SUBORDINATION TREE PROJECT
CG-VAK
February 3, 2025

ABSTRACT - The following describes a web application that visualizes a user-subordination tree. It includes simple AUTHENTICATION, ONE MAIN SCREEN with a CHANGE MY PASSPHRASE button, a CREATE NEW USER button, and a FORGOTTEN PASSPHRASE drop-down with associated button.

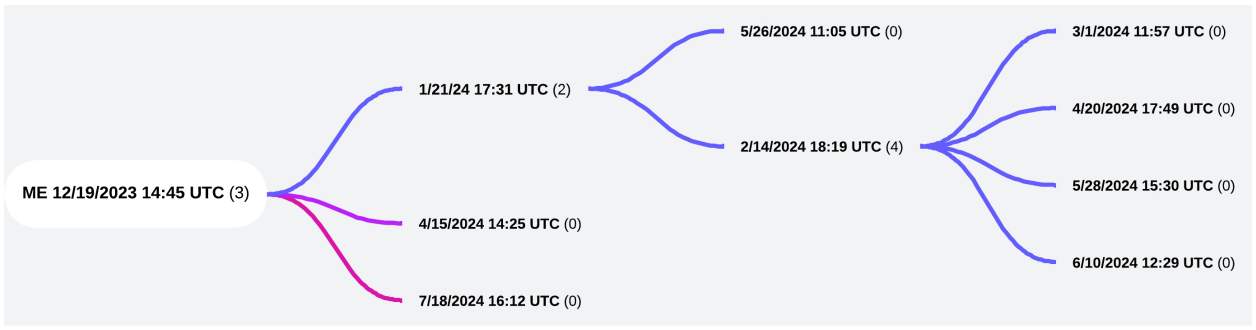
AUTHENTICATION - Authentication is a single field and there is no MFA. A simple Capthca-type challenge or equivalent should be implemented. The user enters an access code or passphrase that can include spaces, is at least 20 characters long, uses at least one special character, includes at least one capital letter, and includes at least one special character. If the code is found, then the user should gain access. If not, then the user does not have access. Once a user authenticates, they are considered the root user for the purposes described below.

CHANGE MY PASSPHRASE BUTTON - This button takes the user through a common flow to update his/her access code/passphrase. This update is required on first authentication.

CREATE NEW USER BUTTON - Clicking this button creates a new temporary access code/passphrase that can be given away to another user. After the button is clicked and before the access code/passphrase is created, a follow-up confirmation (YES/NO dialog box) is shown confirming that the user actually wants to create a new access code/passphrase. Every time a new access code/passphrase is created, it is stored and associated in subordination to the user who created it. One user can create an unlimited number of access codes/passphrases. If a new user is given an access code/passphrase from an existing user, they too can create and unlimited number of access codes/passphrases after they authenticate.

SCREEN - The second main use case is the ability to see a visualization tree of the authenticated user and the users subordinate to that user, displaying tallies at each level. A tree sample is shown below and the resulting tree in the application does not have to be exact, but should be the nicest cleanest equivalent of the tree. In this case, subordinations are shown to the right. Each user, including the main user, is represented only by a date/time stamp (4/15/2024 14:25 UTC) as to when they were created. Next to each user's date/

time stamp is a total, in parenthesis, showing the total number of users directly created by (directly subordinate to) the root user. At the top of the screen is a nicely displayed widget showing the total number of users and subordinate user underneath a particular user, including the root user, but not counting anyone above the root user. The total from the tree sample below would be 10.



FORGOTTEN PASSPHRASE DROP-DOWN and BUTTON - If an access code/passphrase is forgotten, there is NO option such as “Forgot Password” for a user to restore access. Instead, if a user created by the root user forgets his/her access code/passphrase, he/she must go to the root user who initially gave them their access code/passphrase. That root user must select the date/time indicator of the correct subordinate user from the drop-down and select the associated button. A confirmation YES/NO dialog should display in regard to creating a new temporary access code/passphrase for that user. Once confirmed, the new access code/passphrase should be shown so the root user can give it to the subordinate user.

OTHER GENERAL INFORMATION
-We will set up a repository for you to use.
-We will provide the AWS account access and a URL.
-They should use a serverless approach, a single-table Dynamo DB no-SQL database, Next.js, and Serverless Stack version 3 to implement this project. The approach should be as cost-effective as possible in terms of limiting monthly AWS costs.
-Any libraries used should be open source.
-The look and feel should be clean, neat, and modern. If responsiveness is considered, that would be a plus.