Requirements:

For this task, you are expected to implement a single page website using Angular and JSON Placeholder API.

This website contains the following entities:

- Users
- Albums
- Photos

Each user can have multiple albums, each containing multiple photos. The website should allow adding new user, listing of current users and viewing their respective albums. The following views need to be implemented (you will find the designs in the same folder as this document):

1- Header:

The header exists in all pages and is fixed on the top of the viewport and is always visible whenever you scroll in the pages. All links in the header are dummy links and shouldn't route anywhere, the only exception is the logo which should redirect to the landing page.

2- Landing Page (url: /):

In this page, you should display a list of all users in a table where it should display the following information about the user: name, email, address. Adding a new user can be done in this page by clicking on a button above the table to open a modal (popup) that contains a form with the following data:

- Name
- E-mail
- Phone
- Address
    - Street
    - Suite
    - City

All form fields are required, and the E-mail field value should be a valid E-mail format (ex: example@domain.com). Since it is a fake API you will only receive the response containing the user data, but it will not be added to a database. So, your implementation should add the newly created user (from the response) to the table (This of course won't persist the next time you load the page).

3- User Page (url: /user/:userId):

This page displays the user's info:

- Name
- E-mail
- Phone
- Address
    - Street
    - Suite
    - City

Under the user info, user's albums can be viewed. By clicking on one album, all the photos in the album should be viewed in the same page and the url should be updated with the album id (/user/:userId/album/:albumId).

Technology:

- Angular 2+
- The API URL: http://jsonplaceholder.typicode.com/
- Bootstrap/Flexbox layout

Additional notes

- Structure your application to make it scalable.
- Forms should be implemented with Angular Reactive forms
- Functions should be documented detailing what their purpose is and the functionality they carry out.

- Upload your project to a github repository so we can take a look at it any review your code.