Milestone 3

Key Algorithms:

Find LCA (Lowest Common Ancestor) of X and Y and relationship of X and Y:

1. Find path from X to its most root node
2. Find path from Y to its most root node
3. Iterate paths from root to X / Y until the node is not matched
4. Get LCA using step 3
5. Count level (generation) using LCA to last node in each path and store it in Nx
6. Count level (generation) using LCA to last node in each path and store it in Ny
7. Find relationship between x and y using min{nx - ny} – 1 cousins |nx - ny| removed

list the ancestors of person X for Z generations:

1. Find parent of X in db using parent child relationship table
2. Increment count to 1
3. Add into array list
4. Repeat step 1,2 and 3until count reaches to Z

list the descendents of person X for Z generations:

1. Find child of X in db using parent child relationship table
2. Increment count to 1
3. Add into array list
4. Recursice call to child of X
5. Repeat step 1,2,3 and 4 until count reaches to Z

list the pictures of a particular tag within some time range:

1. Find all images from table of tag and file
2. Perform join operation using file_id and attributes of file
3. Get final set of pictures

list the pictures of a particular place within some time range:

1. Find all images from table of locations and file
2. Perform join operation using file_id and attributes of file
3. Get final set of pictures

list the pictures of a given set of people (like a couple) within some time range chronologically:

1. Iterate the set of people
2. Find pictures related to that person using person id

3. Filter that picture using other table using dates
4. Record final set of pictures

list the pictures that include all the immediate family members of person X (immediate children):

1. Find immediate family member of X
2. Find a file that contain all the family member (immediate) of X
3. Return final set of pictures

Method distribution to the various classes:

---

1. **PersonIdentity:**
    - PersonIdentity addPerson(String name)
    - Boolean recordAttributes(PersonIdentity person, Map<String, String> attributes)
    - Boolean recordReference(PersonIdentity person, String reference)
    - Boolean recordNote(PersonIdentity person, String note)
    - PersonIdentity findPerson(String name)
    - FileIdentifier findMediaFile(String name)
    - String findName(PersonIdentity id)
    - List<String> notesAndReferences(PersonIdentity person)

2. **Genelogy:**
    - Boolean recordChild(PersonIdentity parent, PersonIdentity child)
    - Boolean recordPartnering(PersonIdentity partner1, PersonIdentity partner2)
    - Boolean recordDissolution(PersonIdentity partner1, PersonIdentity partner2)
    - Boolean peopleInMedia(FileIdentifier fileIdentifier, List<PersonIdentity> people)
    - Set<FileIdentifier> findMediaByTag(String tag , String startDate, String endDate)
    - Set<FileIdentifier> findMediaByLocation(String location, String startDate, String endDate)
    - List<FileIdentifier> findIndividualsMedia(Set<PersonIdentity> people, String startDate, String endDate)

3. **FileIdentifier**:
    - FileIdentifier addMediaFile(String fileLocation)
    - Boolean recordMediaAttributes(FileIdentifier fileIdentifier, Map<String, String> attributes)

- Boolean tagMedia(FileIdentifier, fileIdentifier, String tag)
- String findMediaFile(FileIdentifier fileId)

4. BiologicalRelation:

- BiologicalRelation findRelation(PersonIdentity person1, PesonIdentity person2)
- Set<PersonIdentity> descendents(PersonIdentity person, Integer generations)
- Set<PersonIdentity> ancestores(PersonIdentity person, Integer generations)
- List<FileIdentifier> findBiologicalFamilyMedia(PersonIdentity person)

Database schema:

1. Family tree

Person information:

Person_id	Name

Person attributes:

Person_id	Key	Value

Person reference:

Person_id	Reference

Person note:

Person_id	Note

Person child / parent relation:

Id	Parent_id	Child_id

Person partnering relation:

Id	Partner_1	Partner_2

Person dissolution relation:

Id	Partner_1	Partner_2

2. Media archive

File information:

File_id	Name

File attributes:

File_id	Key	Value

File with person:

File_id	Person_id

File with tags:

File_id	tag

Data structure used in code:

1. Array
2. Sorted List
3. ArrayList
4. Basic OOPs concepts

5. Interface
6. Tree based implementation
7. LinkedList

White Box tests:

- Add person name that is out of bound to memory of database
- Add person name with duplicate name in database
- Add relationship in which already other type of relationship is exist (like if person 1 and person 2 has parent child relationship then they don't have an partnering relationship and dissoultion resltionship)
- Add parent child relationship between person X and person Y, person Z with person Y. So, person Y is child of both X and Y