

Mobile Backend Development

Lab - Restful API (1)

Wan Muzaffar Wan Hashim

1 - ExpressJS

Creating a NodeJS project

- 1) Initialize a npm project / NodeJS project - `npm init`
- 2) Install the required modules - `npm install <modules>`
- 3) Create the `index.js` file and fill in the boilerplate/starter code.
- 4) Run the API with command line : `node index.js` or `nodemon index.js`
- 5) Test the API on Postman

The required modules

- 1) `express`
- 2) `body-parser`

`mongoose` is another module that will be used in Building REST API part 2

Required modules: express

The image shows a screenshot of the Express.js website homepage. At the top, there is a navigation bar with the 'Express' logo on the left and links for 'Home', 'Getting started', 'Guide', 'API reference', 'Advanced topics', and 'Resources'. The main content area features the 'Express 4.15.3' logo and a tagline: 'Fast, unopinionated, minimalist web framework for Node.js'. Below this is a terminal-style code box containing the command '\$ npm install express --save'. To the right of the main text is a dark-themed promotional image for a keynote titled 'KEYNOTE: Express, State of the Union' by Doug Wilson, featuring the Node.js logo and 'node Interactive' branding. At the bottom, there are four columns of text describing Express's features: 'Web Applications', 'APIs', 'Performance', and 'Frameworks'.

Express

Home Getting started Guide API reference Advanced topics Resources

Express 4.15.3

Fast, unopinionated,
minimalist web
framework for Node.js

```
$ npm install express --save
```

KEYNOTE: Express, State of the Union
Doug Wilson, Express

Web Applications
Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications.

APIs
With a myriad of HTTP utility methods and middleware at your disposal, creating a robust API is quick and easy.

Performance
Express provides a thin layer of fundamental web application features, without obscuring Node.js features that you know and love.

Frameworks
Many popular frameworks are based on Express.

Required modules: body-parser

body-parser

npm v1.17.2 downloads 8M/month build passing coverage 100% tips \$2.35/week

Node.js body parsing middleware.

Parse incoming request bodies in a middleware before your handlers, available under the `req.body` property.

[Learn about the anatomy of an HTTP transaction in Node.js.](#)

This does not handle multipart bodies, due to their complex and typically large nature. For multipart bodies, you may be interested in the following modules:

- [busboy](#) and [connect-busboy](#)
- [multiparty](#) and [connect-multiparty](#)
- [formidable](#)
- [multer](#)

This module provides the following parsers:

- [JSON body parser](#)
- [Raw body parser](#)
- [Text body parser](#)
- [URL-encoded form body parser](#)

The boilerplate code : index.js file (1) - Import and Setting up middleware

```
var express    = require('express');           // call express
var app        = express();                   // define our app using express
var bodyParser = require('body-parser');

app.use(bodyParser.urlencoded({ extended: true }));
app.use(bodyParser.json());

var port = process.env.PORT || 8080;         // set our port
```

The boilerplate code : index.js file (2) - Setting route and path

```
var router = express.Router();

router.get('/', function(req, res) {
  res.json({ message: 'hooray! welcome to our api!' });
});

app.use('/api', router);

app.listen(port);
console.log('Magic happens on port ' + port);
```


The boilerplate code : index.js file (2) - Setting route and path

```
var router = express.Router();

router.get('/', function(req, res) {
  res.json({ message: 'hooray! welcome to our api!' });
});

app.use('/api', router);

app.listen(port);
console.log('Magic happens on port ' + port);
```

Test the API Using POSTMAN

The screenshot shows the Postman application interface. The top bar includes a navigation menu with 'New', 'Import', and 'Runner' buttons, and a workspace name 'My Workspace'. The main area is titled 'Untitled Request' and shows a POST request to the URL 'http://morning-gorge-50959.test/api/v2/video'. The 'Send' button is highlighted in blue. Below the URL bar, there are tabs for 'Params', 'Authorization', 'Headers (9)', 'Body', 'Pre-request Script', 'Tests', and 'Settings'. The 'Params' tab is active, showing a table for 'Query Params' with columns for 'KEY', 'VALUE', and 'DESCRIPTION'. The table contains one entry: 'Key' with 'Value' and 'Description'. The 'Response' section is currently empty, displaying a rocket launch icon and the text 'Hit Send to get a response'. A notification banner at the bottom of the interface reads 'Learn how to debug requests and perform manual testing Start X'. The left sidebar shows a 'History' list of previous requests and a date filter for 'March 28'.

Postman

New Import Runner

My Workspace Invite

Filter

History Collections APIs

Save Responses Clear all

POST https://dashboard.anak2u.com.my/api/v2/receipts/565

POST https://dashboard.anak2u.com.my/api/v2/receipts/867

POST https://dashboard.anak2u.com.my/api/v2/receipts/892

POST https://dashboard.anak2u.com.my/api/v2/receipts/880

POST https://dashboard.anak2u.com.my/api/v2/receipts/869

March 28

POST https://dashboard.anak2u.com.my/api/v2/receipts/893

POST https://dashboard.anak2u.com.my/api/v2/receipts/891

POST https://dashboard.anak2u.com.my/api/v2/receipts/890

POST https://dashboard.anak2u.com.my/api/v2/receipts/887

POST https://dashboard.anak2u.com.my/api/v2/receipts/883

POST https://dashboard.anak2u.com.my/api/v2/receipts/881

POST https://dashboard.anak2u.com.my/

h POST h POST POST h POST h POST h POST h PUT

No Environment

Comments 0

Untitled Request

POST http://morning-gorge-50959.test/api/v2/video Send Save

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies Code

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Response

Hit Send to get a response

Learn how to debug requests and perform manual testing Start X

Bootcamp Build Browse

Add the following functionalities

- 1) A route hello that will return "Hello World" as body
- 2) A route goodbye and will take "name" as parameter.

It will return "Goodbye <name>" in return

- 3) A route sum that will take num1 and num2 in body.

It will return the sum of num1 and num2 in body.

2 - Deploying in HEROKU



Automatically deploy GitHub pull requests

Check out [Review Apps](#): automatically deploy GitHub Pull Requests to disposable apps as part of a continuous delivery workflow.



Learn about building, deploying and managing your apps on Heroku.



Node.js



Ruby



Java



PHP



Python



Go



Scala



Clojure

Create a Heroku Account

- 1) Go to [Heroku.com](https://heroku.com)
- 2) Register for Heroku account.
- 3) Download the Heroku CLI.

Install Git

- 1) Go to <https://git-scm.com/downloads>
- 2) Download Git for command line.

Steps to committing a project.

- 1) Go to your project folder, type **git init**.
- 2) Type **git add .** to add all your files into a commit list.
- 3) Type **git commit -m "First commit"** to commit your change.

Important Git command

Command	Description
<code>git init</code>	Creating a git repository in the folder.
<code>git add .</code>	Add all files into commit list
<code>git commit -m "<commit message>"</code>	Commit/Save the change that you have done.
<code>git push</code>	Push the change to server.

Log into Heroku

```
$ heroku login
```

```
Enter your Heroku credentials.
```

```
Email: zeke@example.com
```

```
Password:
```

```
...
```

Create a Heroku project.

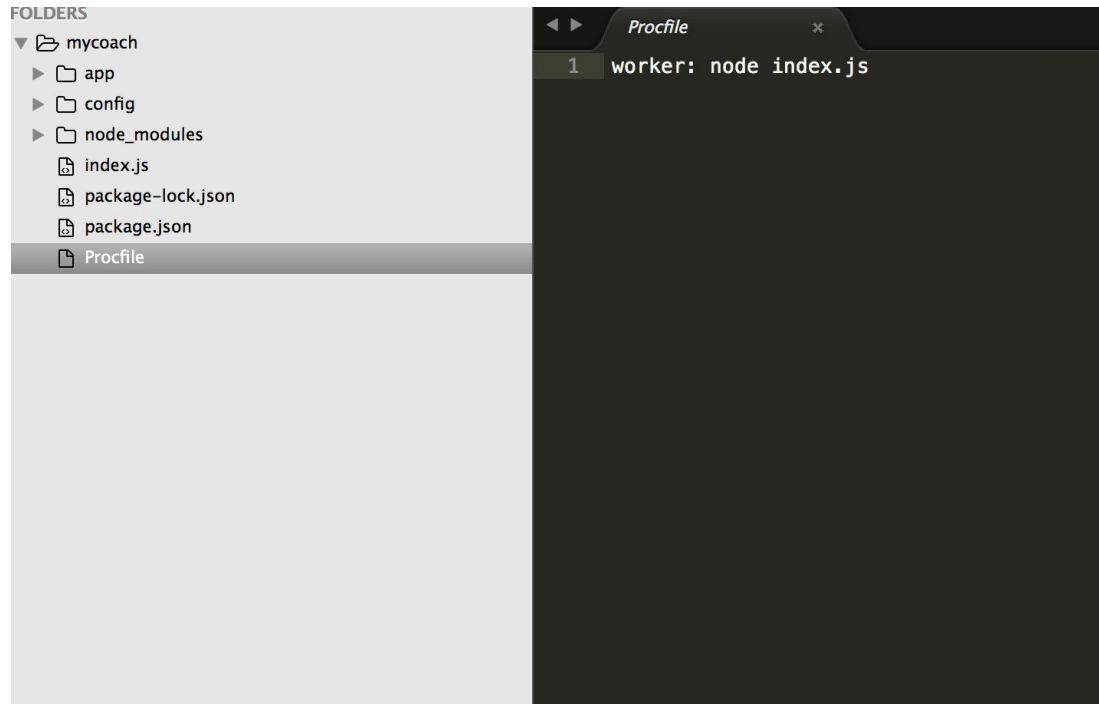
```
$ heroku create
```

```
Creating sharp-rain-871... done, stack is cedar-14
```

```
http://sharp-rain-871.herokuapp.com/ | https://git.heroku.com/sharp-rain-871.git
```

```
Git remote heroku added
```

Create a Procfile



In package.json add the following lines.

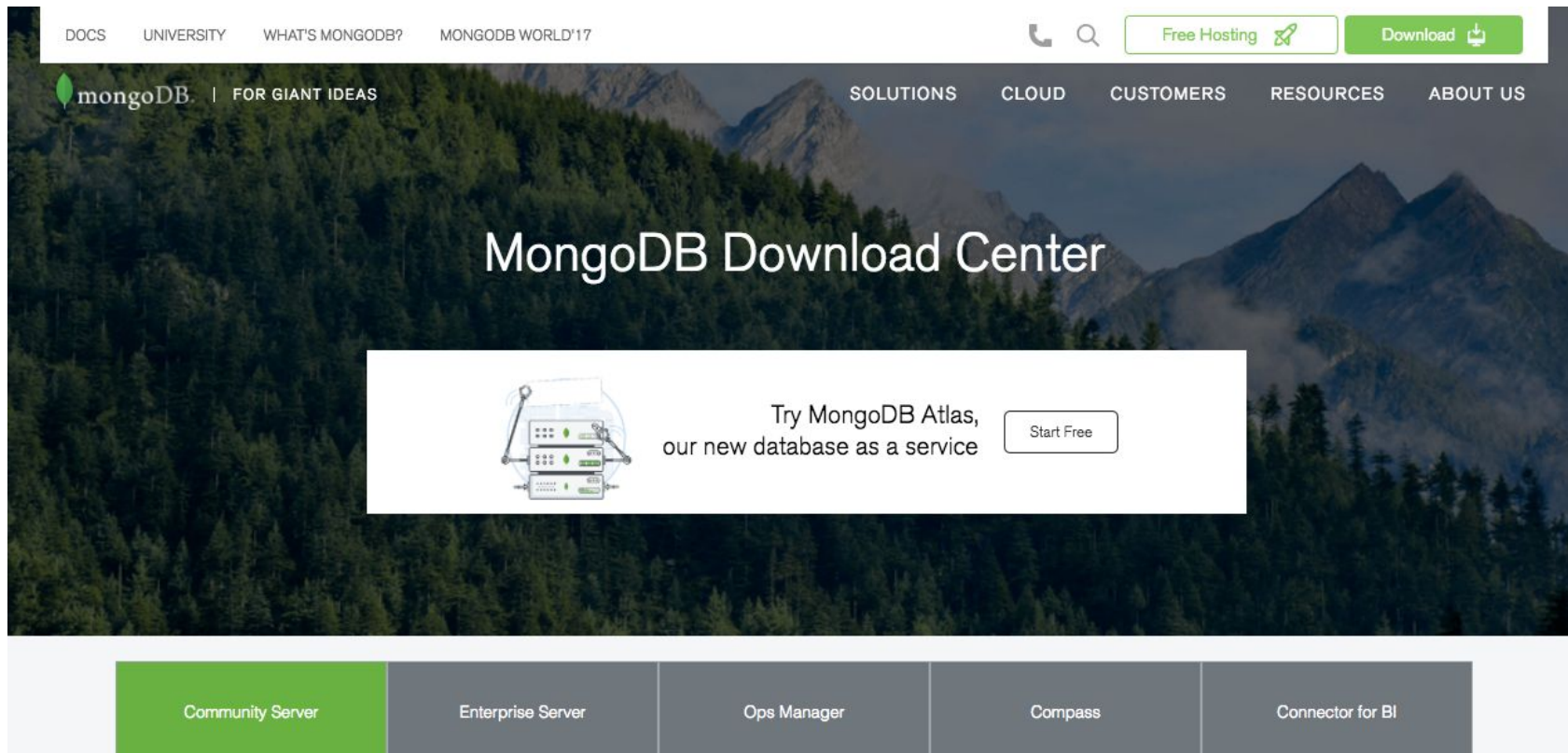
```
main : index.js ,  
"scripts": {  
  "test": "echo \"Error: no test specified\" && exit 1",  
  "start": "node index.js"  
},  
"author": ""
```

Push the project

```
$ git push heroku master
Counting objects: 343, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (224/224), done.
Writing objects: 100% (250/250), 238.01 KiB, done.
Total 250 (delta 63), reused 0 (delta 0)
remote: Compressing source files... done.
remote: Building source:
remote:
remote: ----> Node.js app detected
remote:
remote: ----> Creating runtime environment
remote:
remote:           NPM_CONFIG_LOGLEVEL=error
remote:           NPM_CONFIG_PRODUCTION=true
remote:           NODE_MODULES_CACHE=true
remote:
remote: ----> Installing binaries
remote:           engines.node (package.json): 5.9.1
remote:           engines.npm (package.json):  unspecified (use default)
remote:
remote:           Downloading and installing node 5.9.1...
remote:           Using default npm version: 2.7.4
remote:
remote:     ....
remote: ----> Build succeeded!
remote:           └─ ejs@2.4.1
remote:           └─ express@4.13.3
remote:
remote: ----> Discovering process types
remote:           Procfile declares types =>
```

3 - MongoDB

Installing MongoDB locally



The screenshot shows the MongoDB Download Center website. The background is a scenic view of a forested mountain range. At the top, there is a navigation bar with links for 'DOCS', 'UNIVERSITY', 'WHAT'S MONGODB?', and 'MONGODB WORLD'17'. On the right side of the navigation bar, there are icons for a phone, a search magnifying glass, a 'Free Hosting' button with a rocket icon, and a 'Download' button with a download icon. Below the navigation bar, the MongoDB logo is followed by the tagline 'FOR GIANT IDEAS'. To the right of the logo, there are links for 'SOLUTIONS', 'CLOUD', 'CUSTOMERS', 'RESOURCES', and 'ABOUT US'. The main heading in the center is 'MongoDB Download Center'. Below this heading is a white rectangular box containing an illustration of server racks and the text 'Try MongoDB Atlas, our new database as a service' with a 'Start Free' button. At the bottom of the page, there is a horizontal menu with five buttons: 'Community Server' (highlighted in green), 'Enterprise Server', 'Ops Manager', 'Compass', and 'Connector for BI'.

DOCS UNIVERSITY WHAT'S MONGODB? MONGODB WORLD'17

Free Hosting Download

mongoDB | FOR GIANT IDEAS SOLUTIONS CLOUD CUSTOMERS RESOURCES ABOUT US

MongoDB Download Center

Try MongoDB Atlas, our new database as a service [Start Free](#)

Community Server Enterprise Server Ops Manager Compass Connector for BI

CLUSTERS > CREATE NEW CLUSTER

Create New Cluster

Global Cluster Configuration >

Cloud Provider & Region

AWS, N. Virginia (us-east-1) ▾

Select **Multi-Region, Workload Isolation, and Replication Options** (M10+ clusters)Increase region availability, configure tagged analytics nodes, and optimize for local service areas. [Read more](#) NOCreate a **free tier cluster** by selecting a region with **FREE TIER AVAILABLE** and choosing the **M0** cluster tier below.

★ Recommended region ⓘ

NORTH AMERICA

EUROPE

ASIA

\$0.54/hour

Pay-as-you-go You will be billed hourly and can terminate your cluster anytime. Excludes variable **data transfer, backup, and taxes.**

Cancel

Create Cluster



Privacy - Terms





DATA STORAGE

Clusters

Triggers

Data Lake

SECURITY

Database Access

Network Access

Advanced

We are deploying your changes: 0 of 3 servers complete (current action: provisioning 3 servers)

ME-TECH SOLUTION SDN BHD (ORGANIZATION) > I4.0

Clusters

Create a New Cluster

Find a cluster...

SANDBOX

Cluster0

Version 4.2.8

CONNECT

METRICS

COLLECTIONS



CLUSTER TIER

M0 Sandbox (General)

REGION

AWS / N. Virginia (us-east-1)

TYPE

Replica Set - 3 nodes

LINKED REALM APP

None Linked

Your cluster is being created..
New clusters take between 1-3 minutes to provision.



Insert collections

Insert the three or more collections following these criterias.

- a) One of the places will have 'France' as countries.
- b) One of the places will have new items which are comments, which have **user**, **message** and **rating** as items.
- c) Give different number of likes, eg: some will have more likes and some will have less like.

MongoDB Query Exercise

- 1) Retrieve all the places.
- 2) Retrieve all the places in France.
- 3) Get the query to get the top 3 highest rating places in your database.
- 4) Get the query of top 3 highest rating places in Malaysia.
- 5) Get the query to get a place in France that has rating/like more than 80.
- 6) Return all the interesting places in France and Italy.

Additional Exercise

Refer to MongoDB Extra Exercise document.