# Node.js Coding Challenge

You are being asked to create a simple contact entry system. You should create a REST API that will enable a client to perform CRUD operations on the contact collection. The exercise is expected to take 2-4 hours to complete, but there are no constraints on how much time you can devote to it. Please use code comments or the README to explain any unconventional decisions or shortcuts that are apparent in your implementation.

##### **Requirements**

1. Create a new REST API using Node.js technologies with the following endpoints:

|  |  |  |
| --- | --- | --- |
| HTTP Method | Route | Description |
| GET | /contacts | List all contacts |
| POST | /contacts | Create a new contact |
| PUT | /contacts/{id} | Update a contact |
| GET | /contacts/{id} | Get a specific contact |
| DELETE | /contacts/{id} | Delete a contact |
| GET | /contacts/call-list | Get a call list (see detailed requirements in item #4 below) |

1. The contact entry request when creating or updating a contact will be JSON and have the following format:

|  |  |
| --- | --- |
| Format | Example |
| "name": {  "first": string,  "middle": string,  "last": string  },  "address": { "street": string,  "city": string,  "state": string,  "zip": string  },  "phone": [  {  "number": string,  "type": string ["home" | "work" | "mobile"]  }  ],  "email": string } | {  "name": {  "first": "Harold",  "middle": "Francis",  "last": "Gilkey"  },  "address": {  "street": "8360 High Autumn Row",  "city": "Cannon",  "state": "Delaware",  "zip": "19797"  },  "phone": [  {  "number": "302-611-9148",  "type": "home"  },  {  "number": "302-532-9427",  "type": "mobile"  }  ],  "email": "harold.gilkey@yahoo.com" } |

1. Contact entries returned to the client by the API will be JSON and have the following format:

|  |  |
| --- | --- |
| Format | Example |
| {  "id": number,  "name": {  "first": string,  "middle": string,  "last": string   },  "address": {  "street": string,  "city": string,  "state": string,  "zip": string  },  "phone": [  {  "number": string,  "type": string ["home" | "work" | "mobile"]  }  ],  "email": string } | {  "id": 101,  "name": {  "first": "Harold",  "middle": "Francis",  "last": "Gilkey"  },  "address": {  "street": "8360 High Autumn Row",  "city": "Cannon",  "state": "Delaware",  "zip": "19797"  },  "phone": [  {  "number": "302-611-9148",  "type": "home"  },  {  "number": "302-532-9427",  "type": "mobile"  }  ],  "email": "harold.gilkey@yahoo.com" } |

1. The call list is generated from all contacts that include a home phone. It is sorted first by the contact’s last name, then by first name, and returned as an array of objects that each have the following JSON format:

|  |  |
| --- | --- |
| Format | Example |
| {  "name": {  "first": string,  "middle": string,  "last": string   },  "phone": string  } | {  "name": {  "first": "Harold",  "middle": "Francis",  "last": "Gilkey"  },  "phone": "302-611-9148" } |

1. Provide a storage mechanism for storing the contact entries. If you use a database you do not need to include the database, but you should include instructions to set up and create the database. In order to simplify the overall solution and minimize runtime dependencies, you are strongly encouraged to use an embedded database such as [NeDb](https://github.com/louischatriot/nedb), [LokiJS](http://techfort.github.io/LokiJS/), or something similar. This makes it much quicker and easier for us to run your application locally.
2. Write unit tests to verify functionality where you deem it appropriate.
3. Upload all files and instructions for building the project (if necessary) to [GitHub](https://www.github.org/) and provide a link to us for review.