Implementing Contact App Backend

Firstly the contact app needs a database that will maintain contact details in a profile, for that in MongoDB you need a collection named ‘profile’, the structure of the profile should be

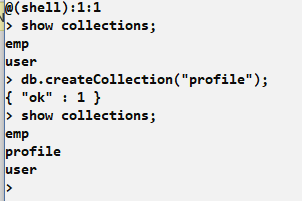
{\_id: value, name: value, password: value, dob: value, phone: value, contacts: [{…}, {…}]}

The contacts is a embedded document of a particular profile whose structure will be

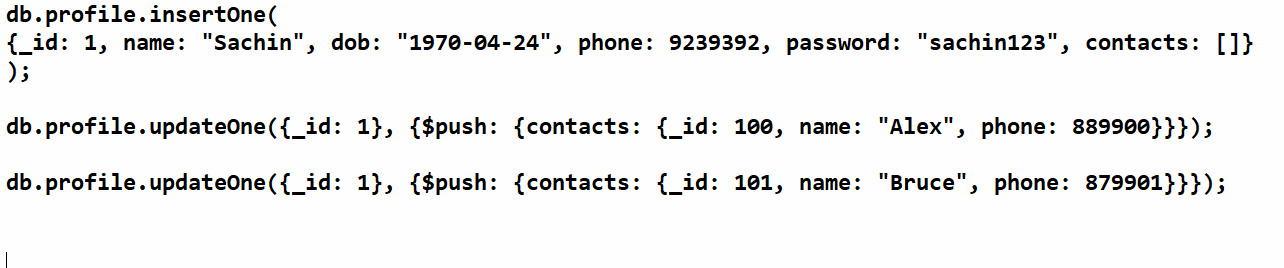
{\_id: value, name: value, phone: value}

profile.\_id is used to access the profile & profile.contacts.\_id is used to access the particular contact of the profile.

Create a collection name ‘profile’



How to store profile in mongodb



You must see the contacts of \_id : 1

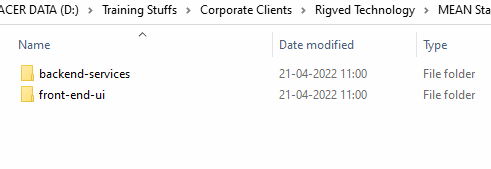


Write an express js program with node.js to store profile that will have profile properties along with the contacts

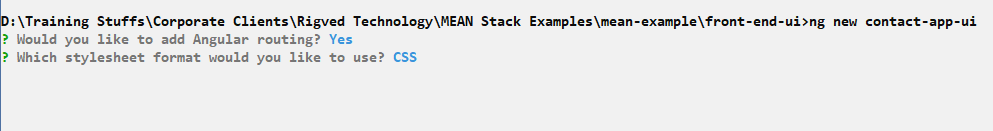
Now we need to write an express.js program that can maintain profile & contacts, as per the case study requirement we need to perform list of operations

1. Registration: Profile registration
2. Login: Profile login that enters id & password
3. Add Contacts: Logged in profile will store contact one by one
4. Edit Profile: Logged in profile can update the profile details
5. Edit Contact: Logged in profile can update the contact details
6. List Contacts: Logged in profile must see all the contacts
7. Search Contact: Logged in profile must see the contacts by name
8. Delete Contact: Logged in profile must able to delete a particular contact
9. Delete Profile: Logged in profile must be deleted
10. Logout

Create 2 folders as below;

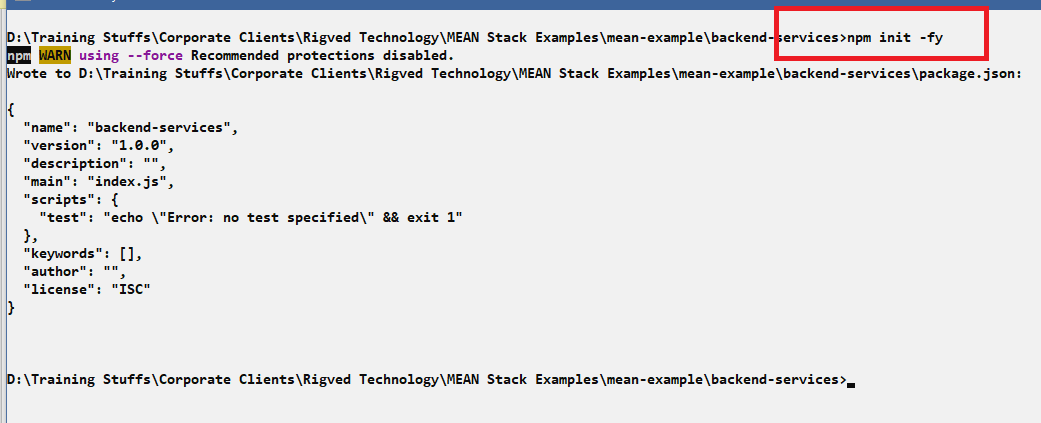


Keep angular project ready for the time begin in the front-end-ui



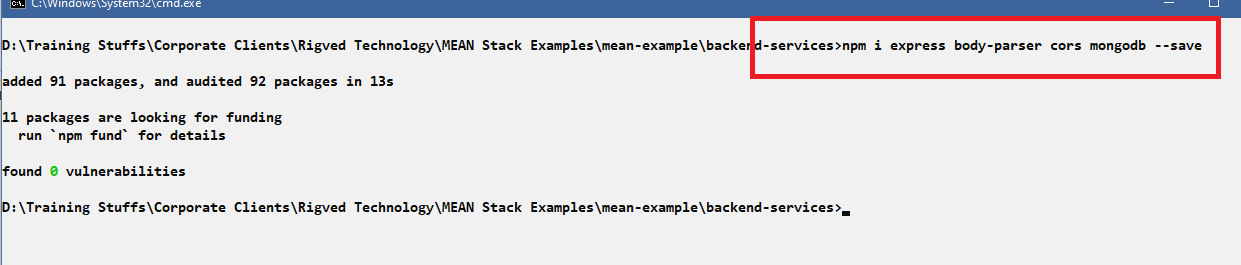
Navigate to the backend-services and create webservices through express.js and mongodb

Create package.json via npm init -fy

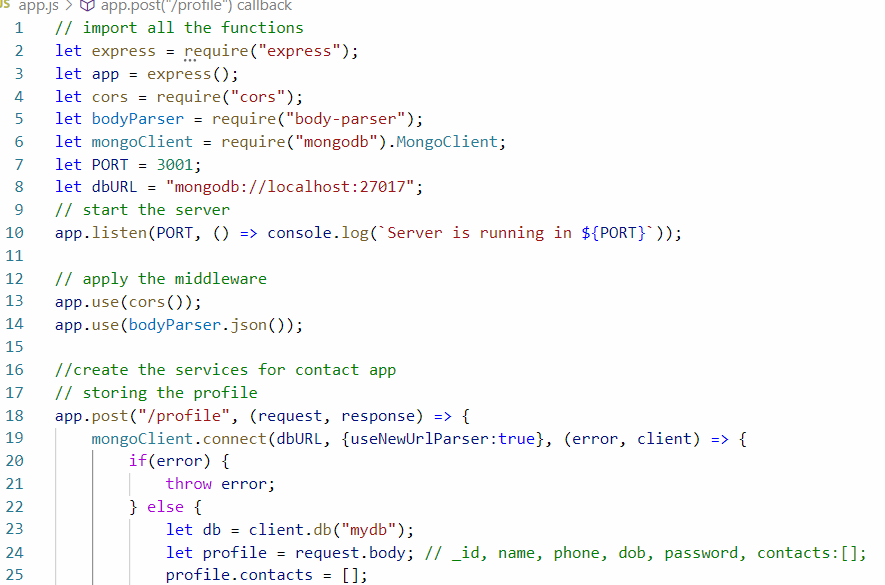


Download the necessary libraries

* express
* body-parser
* cors
* mongodb



app.js



Here the profile.contacts = [] adds a new property inside the profile object so that in the mongodb it will have contacts property.

Adding contacts for a profile

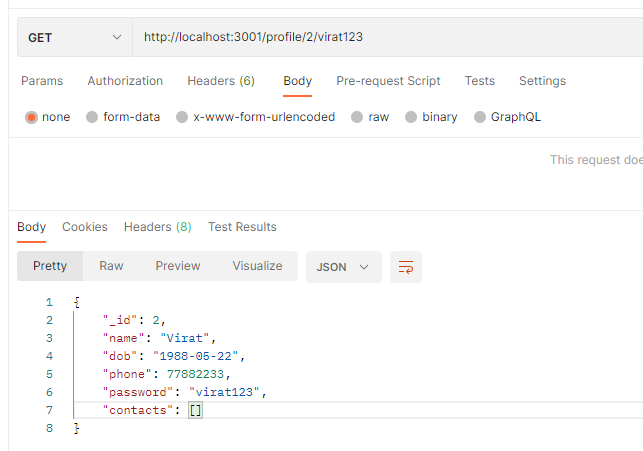
app.js



Authentication the profile with id & password



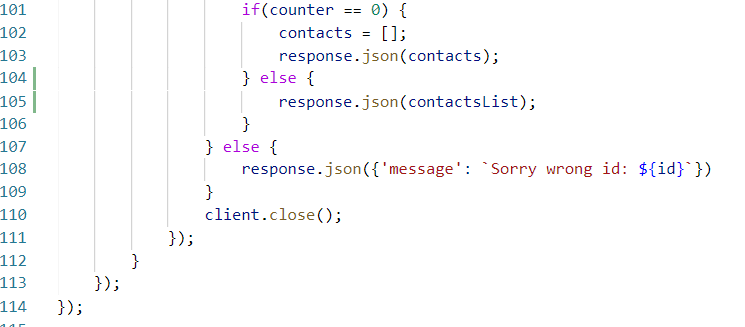
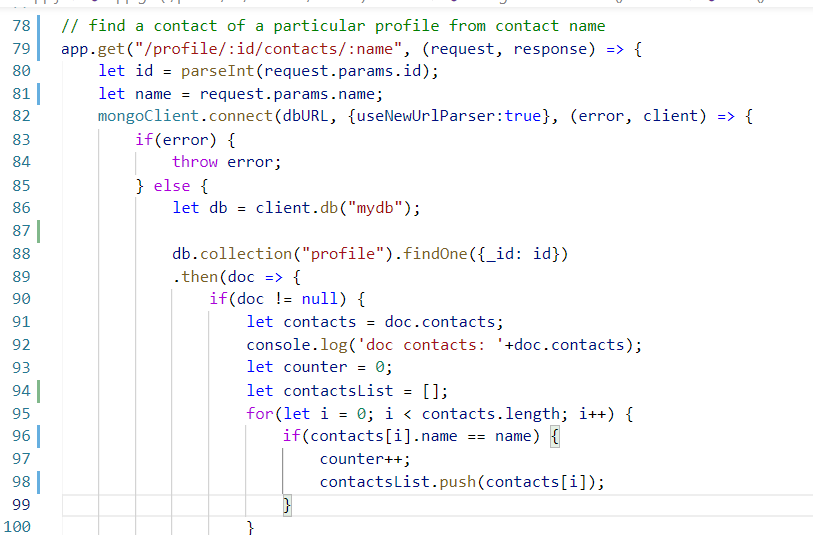
Output:



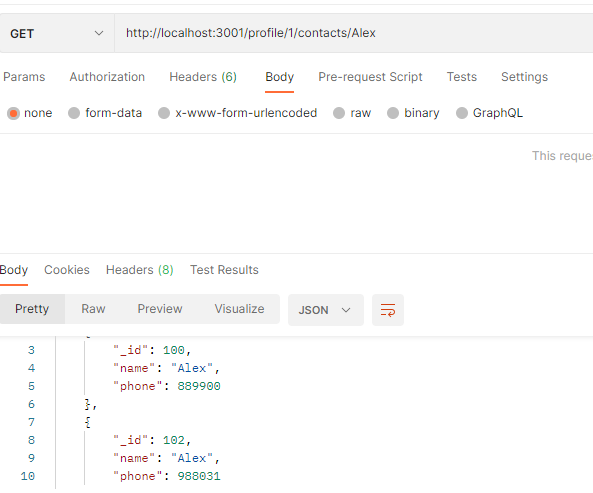
Similarly other services you can implement in express.js like

* find a contact of a particular profile id
* delete contact of a particular profile id
* update contact of a particular profile id
* search contact by name of particular profile id
* update profile of a particular profile id
* delete profile using profile id

Find a contact of particular profile id from name



Output:



Complete:

* Deleting the particular contact based on profile id
* Show all contacts based on the profile id
* Update the contact phone number of a particular profile id
* Update the profile phone number or password or date of birth of a particular profile id

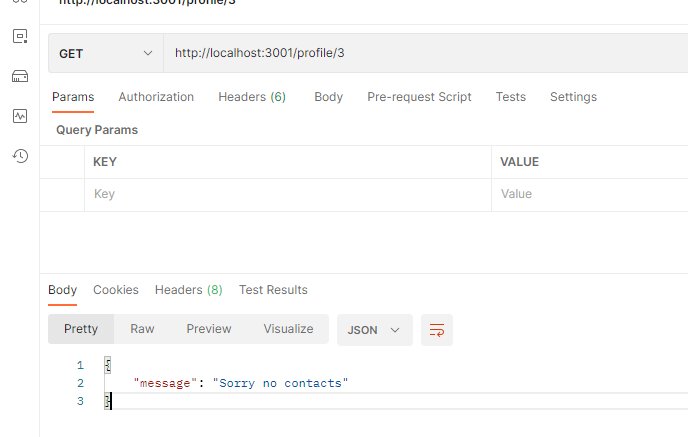
Deleting the particular contact based on profile id



Show all contacts based on the profile id



Output:



Other services you need to implement by your own

You can use angular to create UI that can interact with these services

Following are the URL’s & HTTP methods the angular must use through HttpClient

baseUrl = <http://localhost:3001/profile>

Post = / : storing profile

PUT = /:id/addContact = adding contacts

GET = /:id/:password = Login

GET = /:id/contacts/:name = Finds all contacts having the name in a profile

DELETE = /:id/contacts/:contactId = Deletes the connect based on contact id

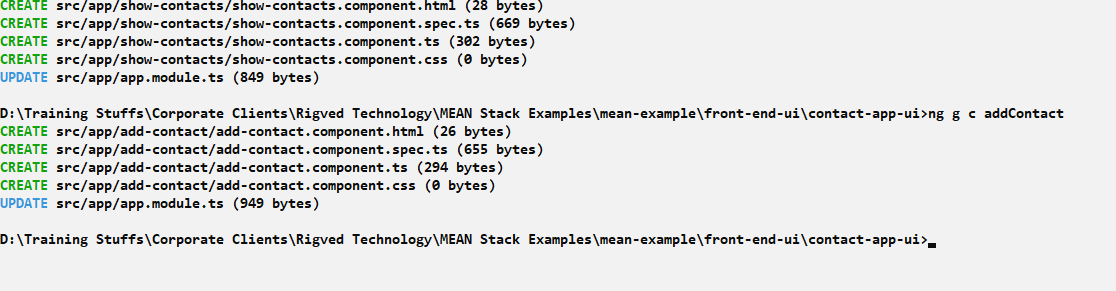
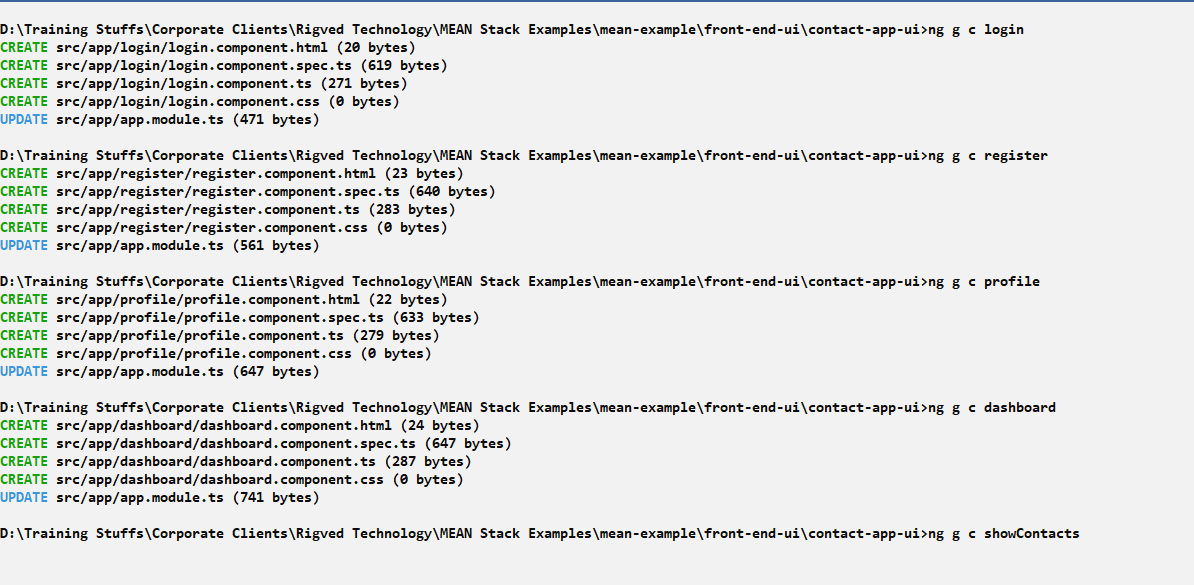
GET = /:id/c/c = Show all contacts based on the profile id

GET = /:id = Shows profile

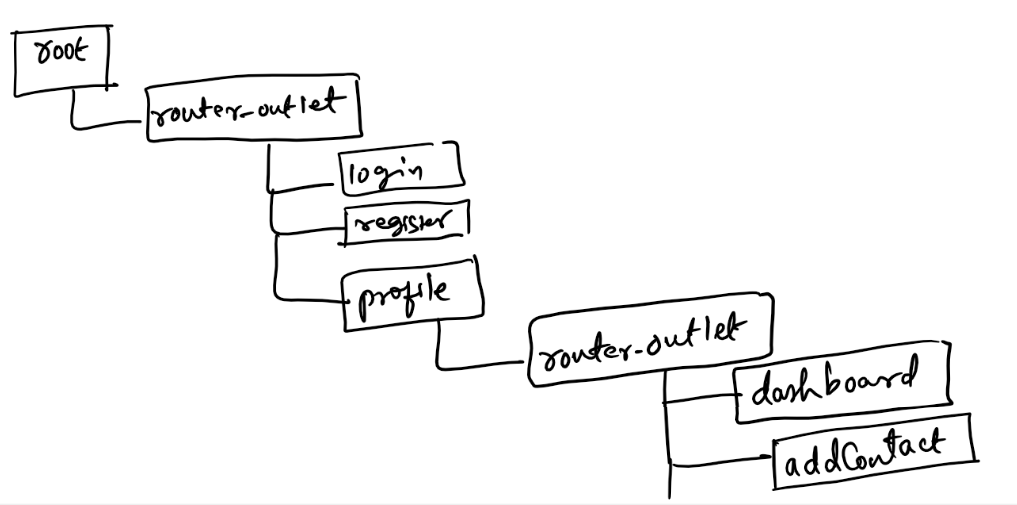
From Angular you need to create components, services and import modules in the app.module

List of components

* LoginComponent
* RegisterComponent
* ProfileComponent
* DashboardComponent
* ShowContactsComponent
* AddContactComponent



Implementation of components structure



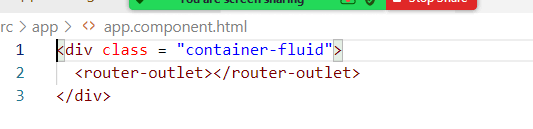
The above structure may change a bit while developing.

Modify the app-routing.module.ts



Add <router-outlet> to app.component.html

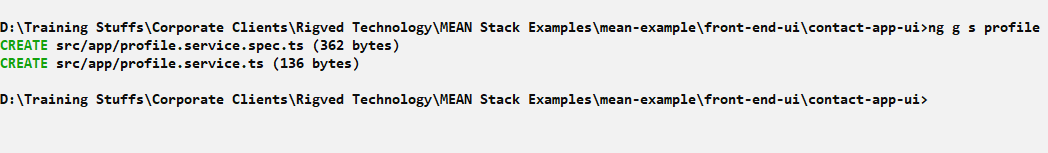
app.component.html



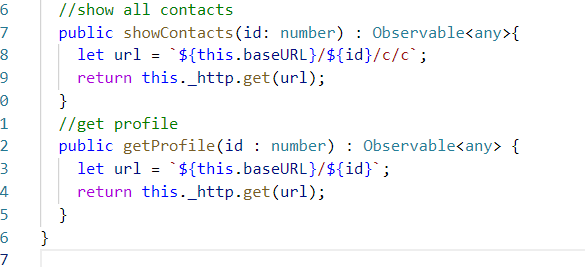
app.module.ts



Generate service that will use HttpClient



profile.service.ts



We can implement components ts & html files

Note: Refer the project files for other components