**JavaScript Basic Web: Phonebook**

1. **Problem**

You have been tasked to create a simple **Phonebook** application. The application should hold **contacts**, which are the main app **entity**.

The functionality of the application should support:

* **Listing contacts**



* **Add Contact**

****

****

1. **Overview**

**Requirements**

* **Express** framework
* **Handlebars** view engine

**Data Model**

The **Contact** holds **2 properties**:

* **name – non-empty text**
* **number – non-empty text**

**Project Skeletons**

You will be given the application **skeletons**, which hold about **90%** of the logic. You’ll be given some **files**. The files will have **partially implemented logic**, so you'll need to write some code for the application to **function properly**.

The application's views will be given to you fully implemented. You only need to include them in your business logic.

Everything that has been given to you inside the skeleton is **correctly implemented** and if you write your code **correctly**, the application should work just fine. You are free to change anything in the Skeleton on your account.

1. **Preparation**

Run **"npm install"** in the terminal in the folder of the project to install all of the dependencies you will need. (express and handlebars)

1. **Database**

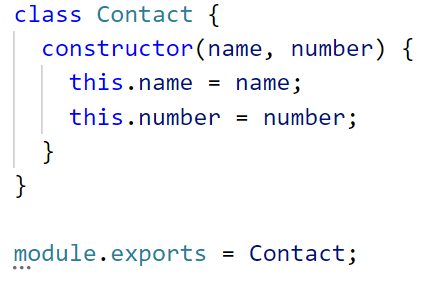
It's time to create our database. Open the **"phonebook.js"** file and add the following functionality:



* We **create a phonebook array**, which will store all the contacts (**memory storage**). The array **will be empty** each time we **restart the server**
* We add the function that will **return the whole phonebook**
* We add the function that will **add a new contact**
* Finally, we **export those functions**, so we can **use them outside this file**

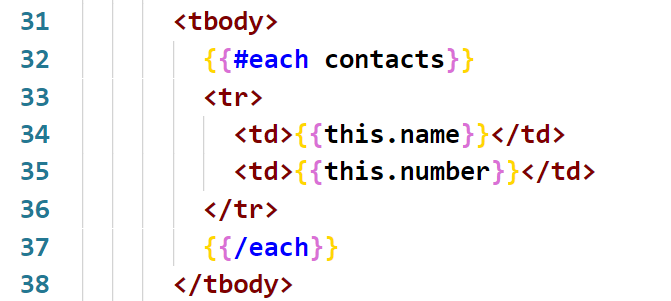
1. **Model**

We should create our **contact model** which has a **name** and **number** as properties. Inside **Contact.js** creates a **class** that has a constructor and accepts and sets both properties. After that export it:



1. **View**

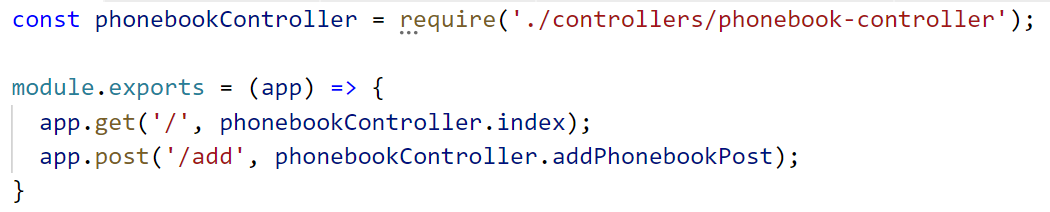
Go to the **index.hbs** file and find the following code:



What this does is, it should **receive variables called contacts**, it **iterates through** them, and for each, it adds a row in the table with this **person's name and number**.

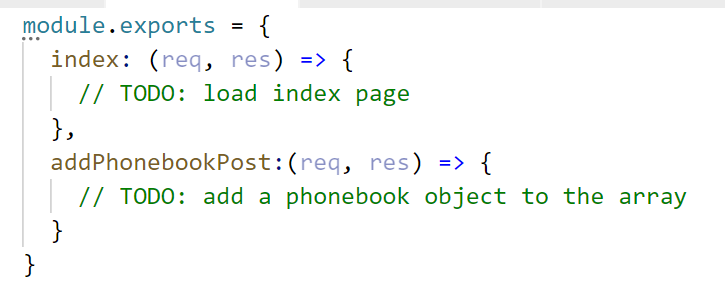
1. **Routing**

The routes in our application are defined inside **routing.js**. There are a total of **two** routes inside **'/'** which is called when we load the application (HTTP GET) and **'/add'** which is called when we **add a new contact** inside the database array (HTTP POST). Every route leads to a function from the controller:



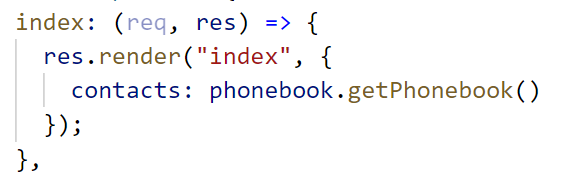
1. **Phonebook Controller**

To **list** and **add** contacts we need a controller so go to **controllers/phonebook-controller.js**:



**Listing Contacts**

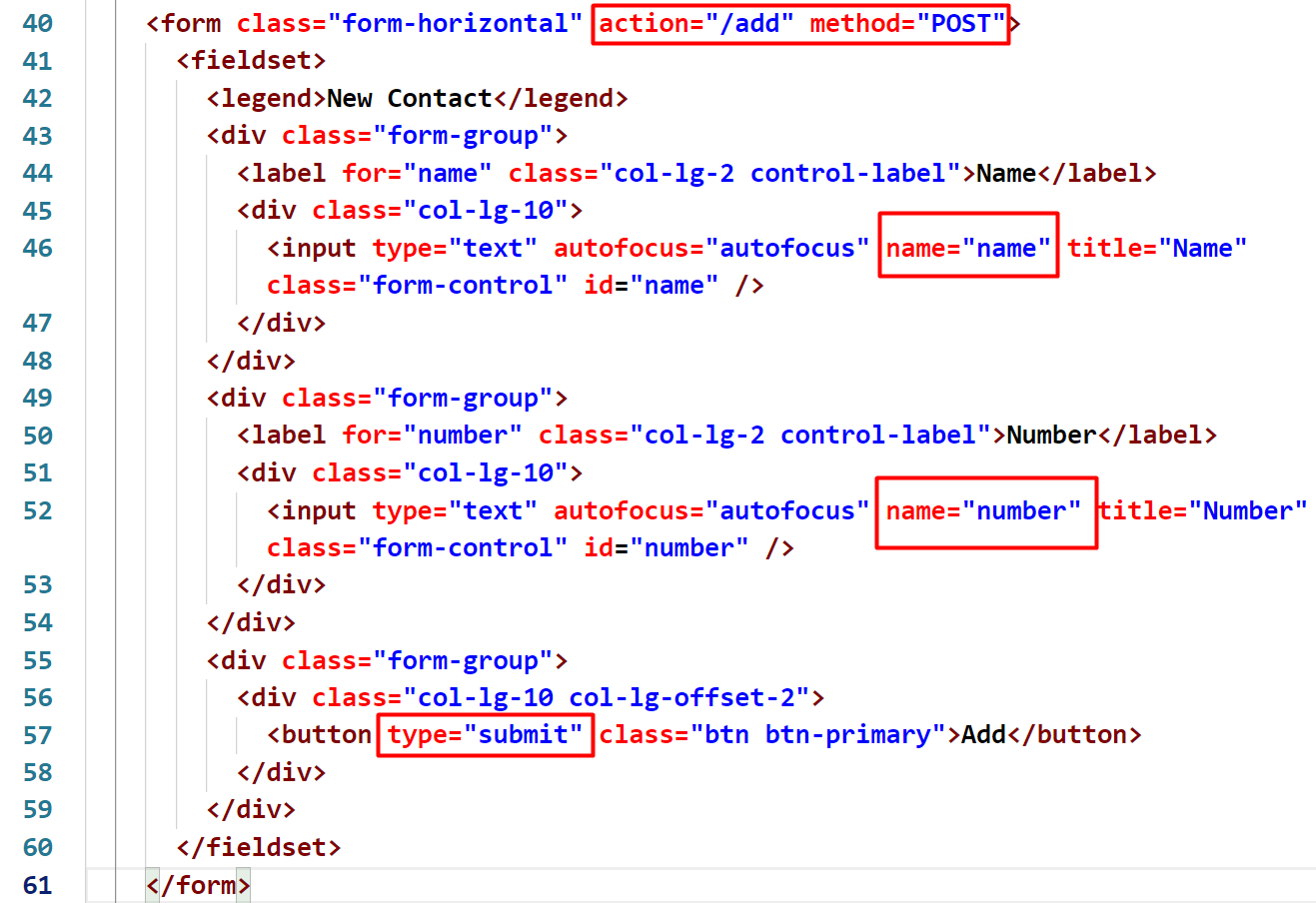
Then when we render the main page, we need to **pass the stored contacts to the view**, so we can see them. To do that, add the following code in the **index** method:



* We pass the contacts as an object that has a value for **"contacts"** in the phonebook we stored.
* We do that by using our **getPhonebook()** method we wrote earlier

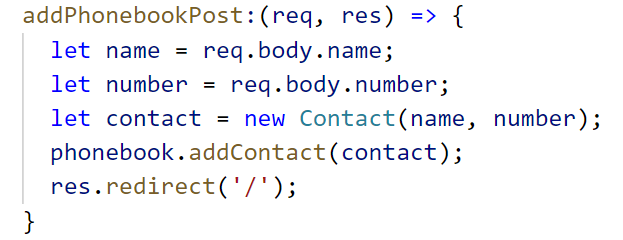
**Adding Contact**

First, let us see what we need to do. Go to the **index.hbs** again and find the following lines of code:



* Here we see, that when we **press the submit button**, we make a **POST** request on **route "/add"**.
* The information that we submitted in this form will come as an **object with properties name and number** (since those are the **names of the input fields**)
* This is only possible because we imported **"body-parser"**

So let us now go to **phonebook-controller.js** and add the final functionality, which goes in the **app.post()** method:



* Don't forget to require the **Contact class** from **models/contact.js**



* By typing **req.body** we get that object that we mentioned earlier (you can log that to the console if you want to see it)
* Then we just **use the function** we wrote to **add that contact** and **redirect to the index** page again

1. **Testing the Application**

With that, we finished our **JavaScript Phonebook**. Type **"node index.js"** in the terminal to start the server. Then go to **localhost:3000**. Feel free to **build on your project even further**. ☺