



Coding
Bootcamps



Intro to Node.js, Express.js and MongoDB

By Kaustubh Ghadge from [Coding Bootcamps](https://codingbootcamps.com)

Hello!

**I am Kaustubh Ghadge
("KG")**

A senior instructor at
coding-bootcamps.com



Session Outline

1. Recap of previous sessions
2. Refresher on REST API
3. Building RESTful API

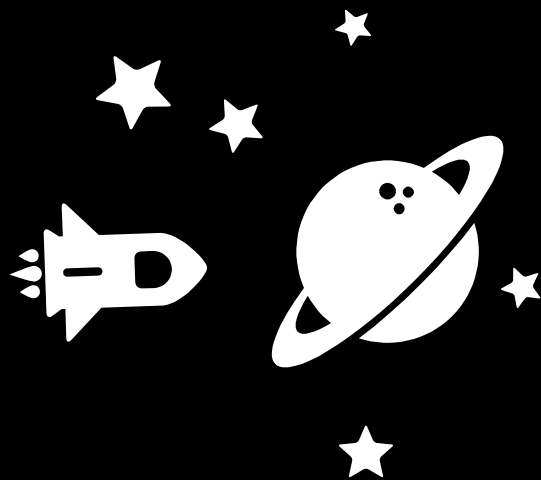
RECAP

1.

Refresher on REST API

REST API

- REST stands for REpresentational State Transfer.
- When a RESTful API is called, the server will transfer to the client a representation of the state of the requested resource.



PROJECT

Building RESTful API using Node.js and
MongoDB

Requirements

You must have all the necessary tools installed in previous session. However, if you haven't here are tools we will be using -

- Install Node.js by following the appropriate guidelines for your local machine given [here](#).
- You may use [Node Version Manager](#) to install multiple versions of Node.js on your local machine.
- Signup for [MongoDB Atlas](#) which is a cloud database as a service.
- Code Editor like Visual Studio Code or Sublime Text

JavaScript Training

You should take the below course first if you are new to JS:

- [Intro to JavaScript](#)

MongoDB Training

We offer the below courses for in-depth database design and MongoDB:

- Introduction to Database Design
- Introduction to No-SQL Database Design

Project Scope

Create a simple API that stores personnel records for an accounting department in our company

Project Source Files

GitHub Link -

<https://github.com/kaustubhghadge/coding-bootcamp/tree/master/node-mongo-restful-api>

Project Overview

1. Project Initialization
2. Install Application Dependencies
3. Run the Application
4. Test the Application
5. Establish Connection with MongoDB
6. Build REST API Endpoints
7. Put Things Together

Project Initialization

- Create an application folder
- Create **package.json** file using **npm**

Install Application Dependencies

- Create main application file - **app.js**
- Install following application dependencies -
 - [Express.js](#) - A node.js framework
 - [MongoDB Driver](#) - official module provided by the MongoDB team to help our Node.js application communicate with MongoDB.
 - Body-parser - This package will allow us to handle request bodies with Express.

What is body-parser?

Let's say you are sending an HTML form data to Node.js server i.e. you made a request to the server. HTTP sends your form data in bits and pieces which are intended to get assembled as they reach their destination. To extract these bits and pieces of data and assemble it so it is in a format that can be useful, we use a body-parser middleware.

Run the application

- Import the downloaded dependencies
- Initialize **Express.js** framework
- Define the **port** on which you want to run the application - e.g. 5000
- Configure **body-parser** middleware

Test the application

- Run the application by running command - **node app.js**
- The server will be listening on port 5000 for requests according to the boilerplate we wrote.

Establish Connection with MongoDB

- Get the connection string from MongoDB Atlas
- Add the string to **app.js**
- Create a MongoDB **collection** property - **personnel** to store **personnel** records
- Run the App again to see if there are any connection errors

Inserting single record

- We create this endpoint to enter personnel records into MongoDB
- Write a **POST** endpoint using Express method `app.post()`
- Insert a **personnel** record into the defined collection using `collection.insert()` method

Test the new endpoint

Let's test this out using **cURL**, a command-line tool for transferring data and supports HTTP; a very good ad-hoc tool for testing REST services.

```
curl -X POST \  
    -H 'content-type:application/json' \  
    -d '{"firstname":"John","lastname":"Doe"}' \  
    http://localhost:5000/personnel
```

Understanding the cURL command

curl is used in command lines or scripts to transfer data. It is also used in cars, television sets, routers, printers, audio equipment, mobile phones, tablets, set top boxes, media players and is the internet transfer backbone for thousands of software applications affecting billions of humans daily.

Learn more - <https://curl.haxx.se/>

Retrieving all records

- We create an endpoint to retrieve all the records data.
- Write a **GET** endpoint using Express method `app.get()`
- The goal here is to return all data in our collection representing people.
- Get all **personnel** records from the given collection using `collection.find()` method

Test the new endpoint

Let's test this out using **cURL**, a command-line tool for transferring data and supports HTTP.

```
curl -X GET http://localhost:5000/personnel
```


Retrieving a single record

- We create an endpoint to retrieve a single record data.
- Write a **GET** endpoint using Express method **app.get()**
- Get a single **personnel** record from the given collection using **collection.findOne()** method
- We will need to retrieve a single record according to its ID

Test the new endpoint

Let's test this out using **cURL**, a command-line tool for transferring data and supports HTTP.

```
curl -X GET
```

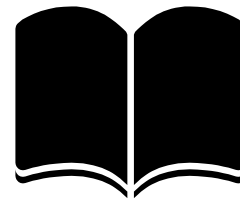
```
http://localhost:5000/personnel/4b103f89403f841059524fd1
```

Here [4b103f89403f841059524fd1](#) is the **ObjectID**, this will change per database.

Following best practices

- Use [POSTMAN](#) instead of curl
- Use environment variables to define -
 - Port
 - Connection URL
- Use [Swagger](#) to document your API

**Putting all things
together**



Resources

Reading

- Curl - <https://curl.haxx.se/>
- RESTFUL API - <https://restfulapi.net/>

Next Session

Project 2 - Building User Authentication
System using Node.js and MongoDB

More JavaScript Courses

- [Intermediate JS with jQuery, JSON and Ajax](#)
- [Intro to Angular.JS Framework](#)
- [Intro to React.JS Framework](#)
- [Vue.JS Framework](#)

Thanks!

Any questions?

You can find me at

- [@kaustubh_ghadge](https://twitter.com/kaustubh_ghadge)
- kaustubh.ghadge@gmail.com

coding-bootcamps.com



Coding Bootcamps

