Axon. JS Chapter
Pre-Interview Task

Prerequisites

The archive *chapter-js-task-master.zip* contains the boilerplate of the server- and client code.

In order to run the application, perform the following actions:

1. unzip chapter-js-task-master.zip
2. cd chapter-js-task-master
3. npm install && npm start

Inside the *chapter-js-task-master* folder there is a *server.js* file containing the handlers for the CRUD operations on user entity. The operations are following:

- *GET /users* - returns the list of the users
- *POST /users* - creates a new user
- *PUT /users/:id* - updates an existing user
- *DELETE /users/:id* - deletes an existing user

Additionally there is a file *data.json* that is a simplest database, accessed synchronously from the above handlers; the backup for the former is located in *backup/data.json*.

It is up to you to decide which additional dependencies to use to accomplish the task. You may also find it useful to use the *axios* & *moment* dependencies, which are already specified within *package.json*.

Task

The task is to create a page consisting of three components:

1. Table that displays the list of users
2. User creation form
3. Table summary

Components:

1. Table that displays the user data
    a. Every user is contains the following attributes:
        i. First name
        ii. Last name
        iii. Date of birth (DD/MM/YYYY)

iv. Location

b. Users list is loaded from server
c. No pagination & sorting is required
d. Each row has the “Delete” button. Once the button is clicked, the user disappears from the database and the table (with summary) is refreshed

2. User creation form
    a. Form consists of 3 text inputs, 1 date input for date-of-birth and 1 submit button
    b. Submit button is disabled until all fields are filled
    c. Once submit button is clicked the new user is created
    d. On successful submit the table (with summary) is refreshed

3. Table summary
    a. Display summary in any convenient way. You may want to use *<dl>*, *<dt>*, *<dd>* for it
    b. Summary consists of 3 values aggregated from the table data: number of users from *Kiev* or *kiev*, sum of ages of three oldest users, longest string of *first name* + *last name*

Please, do not implement functionality, that is not described above.
If *data.json* file accidentally breaks, replace it with a fresh backup from *backup/data.json.*

Optional

If this task is easy enough, create an “edit” button near each “delete” button and utilize PUT request in some way.

Layout Example

Table

▾ First Name	▾ Last Name	▾ DOB	▾ Location	▾ Actions
Bryant	Castillon	01.01.2001	kiev	Edit Delete
Martin	Callender	01.01.2002	Kiev	Edit Delete

Summary

Number of users from "Kiev or kiev": 2

Sum of three oldest user ages: 33

Longest string of first name + last name: Bryant Castillon

Form

First Name:

Last Name:

Date of birth: DD/MM/YYYY

Location:

Submit