(Mobile) Android/iOS Code test: Ramdom Co.

Maximum amount of time allowed: 48 hours

Task

You work for a random company (Randomco). As a good random company they want to show random information about random users. Your task for this code test is design an Android/iOS application with these requirements:

- Download a list of random users from http://randomuser.me/ API.
- Don't show duplicated users. If Random User API returns the same user more than once you have to store just one user inside your system.
- Show a list of random users with this information sorted by name:
  - User name and surname.
  - User email.
  - User picture.
  - User phone.

- Add one button to retrieve more users and add to your current users list.
- Add one button to each cell to delete users. If you press that button your user will not be shown anymore in your user list.
- Add one button to each cell to mark users as favorite.

- If you press the user picture you have to show another view with the user detailed information:
  - User gender.
  - User name and surname.
  - User location: street, city and state.
  - Registered date.
  - User email.
  - User picture.

- **Test your code, think in the most important parts of your application and write some unit tests.**

* *Try to resolve this code test like a real life project. Think in the architecture and design of your model and try to implement it as modular as possible.*

API Description

You can obtain a list of random users in this url: http://api.randomuser.me/?results=40 The api response contains information in JSON with this format:

```
{
  "results": [
    {
      "gender": "female",
      "name": {
        "title": "Mademoiselle",
        "first": "Miranda",
        "last": "Rolland"
      },
      "location": {
        "street": {
          "number": 9536,
          "name": "Montée Saint-Barthélémy"
        },
        "city": "Wiler bei Utzenstorf",
        "state": "Schaffhausen",
        "country": "Switzerland",
        "postcode": 4163,
        "coordinates": {
          "latitude": "-48.3560",
          "longitude": "73.4869"
        },
        "timezone": {
          "offset": "-12:00",
          "description": "Eniwetok, Kwajalein"
        }
      },
      "email": "miranda.rolland@example.com",
      "login": {
        "uuid": "be70f3f9-3825-4c8f-a661-159c045ec178",
        "username": "beautifulbutterfly396",
        "password": "general",
        "salt": "FjwYH96z",
        "md5": "5617535e3c38167bdc32728c017ba2df",
        "sha1": "fd2e44c79fd79fe9484b0b3b19eb0c34c1d8156e",
        "sha256": "b9a055bf30e3c686a10e313a4054e57301207d4b24b11bdc67601fea58
75d209"
      },
      "dob": {
        "date": "1977-03-28T02:13:10.065Z",
        "age": 44
      },
      "registered": {
        "date": "2016-04-01T23:07:07.893Z",
        "age": 5
      },
      "phone": "079 704 59 36",
      "cell": "075 024 43 79",
      "id": {
        "name": "AVS",
        "value": "756.3604.3993.70"
      },
      "picture": {
        "large": "https://randomuser.me/api/portraits/women/3.jpg",
        "medium": "https://randomuser.me/api/portraits/med/women/3.jpg",
        "thumbnail": "https://randomuser.me/api/portraits/thumb/women/3.jpg"
      },
      "nat": "CH"
    }
  ],
  "info": {
    "seed": "724c4e2edd28aec2",
    "results": 1,
    "page": 1,
    "version": "1.3"
  }
}
```

Extra points

- Your user interface should contains a text box to search users by name, surname or email. Once the user stop typing, your list will be updated with users that matches with the search term.
- Support tablets with your application UI implemented to show two lists: one with all users and other with just favorite users.
- Create another view, similar to the first one, to show users marked as favorite.
- Show a list of users next to you. Show them in a list if they are at less than 1km.
- Add some checkboxes to your UI to sort users by name and gender.

What we look at

- We are interested in how you structure your code so it's easily extendable and compiles with best OO practices, we want to see if you know about most common architecture and patterns used in mobile development.
- Try to be clean, you have time enough to clean your code.
- For the extras, you will need to implement some logic, we will look for clean and understandable code and also keep in mind that you don't need to reinvent the wheel.

Hand in

Hand in your solution along with any notes, comments, and assumptions you have made while working on the solution via e-mail to the recruiter who sent you this test.

Usually, mail clients block executable files or with code, so, for sharing the code with the recruiter, a cloud tool should be used. We need a PUBLIC link (reviewers will download it) with the zip file. I you don’t have services like google drive, we recommend https://wetransfer.com/, with a couple of clicks you can get a public download URL with your ZIP file.

Do not publish the problem description or the solution you implement.