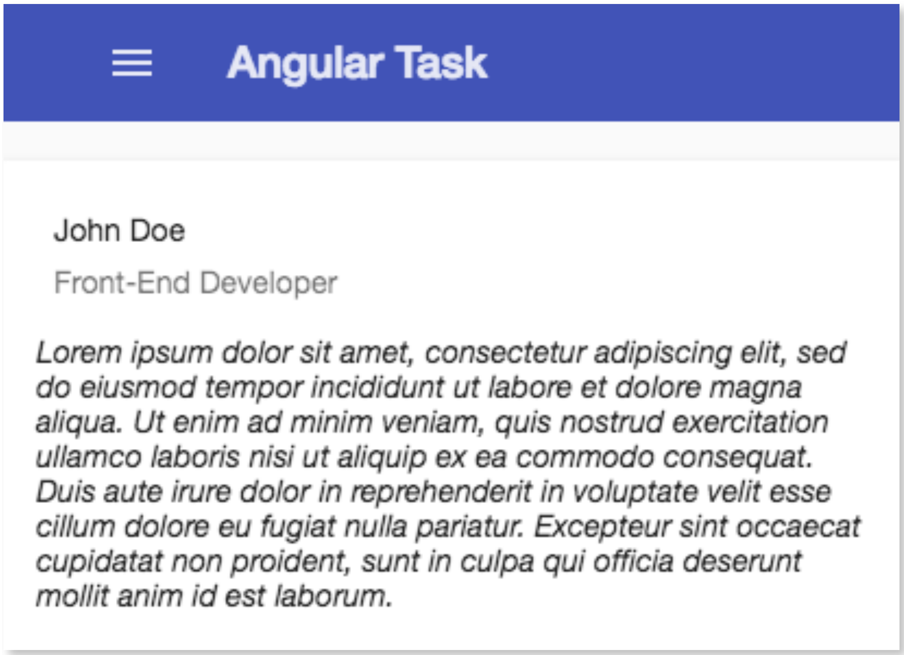
Front-End - Angular Task

Create an Angular app that manages a simple contact list, similar to the one found on every phone. However, instead of phone numbers, each contact can have a number of addresses.

1. The app should have two main pages. The home page, which contains static text, and the contact page, which contains a list of contacts.
2. Contacts are retrieved from a mock api (**see details below**) and displayed on the contacts page. A new contact can be added from the same page. Note: Modification (editing) of contact details is not required for this task.
3. Clicking/Tapping on a contact will display his/her information and also list all the addresses (if any) related to that contact. Existing addresses can be modified or removed, and new addresses can be added.

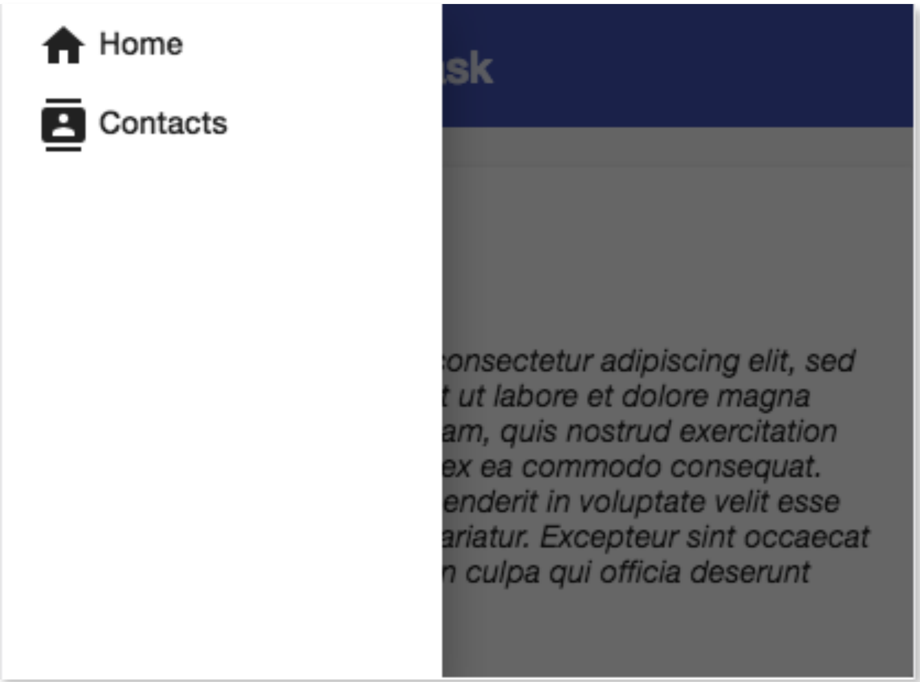


John Doe

Front-End Developer

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Fig 1. Home page example



onsectetur adipiscing elit, sed
t ut labore et dolore magna
am, quis nostrud exercitation
ex ea commodo consequat.
enderit in voluptate velit esse
ariatur. Excepteur sint occaecat
n culpa qui officia deserunt

Fig 2. Example of menu displaying the two navigation links

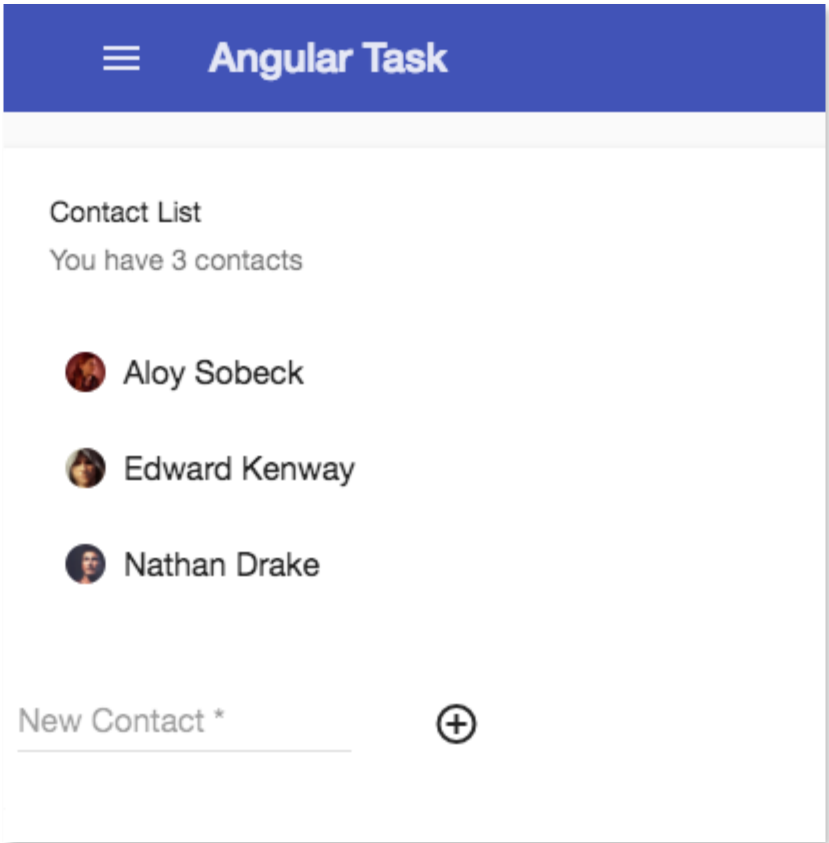
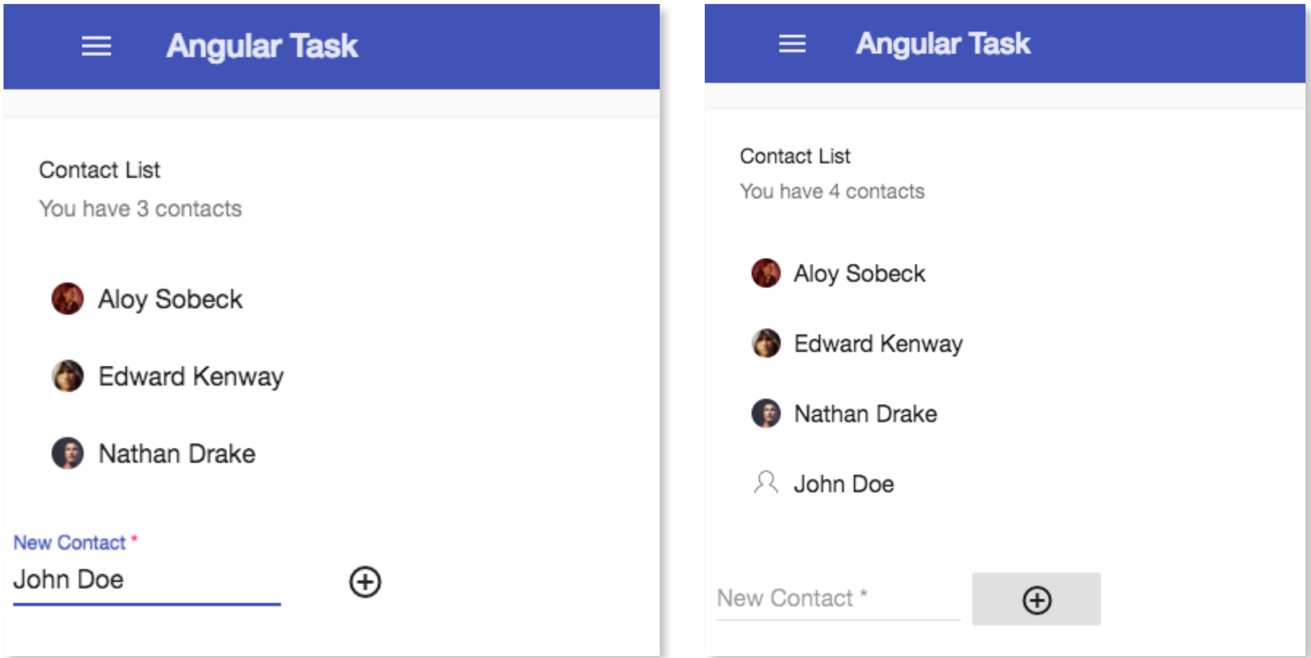


Fig 3. Contacts page example



Angular Task

Contact List

You have 3 contacts

Aloy Sobeck

Edward Kenway

Nathan Drake

New Contact *

John Doe

Angular Task

Contact List

You have 4 contacts

Aloy Sobeck

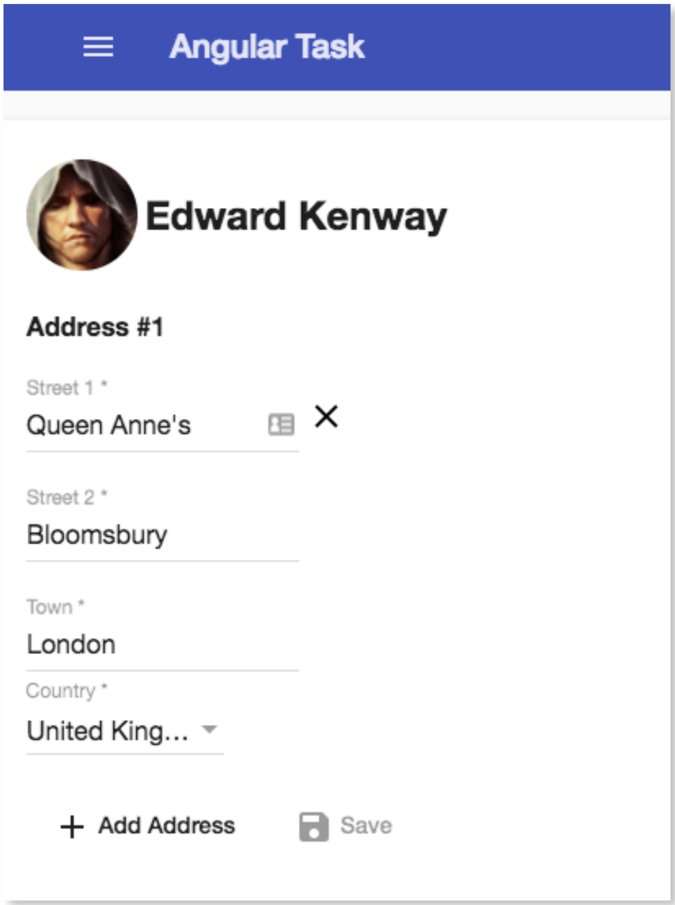
Edward Kenway

Nathan Drake

John Doe

New Contact *

Fig 4 & 5. Adding new contact example



Angular Task

Edward Kenway

Address #1

Street 1 *

Queen Anne's

Street 2 *

Bloomsbury

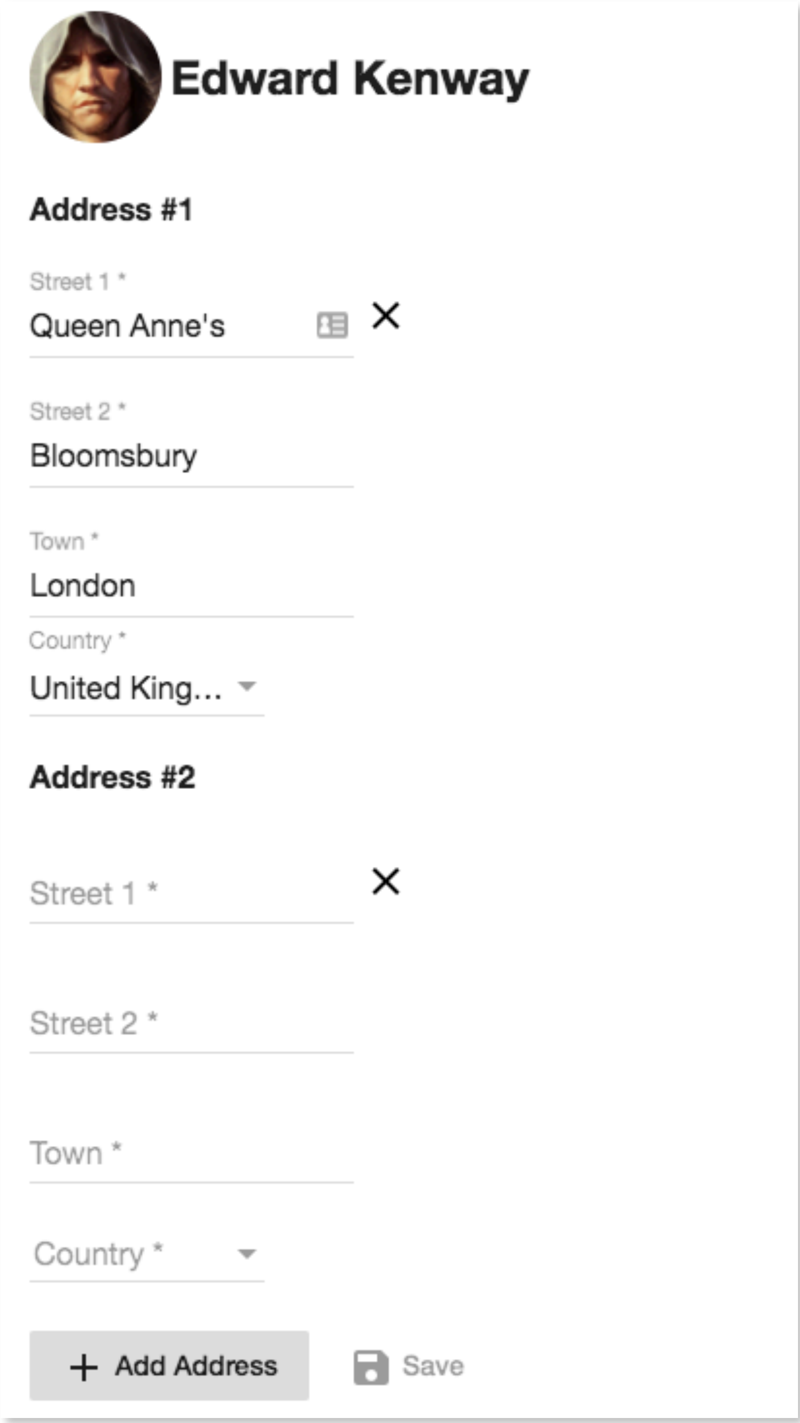
Town *

London

Country *

United King…

Fig 6. Contact details, showing existing address example. Changes can be made and saved.



Edward Kenway

Address #1

Street 1 *: Queen Anne's

Street 2 *: Bloomsbury

Town *: London

Country *: United King…

Address #2

Street 1 *

Street 2 *

Town *

Country *

Fig 7. Adding a second address. Note: For this particular scenario, when saving addresses, all information is saved, meaning if any changes occurred in address 1, those are updated, whereas address 2 is added.

CRUD operations via REST api

For this task we are using json-server as a mock REST api. To install, simply run this command:

```
> npm install -g json-server
```

For more information please visit their github page. https://github.com/typicode/json-server

The *contacts.json* file provided with this document has the initial data required for contacts, addresses, and also a list of countries to be used for addresses. DO NOT modify this file directly. Only open it as a reference for your models.

Once *json-server* is installed globally onto your machine, run this command in the same directory where *contacts.json* is located:

```
> json-server contacts.json
```

This will initialize your mock api on http://localhost:3000. The list of api resources for contacts, addresses and countries will be displayed for reference.

Please note: all resource routes are in plural. Example:

GET	/contacts
GET	/contacts/1
POST	/contacts
PUT	/contacts/1
DELETE	/contacts/1

Contact address relationship is one-to-many. Meaning one contact can have many addresses and to create an address, the following restful standard approach is followed:

POST /contacts/1/addresses

Additional requirements

- For this task, we are NOT looking for pretty UI since this is a throwaway project. However we do encourage the use of UI libraries such as google material or something similar to help make the app more usable.
- Implementation of unit tests is not required but would be considered an asset.
- While this task is quite a simple feature, keep scalability and reusability in mind. Application structure is very important.
- Latest concepts and technologies are highly encouraged.

If possible, upload your project to a public GITHub account and provide us with a URL.