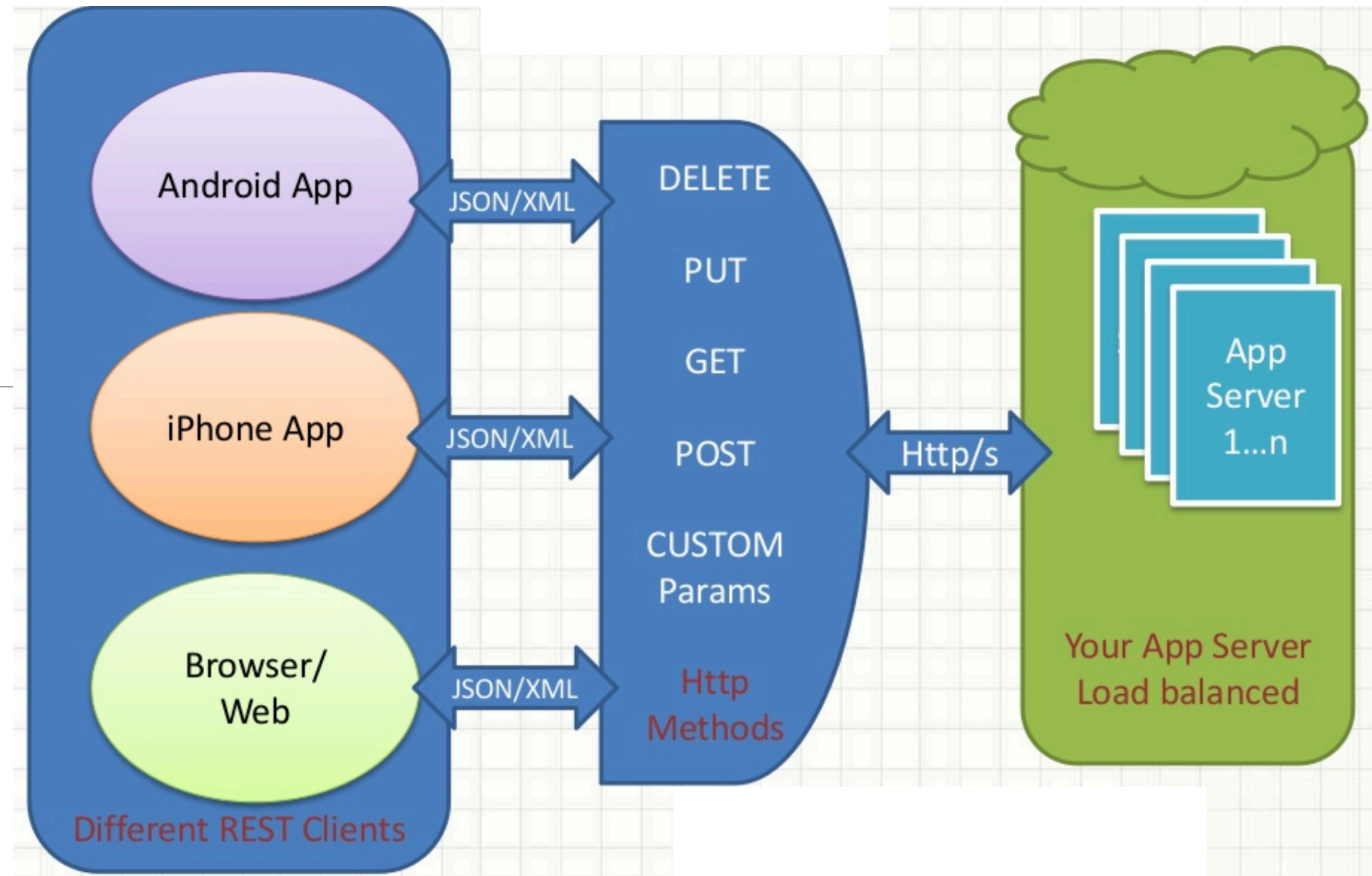


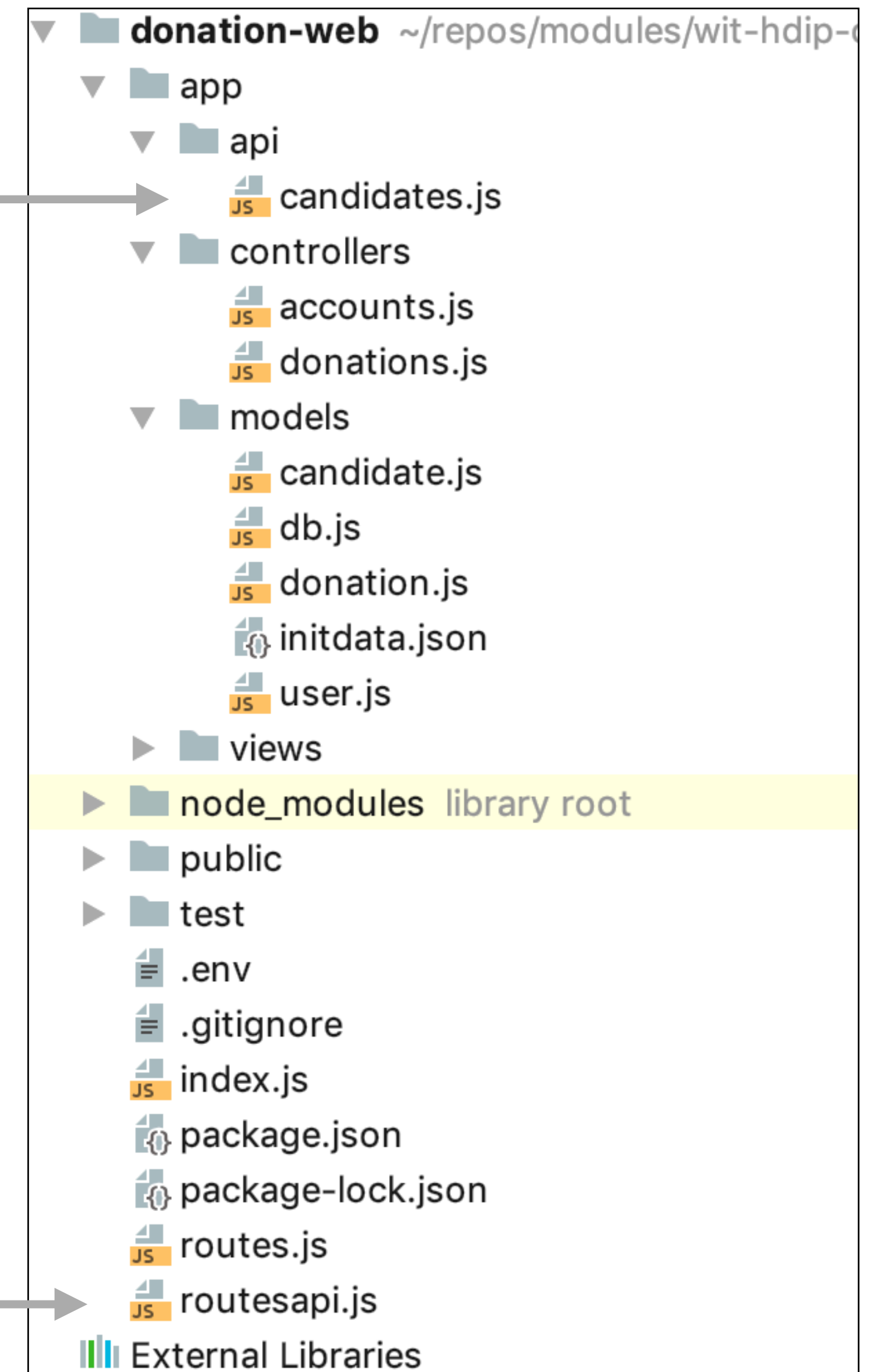
Endpoints



Project Structure

Create 'api' folder to expose endpoints from the application

Create 'routesapi.js' to define these endpoints



Get Candidates Endpoint

routeapi.js

```
const CandidatesApi = require('./app/api/candidates');

module.exports = [
  { method: 'GET', path: '/api/candidates', config: Candidates.find },
];
```

candidates.js

```
const Candidate = require('../models/candidate');

const Candidates = {

  find: {
    auth: false,
    handler: async function(request, h) {
      const candidates = await Candidate.find();
      return candidates;
    },
  },
};
```

index.js

```
server.route(require('./routes'));
server.route(require('./routesapi'));
await server.start();
```

localhost:4000/api/candidates x

Eamonn

←

→

↻

🏠

📄

localhost:4000/api/candidates

☆

🗨️

f?

🔍

📺

📱

🔧

JB

🌐

⋮

+ - [View source](#) ⚙️

```
[  
  - {  
    _id: "57b6b94d701cce8539bddea5",  
    firstName: "Lisa",  
    lastName: "Simpson",  
    office: "President",  
    __v: 0  
  },  
  - {  
    _id: "57b6b94d701cce8539bddea6",  
    firstName: "Donald",  
    lastName: "Simpson",  
    office: "President",  
    __v: 0  
  }  
]
```

[0]



boom

<https://github.com/hapijs/boom>

boom provides a set of utilities for returning HTTP errors. Each utility returns a `Boom` error response object (instance of `Error`) which includes the following properties:

- `isBoom` - if `true`, indicates this is a `Boom` object instance.
- `isServer` - convenience bool indicating status code ≥ 500 .
- `message` - the error message.
- `output` - the formatted response. Can be directly manipulated after object construction to return a custom error response.

Allowed root keys:

- `statusCode` - the HTTP status code (typically 4xx or 5xx).
- `headers` - an object containing any HTTP headers where each key is a header name and value is the header content.
- `payload` - the formatted object used as the response payload (stringified). Can be directly manipulated but any changes will be lost if `reformat()` is called. Any content allowed and by default includes the following content:
 - `statusCode` - the HTTP status code, derived from `error.output.statusCode`.
 - `error` - the HTTP status message (e.g. 'Bad Request', 'Internal Server Error') derived from `statusCode`.
 - `message` - the error message derived from `error.message`.
- inherited `Error` properties.

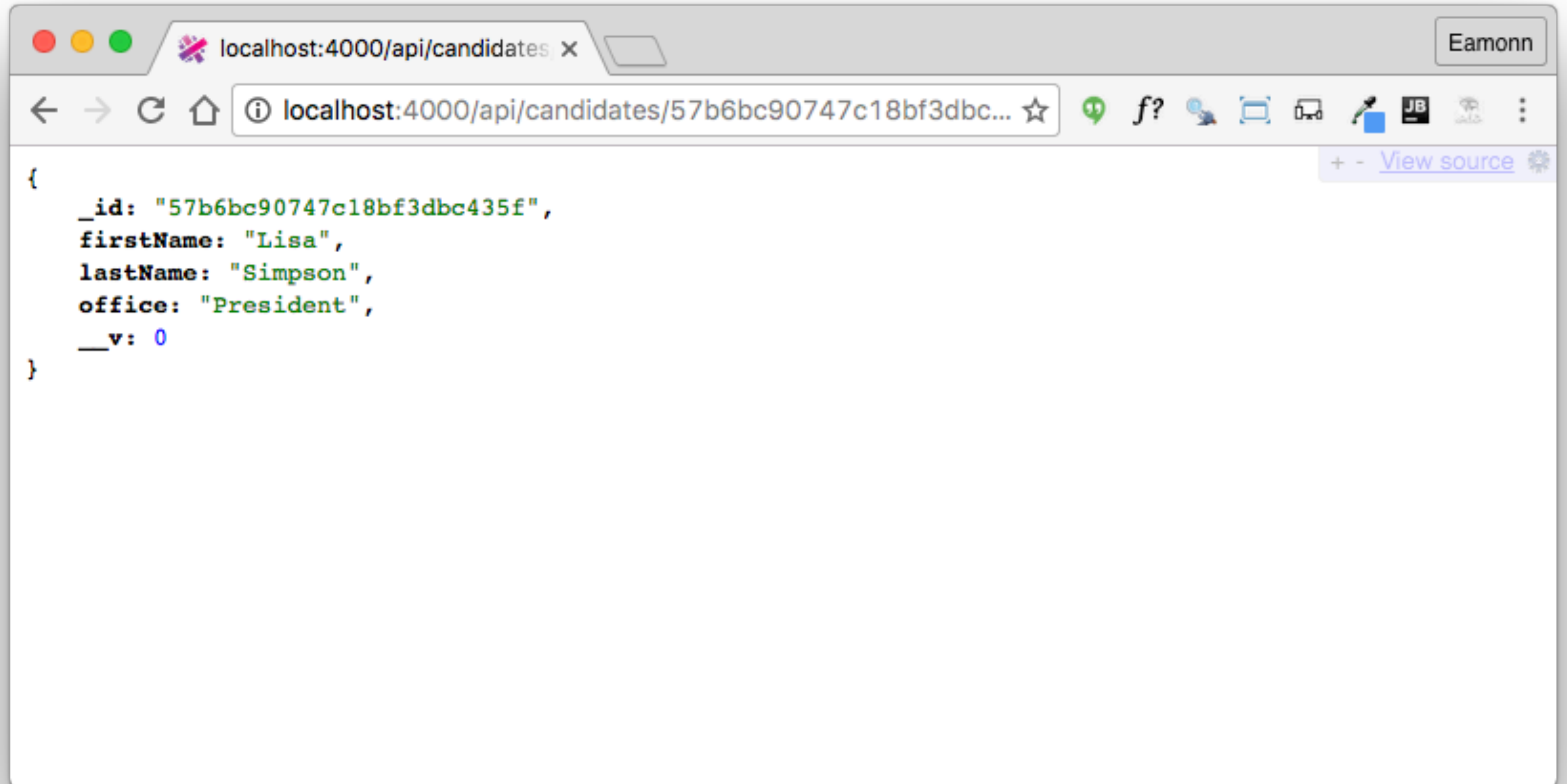
Get Candidate (singular) Endpoint

```
{ method: 'GET', path: '/api/candidates/{id}', config: Candidates.findOne },
```

- Object ID passed as part of path

```
findOne: {
  auth: false,
  handler: async function(request, h) {
    try {
      const candidate = await Candidate.findOne({ _id: request.params.id });
      if (!candidate) {
        return Boom.notFound('No Candidate with this id');
      }
      return candidate;
    } catch (err) {
      return Boom.notFound('No Candidate with this id');
    }
  },
},
```

Get Candidate (singular) Endpoint



Boom Error Messages



Exercising Endpoints: Chrome

localhost:3000/api/candidates

localhost:3000/api/candidates

View source

```
[
  - {
    _id: "5c73a7d12f9f9d5019458575",
    firstName: "Lisa",
    lastName: "Simpson",
    office: "President",
    __v: 0
  },
  - {
    _id: "5c73a7d12f9f9d5019458576",
    firstName: "Donald",
    lastName: "Simpson",
    office: "President",
    __v: 0
  },
  - {
    _id: "5c73a8742f9f9d501945857a",
    firstName: "Bart",
    lastName: "Simpson",
    office: "President",
    __v: 0
  }
]
```

Elements Console Sources Network Performance Memory Application Security Audits

View: [Icons] Group by frame Preserve log Disable cache Offline Online

Filter Hide data URLs All XHR JS CSS Img Media Font Doc WS Manifest Other

10 ms 20 ms 30 ms 40 ms 50 ms 60 ms 70 ms 80 ms 90 ms 100 ms 110 ms

Name

× Headers Preview Response Cookies Timing

candidates

jsonview-core.css

options.png

▼ [,...]

▶ 0: {_id: "5c73a7d12f9f9d5019458575", firstName: "Lisa", lastName: "Simpson", office: "President", __v: 0}

▶ 1: {_id: "5c73a7d12f9f9d5019458576", firstName: "Donald", lastName: "Simpson", office: "President",...}

▶ 2: {_id: "5c73a8742f9f9d501945857a", firstName: "Bart", lastName: "Simpson", office: "President", __v: 0}

3 requests | 3.6 KB transferred | Finish: 64 ms | DOMContentLoaded...

Exercising Endpoints: Postman

The screenshot displays the Postman application interface. The top bar includes navigation buttons like 'New', 'Import', 'Runner', and 'My Workspace'. The left sidebar shows a 'History' tab with a list of requests, including a GET request to 'http://localhost:3000/api/candidates/5c725c911dc3a4f63bb60240'. The main panel shows the details of this request, including the URL, method (GET), and response status (200 OK). The response body is displayed in JSON format, showing a candidate object with fields like '_id', 'firstName', 'lastName', 'office', and '__v'.

GET http://localhost:3000/api/candidates/5c725c911dc3a4f63bb60240

Send Save

Params Authorization Headers Body Pre-request Script Tests Cookies Code Comments (0)

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (6) Test Results Status: 200 OK Time: 29 ms Size: 298 B Download

Pretty Raw Preview JSON

```
1 {
2   "_id": "5c725c911dc3a4f63bb60240",
3   "firstName": "Lisa",
4   "lastName": "Simpson",
5   "office": "President",
6   "__v": 0
7 }
```

More Candidate Routes

```
const Candidates = require('./app/api/candidates');

module.exports = [
  { method: 'GET', path: '/api/candidates', config: Candidates.find },
  { method: 'GET', path: '/api/candidates/{id}', config: Candidates.findOne },
  { method: 'POST', path: '/api/candidates', config: Candidates.create },
  { method: 'DELETE', path: '/api/candidates/{id}', config: Candidates.deleteOne },
  { method: 'DELETE', path: '/api/candidates', config: Candidates.deleteAll },
];
```

- Get all Candidates
- Get One Candidate
- Create One Candidate
- Delete One Candidate
- Delete All Candidates

create a Candidate

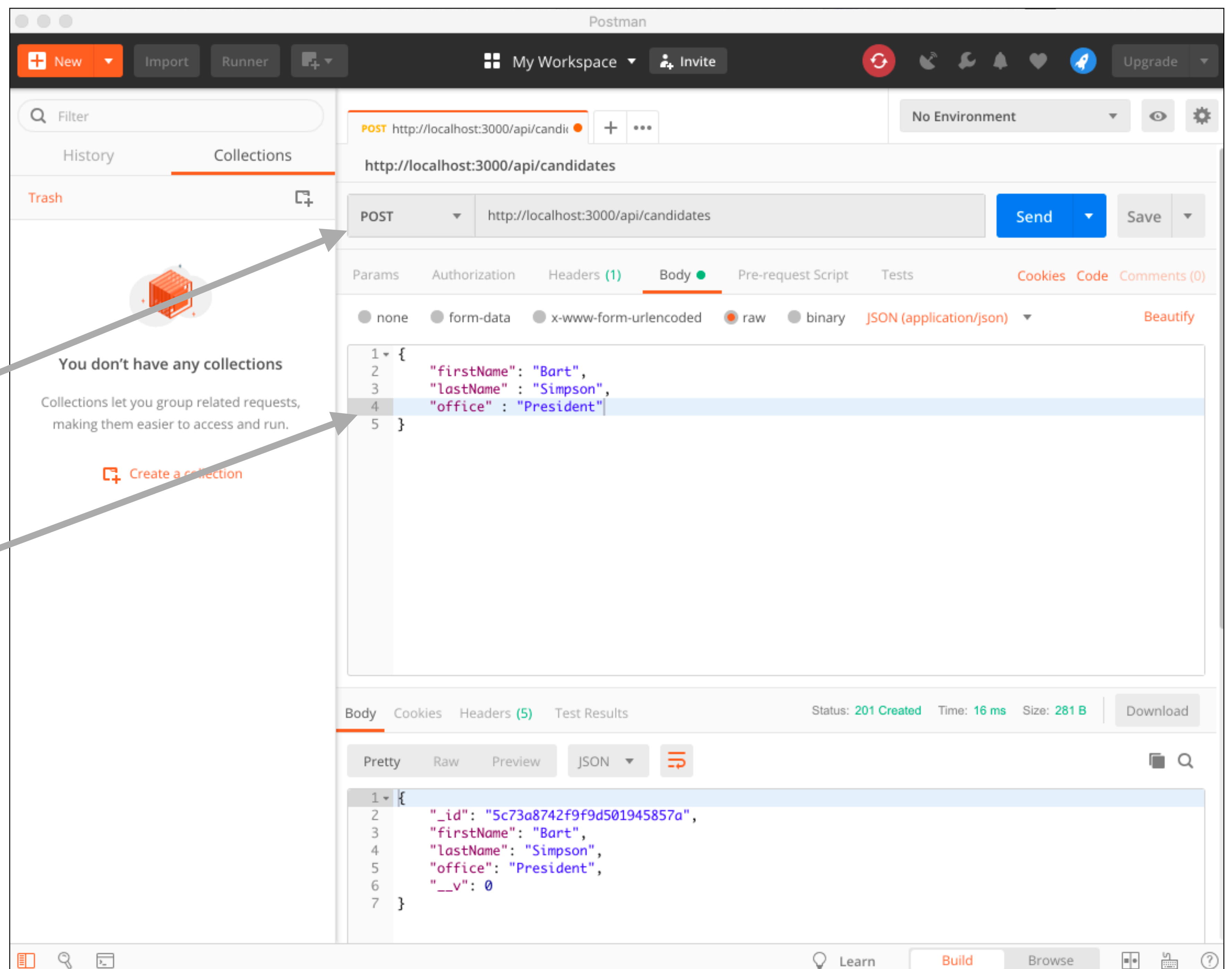
- POST instead of GET

```
{ method: 'POST', path: '/api/candidates', config: Candidates.create },
```

```
create: {  
  auth: false,  
  handler: async function(request, h) {  
    const newCandidate = new Candidate(request.payload);  
    const candidate = await newCandidate.save();  
    if (candidate) {  
      return h.response(candidate).code(201);  
    }  
    return Boom.badImplementation('error creating candidate');  
  },  
},
```

- Create a new Candidate, and return it

- Browser cannot be used to conveniently send a 'POST' request.
- Use Postman instead.
- POST
- Json Body (candidate)



Delete a single candidate

```
{ method: 'DELETE', path: '/api/candidates/{id}', config: Candidates.deleteOne },
```

```
deleteOne: {  
  auth: false,  
  handler: async function(request, h) {  
    const candidate = await Candidate.remove({ _id: request.params.id });  
    if (candidate) {  
      return { success: true };  
    }  
    return Boom.notFound('id not found');  
  }  
}
```

Delete all candidates

```
{ method: 'DELETE', path: '/api/candidates', config: Candidates.deleteAll },
```

```
deleteAll: {  
  auth: false,  
  handler: async function(request, h) {  
    await Candidate.remove({});  
    return { success: true };  
  },  
},
```