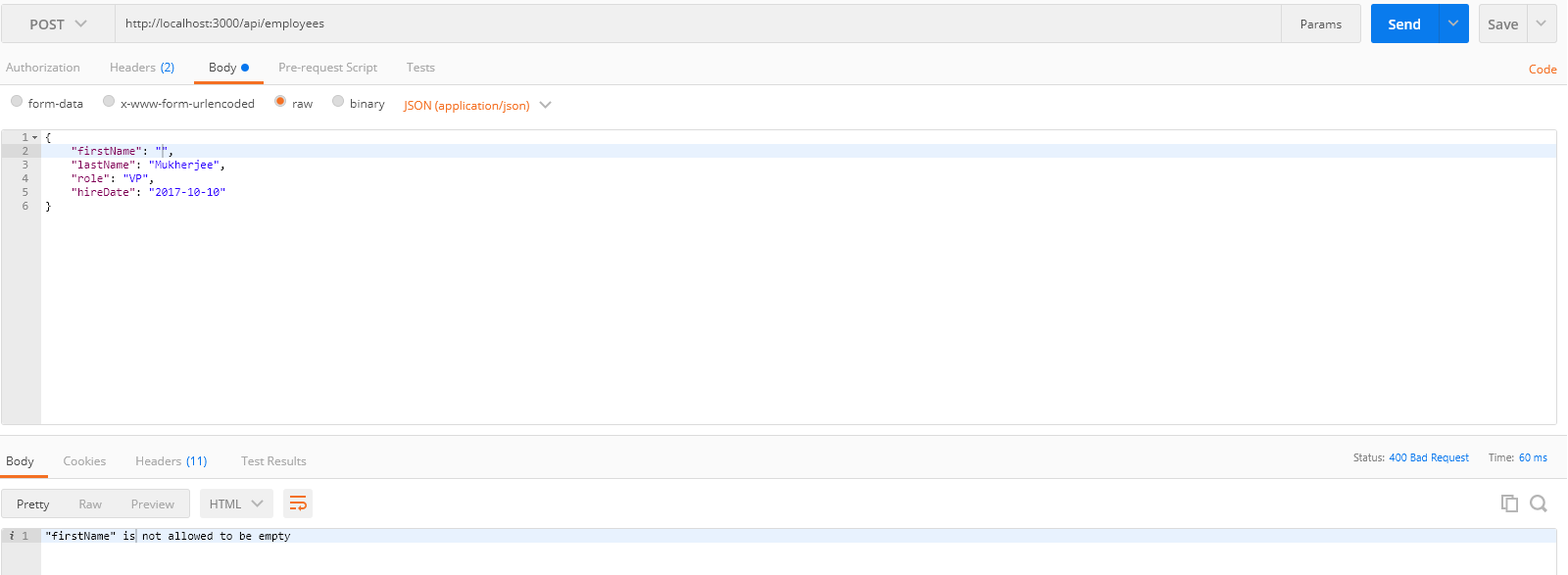
- LACKEY

Following validations are done for both POST and PUT

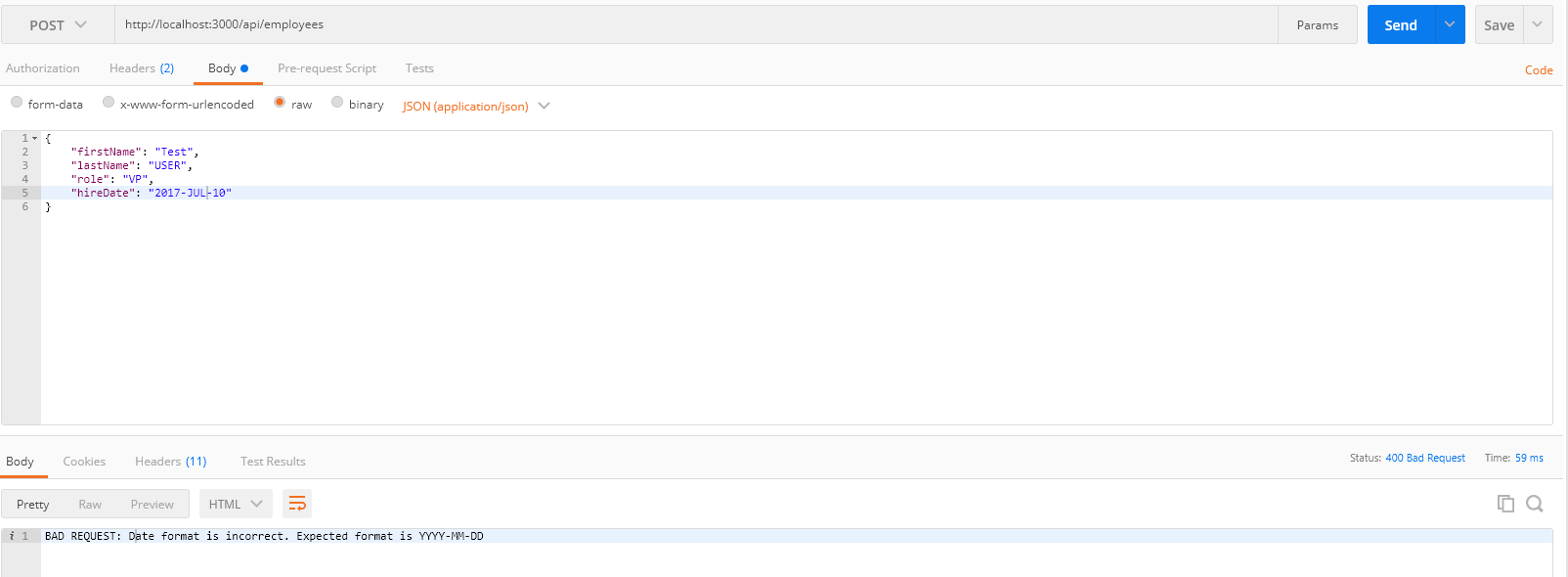
Invalid Role:



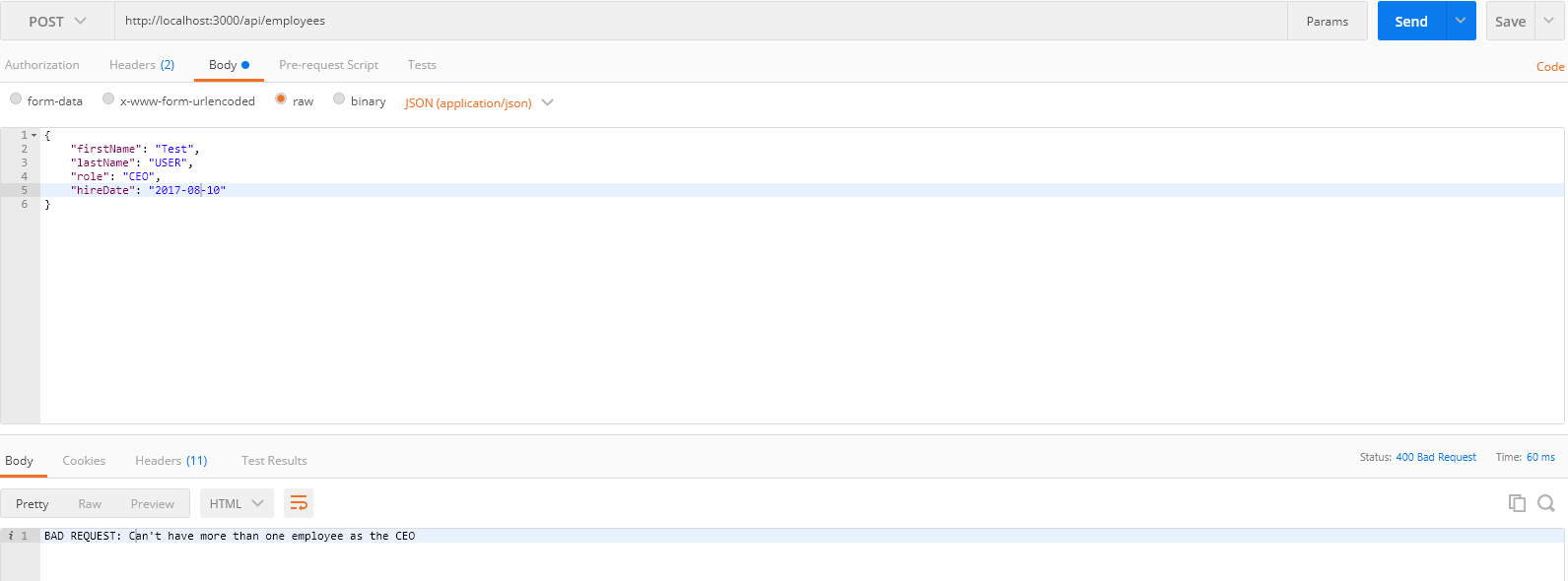
First Name/Last Name empty:



Invalid date format:



Only one CEO:



Task: Update employee by Path Param: id

Endpoint: http://localhost:3000/api/employees/:id

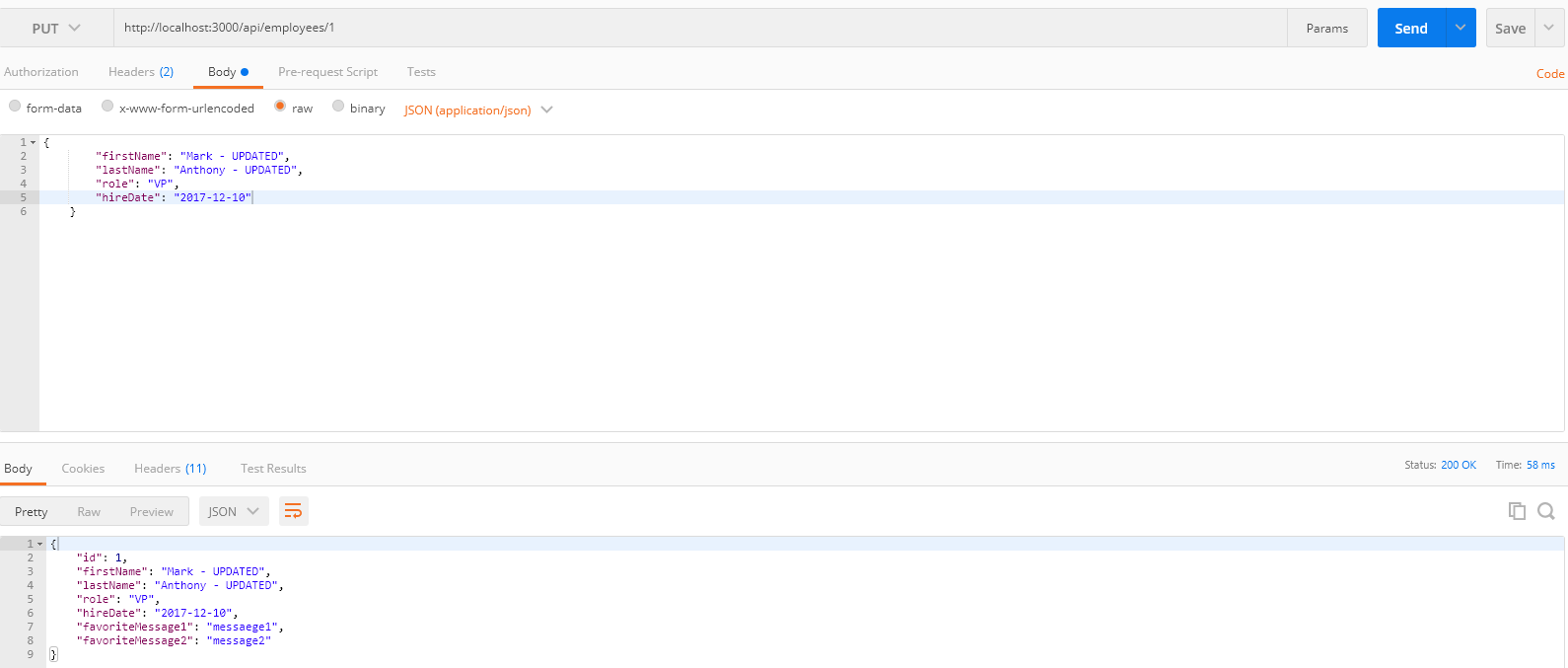
Http Method: PUT

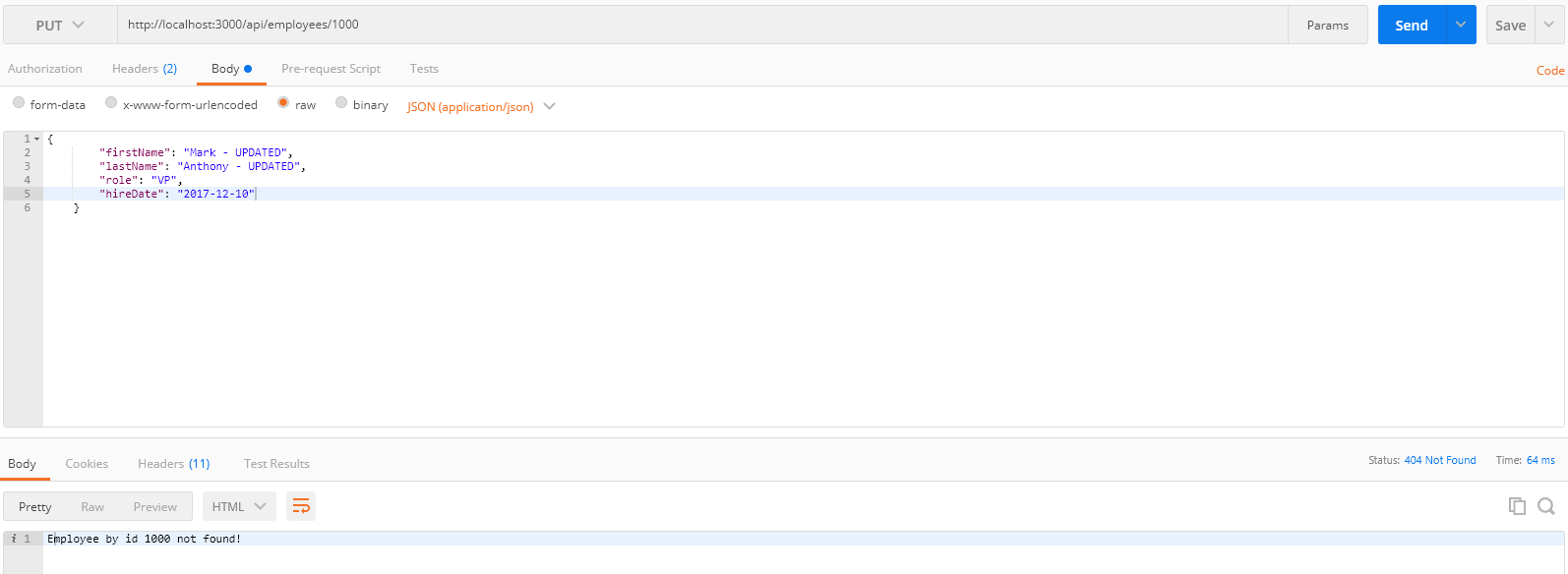
Responses:

Id not found: 404

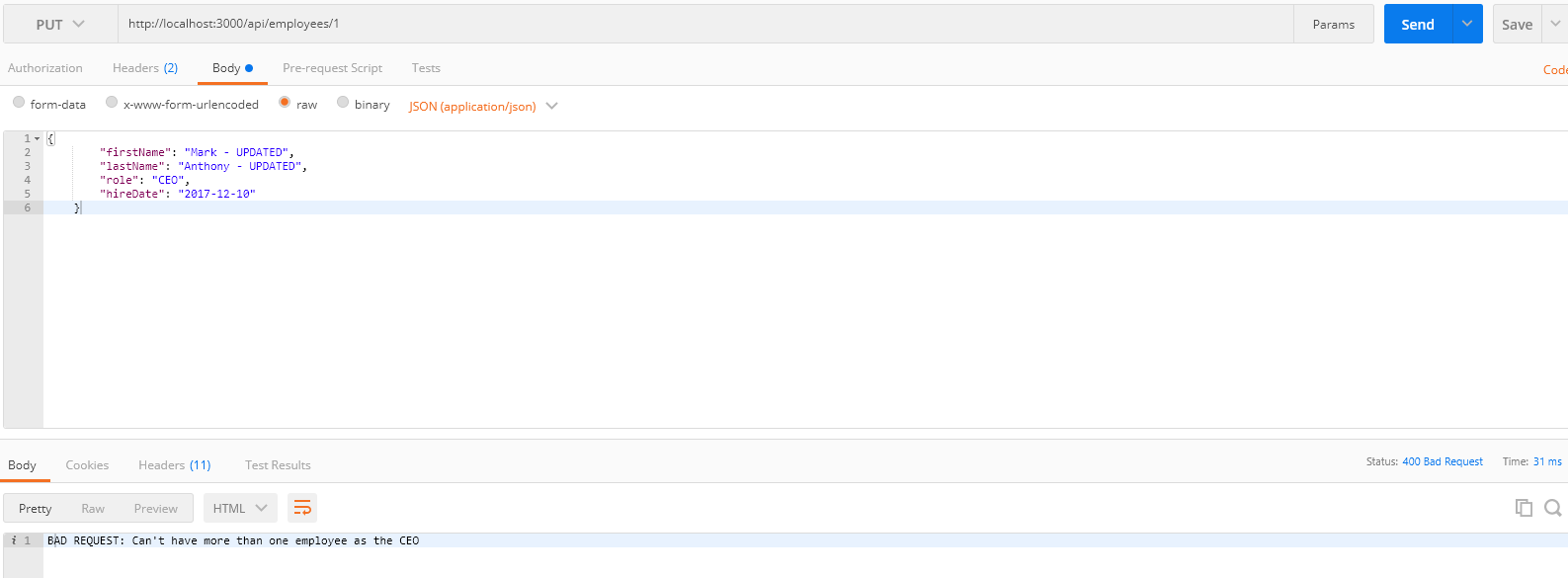
Validation failure: 400

Success: 200

Invalid Id:



Only one CEO:



Task: Delete employee by Path Param: id

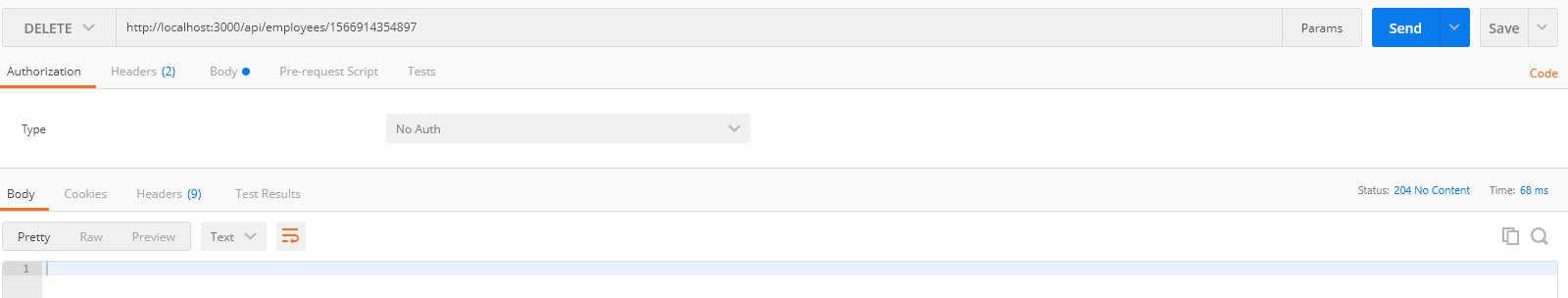
Endpoint: http://localhost:3000/api/employees/:id

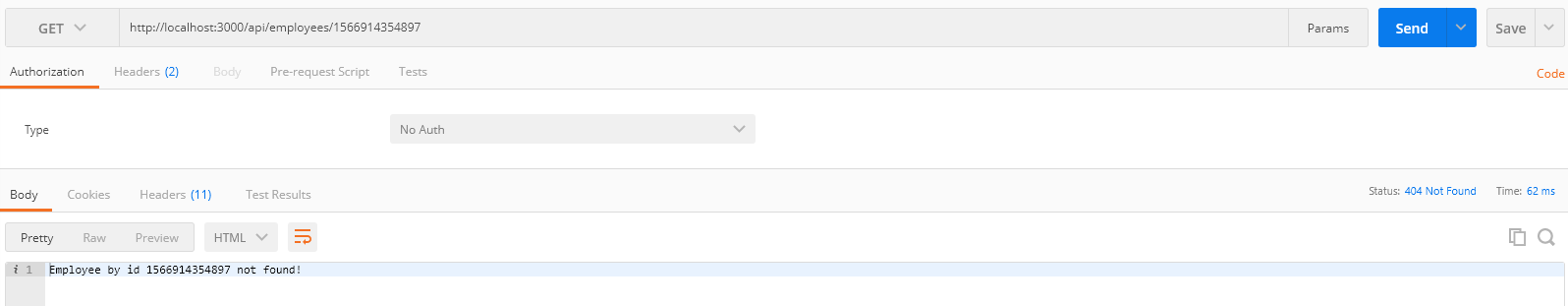
Http Method: DELETE

Responses:

Id not found: 404

Success: 204





Invalid Id:

