Setup

NEXT: PROVISIONING A MONGODB DATABASE

You need to install Node and npm before anything reasonable can be achieved. Installing Node automatically installs npm as well. Node can be installed from the installation instructions (basically download and install) on the website.

To test if the installation was successful, run the following commands:

```bash
# Node version
node -v

# npm version
npm -v
```

If the installed version gets printed, then you are good to go.

Start a new project in your working directory with the following commands. Just like we used the Vue CLI to generate a new project, we'll use the Express CLI tools to do the same.

```bash
# Install express generator
npm install express-generator -g

# generate a new project
express --view=ejs api
```

The generator creates a new express app called `api`. The `--view` flag tells the generator to use ejs template engine for the view files.

You can start the default app in the template by running:

```bash
# Install dependencies
npm install
# Start app
npm start
```

You also need to install Mongoose (we will talk about the responsibilties in next lessons):

```bash
npm install --save mongoose
```

The entry app for this project is named `app.js` and can be found at the root of the project. Let's have a look at this file:

```js
// Imports
var express    = require('express');
var path       = require('path');
var logger     = require('morgan');
var bodyParser = require('body-parser');
const mongoose = require('mongoose');

// Require routes
// They are yet to be created
var index = require('./routes/index');
const api = require('./routes/api/index');

// Setup an express app
var app = express();
// ************************

// Database connection here

// ********************

// ************************

// CORS config will be here

// ********************

// view engine setup
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'ejs');

// Configure middlewares
app.use(logger('dev'));
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: false }));

app.use('/', index);
app.use('/users', users);
app.use('/api/v1', api)

module.exports = app;
```

The file creates and exports an express app. The app is configured with the middlewares we stated in the previous section.

Express is always comprised of routes that will handle requests and responses to and fro the browser. We have made not routes so far, but we are importing files that will represent these routes.