Users-Tree-Heirarchy Application using Spring Boot

1. Spring boot REST API for a sample online Users Tree heirarchy Application. Application follows a typical Spring Boot REST structure as can be seen in code packages under src directory. Spring REST, JPA, MySQL are used.
2. DB MySQL is used. Schema : users\_tree Please refer src\main\resources\dbScripts.sql for the DDL and DML scripts which explains the Tables created and the data prepopulated. Added few users as per the Tree view diagram provided in the requirement with CEO as the root of the node tree. In User table, I defined a column BOSS\_ID which is mapped to the same table’s primary key USER\_ID. Thus for each user we will store its boss id also. Boss will be yet another user in this table. Please refer to the Screenshot Screenshot (9).png for the table row view with data in the DB for reference.
3. src\main\resources\application.properties is provided with necessary MySQL and JPA configurations.
4. Building and running the code. Like any spring boot application, first do a Run As--> Maven install. And execute the API requests in tool like POSTMAN based on the data loaded in the tables as per the SQL scripts mentioned in 2 above. Details explained below
5. As mentioned above, I have added a few users in the DB table first and passed GET request <http://localhost:8080/api/v1/users> using POSTMAN tool.

CURL command is:

curl --location --request GET '<http://localhost:8080/api/v1/users>'  
--data-raw ''

The following users are displayed recursively with their parent/boss users

[ [

    {

        "userId": 1,

        "userName": "Stella",

        "department": "CEO",

        "boss": **null**

    },

    {

        "userId": 2,

        "userName": "Luke",

        "department": "VP Mktg",

        "boss": {

            "userId": 1,

            "userName": "Stella",

            "department": "CEO",

            "boss": **null**

        }

    },

    {

        "userId": 3,

        "userName": "Peggy",

        "department": "VP Engg",

        "boss": {

            "userId": 1,

            "userName": "Stella",

            "department": "CEO",

            "boss": **null**

        }

    },

    {

        "userId": 4,

        "userName": "Meg",

        "department": "Sales",

        "boss": {

            "userId": 2,

            "userName": "Luke",

            "department": "VP Mktg",

            "boss": {

                "userId": 1,

                "userName": "Stella",

                "department": "CEO",

                "boss": **null**

            }

        }

    },

    {

        "userId": 5,

        "userName": "Ligori",

        "department": "Mktg",

        "boss": {

            "userId": 2,

            "userName": "Luke",

            "department": "VP Mktg",

            "boss": {

                "userId": 1,

                "userName": "Stella",

                "department": "CEO",

                "boss": **null**

            }

        }

    },

    {

        "userId": 6,

        "userName": "Saul",

        "department": "Manufg",

        "boss": {

            "userId": 3,

            "userName": "Peggy",

            "department": "VP Engg",

            "boss": {

                "userId": 1,

                "userName": "Stella",

                "department": "CEO",

                "boss": **null**

            }

        }

    },

    {

        "userId": 7,

        "userName": "Xavier",

        "department": "Engg",

        "boss": {

            "userId": 3,

            "userName": "Peggy",

            "department": "VP Engg",

            "boss": {

                "userId": 1,

                "userName": "Stella",

                "department": "CEO",

                "boss": **null**

            }

        }

    }

]

1. Now for registration or adding a new user and link to a boss/another user (the boss user id and details should already exist in the table), submit a POST request <http://localhost:8080/api/v1/user> with user details as a JSON body request as example below

{

"userName": "Lotta",

"department": "Sales Rep",

"boss": {

"userId": 4,

"userName": "Meg",

"department": "Sales"

}

}

CURL command as follows :

curl --location --request POST '<http://localhost:8080/api/v1/users>'  
--header 'Content-Type: application/json'  
--data-raw '{ "userName": "Lotta", "department": "Sales Rep", "boss": { "userId": 4, "userName": "Meg", "department": "Sales"

}

}'

After successful return of the POST request. We can again query all users using GET <http://localhost:8080/api/v1/users>

as explained in step 5 above. The newly added users with their boss and recursive bosses are displayed in the response see userid : 14 and 16 as below for new users added via POST request.

[

    {

        "userId": 1,

        "userName": "Stella",

        "department": "CEO",

        "boss": **null**

    },

    {

        "userId": 2,

        "userName": "Luke",

        "department": "VP Mktg",

        "boss": {

            "userId": 1,

            "userName": "Stella",

            "department": "CEO",

            "boss": **null**

        }

    },

    {

        "userId": 3,

        "userName": "Peggy",

        "department": "VP Engg",

        "boss": {

            "userId": 1,

            "userName": "Stella",

            "department": "CEO",

            "boss": **null**

        }

    },

    {

        "userId": 4,

        "userName": "Meg",

        "department": "Sales",

        "boss": {

            "userId": 2,

            "userName": "Luke",

            "department": "VP Mktg",

            "boss": {

                "userId": 1,

                "userName": "Stella",

                "department": "CEO",

                "boss": **null**

            }

        }

    },

    {

        "userId": 5,

        "userName": "Ligori",

        "department": "Mktg",

        "boss": {

            "userId": 2,

            "userName": "Luke",

            "department": "VP Mktg",

            "boss": {

                "userId": 1,

                "userName": "Stella",

                "department": "CEO",

                "boss": **null**

            }

        }

    },

    {

        "userId": 6,

        "userName": "Saul",

        "department": "Manufg",

        "boss": {

            "userId": 3,

            "userName": "Peggy",

            "department": "VP Engg",

            "boss": {

                "userId": 1,

                "userName": "Stella",

                "department": "CEO",

                "boss": **null**

            }

        }

    },

    {

        "userId": 7,

        "userName": "Xavier",

        "department": "Engg",

        "boss": {

            "userId": 3,

            "userName": "Peggy",

            "department": "VP Engg",

            "boss": {

                "userId": 1,

                "userName": "Stella",

                "department": "CEO",

                "boss": **null**

            }

        }

    },

    {

        "userId": 14,

        "userName": "Lotta",

        "department": "Sales Rep",

        "boss": {

            "userId": 4,

            "userName": "Meg",

            "department": "Sales",

            "boss": {

                "userId": 2,

                "userName": "Luke",

                "department": "VP Mktg",

                "boss": {

                    "userId": 1,

                    "userName": "Stella",

                    "department": "CEO",

                    "boss": **null**

                }

            }

        }

    },

    {

        "userId": 16,

        "userName": "Dot",

        "department": "Sales Rep",

        "boss": {

            "userId": 4,

            "userName": "Meg",

            "department": "Sales",

            "boss": {

                "userId": 2,

                "userName": "Luke",

                "department": "VP Mktg",

                "boss": {

                    "userId": 1,

                    "userName": "Stella",

                    "department": "CEO",

                    "boss": **null**

                }

            }

        }

    }

]

1. The following features not implemented:

7.1 ) Front end view based on the JSON returned due to time limiation to explore and implement. I believe the front end view can be implemented using JQuery and its some plugin..

7.2 ) Points based system ie points credited to a user recursively based on a new user registration and linking.