Team M
Cycle 2
Requirements/Use Cases Document

Functional Requirements:

1. A user can search for an existing person by entering the first or last name.
    a. Information to be provided by the user:
        i. First name or Last name
    b. The results will provide the user with the person or persons which share the name given. If no person is found by this name there will be an error returned
2. A user can search for the grandparents of an existing person
    a. Information to specify:
        i. pID of the known person
    b. The results will provide the person who was searched for and their grandparents.
    c. Limits: the user must know the pID of the person whose grandparents they want to find
3. A user can search for an existing person by their pID
    a. Information to be provided:
        i. pID
    b. The result will be a person associated with that specific pID, or an error will shown that the pID does not belong to anyone
4. A user can add a new person to the app
    a. Automatically pass (1) to check if the person is in the app or not
    b. Information the user needs to specify:
        i. Name
        ii. Birthday
        iii. City of birth/residence
        iv. Any known family relationships (spouse, parents, children)
        v. pID
    c. The new person will then be added to the existing tree.
5. For any person, find people of a specified relationship
   **Utilizing the relationship using "mother of"," father of" or "child of" is the least erroneous way to navigate a specific relationship. This is because there can be a possibility of multiple grandparents and multiple grandchildren within any branch of the overall family tree.
    a. Parent is <u>Mother(Full name) is mother of "c"</u>, <u>Father(Fullname) is father of "c".</u>
    b. Grandparent is searched through "mother of".
    c. Child is <u>Child(Full name) is child of "a"</u>
    d. Grandchild is searched through "child of".

    e. Cousin is searched through “child of”.
    f. Partner is a person who does not have “child of” defaults as partner.

6. A user can provide two pIDs to determine if they are related (they will have a common ancestor somewhere “up” in their family tree)
    a. The user will provide the pID of each of the existing persons they want to search for a relationship of.
    b. The system will search for the common ancestor and return them or return an error if not found.
    c. Limits: only shows first related ancestor, does not continue to check for other ones
7. A user can search for an existing relationship using an rID
    a. User has to specify:
        i. rID
    b. The result will be the information for the two people of the relationship specified or if rID has no relationship associated with it, an error will be returned.

Nonfunctional Requirements:

1. Use Java v11 and IntelliJ for our program development. Save our work in the course GIT repository for your team: Comp330Fall2020TeamM
2. Structure the system so that the user interface is separate from the logic and searching functions.
3. Have a well-structured functional decomposition of the app into separate parts. This decomposition should support separate development of key components by individual programmers.

Use Case Narratives:

Event: A user provides a file to read in
Actor: Person
Purpose: To read in a file from which to create a family tree from
Overview: The actor provides the system with a file, the system reads the file and provides the graph to the user. If the file is wrongly formatted, the system returns an error to the actor.
Precondition: The user must have a file to input.
Postcondition: The user can print a visual of a family tree.

Use Case Narrative:
User
1. The user uploads a file to be read into the app
4. The user receives the output of a graph

System
2. The system reads the file and separates the pieces of information into a graph data structure
3. The system outputs the graph to the user

Alternative Flow:
Line 2: File structure is incorrect and the system cannot process information. Print error. Return to step 1.

Event: A user wants to determine if a person is known to the app
Actor: Person
Purpose: A user can check if a person is known in the family tree to the system
Overview: The actor provides the necessary information to search for a person and submits it. The system traverses the tree until it finds the person and returns them along with related info. If no person found the system prints an error.
Precondition: The user must have information to provide about a person
Postcondition: The user will know if a person is or is not known to the app

Use Case Narrative:
User
1. The user is given options for information to be entered
    a. Name (required)
    b. Birthdate
    c. Place of birth
    d. Spouse/Parents/Children

    e. pID
2. The user submits the entered information for the search
5. The system outputs the person along with related information

System
3. The system searches through the tree of people until a person with matching information is found.
4. The system outputs the person and their information and relationships to the user

Alternative Flow:
Line 3: The system searches through the system and does not find a match. Prints a system error. Return to step 2.

Event: A user wants to determine if two existing people are related
Actor: Person
Purpose: Determine if a relationship exists between two people.
Overview: The actor provides the needed info for two people and submits it. The system will traverse the tree to find both people, if one, both or none are found an error is returned. If found, the system will then traverse the nodes related to each until a relationship is found, if no relationship found an error is outputted. If a relationship is found, the system returns the relationship to the actor.
Precondition: The user must provide info on two existing people
Postcondition: The user will know whether two people are related or not.

Use Case Narrative:
User:
1. User inputs names of the two people they want to find a relationship for
2. User submits the info into the search
7. The user receives the output displaying the relationship

System:
3. The system will traverse the tree towards the root for each person
4. It will create a list for each person of related ancestors and compare each person added, checking if they are identical
5. If the system finds a person who overlaps for both, this person is outputted to the user as the member both are related to
6. The system outputs the relationship

Alternative Flow:
Line 3: If the tree is traversed and one or both people cannot be located, the system will print an error. Return to step 2.

Line 4: Once the whole tree is traversed and no relationship is found between the people the system will output an error that there is no relationship between the two people. Go to step 7