# Completing the profile service functionality

## Profile API

1. Create endpoints for the following:
    1. Get list of profiles that takes a sortBy alphabetical or reputation
    2. Get individual profile
    3. Edit profile

Test the functionality by running the postman requests in Postman and confirm working before moving on.

In order to test the sorting you might want to update the reputation score of the user that comes last alphabetically to 1. If using pgWeb you can use the following query to achieve this:

```sql
UPDATE "UserProfiles"
SET "Reputation" = 1
WHERE "Id" = '73569a32-6903-4538-89cc-9a126132ab59';
```

## NextJS App

1. Create a new link in the SideMenu.tsx to go to the profiles page.tsx
2. Create an editProfileSchema. The user can update the DisplayName and the Description.
3. Create a new actions file called profile-actions.ts and create the following functions:
    1. getProfiles(string? sortBy)
    2. getProfileById(string id)
    3. editProfile(EditProfileSchema data)
4. Create a new page in app/profiles/page.tsx. This will display a list of users and their profiles in a table. Check the HeroUI docs on how to use a Table for this. You can consult the HeroUI docs which has good examples of using Tables for this.
    1. Each row should be clickable and be a link to the users individual profile.
    2. Each column heading should be clickable to allow sorting by the column name (hint: you may want to use pgAdmin to update one of the users reputation scores so you can see this visibly working in the UI).
5. Create an individual profile page which displays the user display name, their description and their reputation. I went for a very simple approach and here is the JSX for this one just using a simple HeroUI card. Feel free to be more creative than this!

Example Profile card.

```jsx
        <Card>
            <CardHeader className='text-3xl font-semibold flex justify-between'>
                <div className='flex items-center gap-3'>
                    <Avatar className='h-30 w-30' color='secondary' />
                    <span>Display name</span>
                </div>
                <Button
                    variant='bordered'
                >
                    Edit profile
                </Button>
            </CardHeader>
            <Divider />
            <CardBody>
                <p>No profile description added yet</p>Ï
            </CardBody>
            <CardFooter className='text-xl font-semibold'>
                Reputation score: 42
            </CardFooter>
        </Card>
```

6. Create a form to edit the users profile. We are only updating the display name and description here so you can keep it simple and just use local state to swap the profile description with a form.
    1. The button will need to be visible only when the current user is viewing their own profile.
    2. Ensure on form submission the profile is visibly updated and the form is closed.
7. Update the link in the UserMenu to take them to their own profile.

## Challenge complete!!