Business user test plan

1. Manage the family tree

PersonIdentity addPerson (String name)

Input Validation

- Name is either empty or null. Should throw Illegal argument exception.
- Name again comes for addition with same case or different lower/upper case. Should add a new person.
- Only numbers passed. Should add name.
- When alphanumeric name is passed. Should add name.

Boundary

- 1 letter name passed. Should add name.
- Long string passed. Should add name.

Control flow

- Unable to add name inside database. Should throw IO Exception.

Boolean recordAttributes (PersonIdentity person, Map<String, String> attributes)

Input Validation

- When null Person Identity is passed. Should throw Illegal argument exception.
- When null map is passed, or map size is 0 for attributes. Should throw Illegal argument exception.
- When attributes with either key or value as null or blank is passed. Should throw Illegal argument exception.
- When pre-defined attributes are passed inside the map. Should add them to the person's table.
- When new attributes or notes or references are passed. Should add them to the person_attributes table.
- When new attributes but already added before are passed existing values are updated except for references and notes where new entries are created every time inside person_attributes table.
- If same value is passed for note or reference for a person, returns false.
- When any of the attribute fails to get added, do not add any attribute to the database.
- If location attribute is passed and locations table has an entry for it, do not add it again use that value.
- If gender attribute is passed and gender table has an entry for it, do not add it again use that value.
- If occupation attribute is passed and occupations table has an entry for it, do not add it again use that value.

- If date of birth is greater than date of death, do not record it.
- Should add date when only year is passed.

Boundary

- When 0 attribute is passed inside the Map. Should return false.
- When only 1 attribute is passed inside the Map. Should return true.
- When all the attributes are passed inside the Map. Should return true.

Control flow

- When a new attribute comes when it is not present inside database. Should add new attribute and its value.
- When one of the attributes is not saved inside database. Return false.

Boolean recordReference (PersonIdentity person, String reference)

Input Validation

- When null Person Identity is passed. Should throw Illegal argument exception.
- When empty or null reference is passed. Should throw Illegal argument exception.
- When valid reference is passed. Should add reference and return true.

Boundary

- When reference is added for the first time. Should add reference and return true.
- When reference for the same person is added n time. Should add reference and return true.
- When reference with same value is added for a person. Should not add reference and return true.

Boolean recordNote (PersonIdentity person, String note)

Input Validation

- When null Person Identity is passed. Should throw Illegal argument exception.
- When empty or null note is passed. Should throw Illegal argument exception.
- When alpha numeric note is passed. Should add note and return true.
- When valid note is passed. Should add note and return true.

Boundary

- When note is added for the first time. Should add note and return true.
- When note for the same person is added n time. Should add note and return true.
- When note with same value is added for a person. Should not add note and return true.

Boolean recordChild (PersonIdentity parent, PersonIdentity child)

Input Validation

- When null is passed in either of parent or child or in both. Should throw Illegal argument exception.
- When a parent or a child that does not exist is passed. Should return false.
- When both the parent and child are same. Should throw Illegal argument exception.
- When there is already a similar relation between parent and child. Should return true.
- When there is already an inverse relation between parent and child. Should return false.
- When there is already a partnership/dissolution relation between parent and child. Should return false.

Boundary

- When relation is added for the first time between parent and the child. Should add relation and return true.
- When relation is added for the nth time between any parent and the child. Should add relation and return true.

Data flow

- Check if both the parent and child has been updated. Parent should have a new child and the child should have a new parent inside the person's map.
- Check if parent has a partner, if it has, add parent child relationship between partner and child.
- Check if any of the change fails midway in the database, it should rollback done the previous changes done.
- Check if persons map gets updated after all the changes are updated inside the database.

Boolean recordPartnering (PersonIdentity partner1, PersonIdentity partner2)

Input Validation

- When null is passed in either of partner 1 or partner 2 or in both. Should throw Illegal argument exception.
- When a partner 1 or partner 2 that does not exist is passed. Should return false.
- When both the partner 1 and partner 2 are same. Should throw Illegal argument exception.
- When there is already a partner relation between partner 1 and partner 2. Should return true.
- When there is already a parent-child relation between partner 1 and partner 2. Should return false.
- When there is already a dissolution relation between partner 1 and partner 2. Should add relation and return true.

Boundary

- When relation is added for the first time between partner 1 and partner 2.
- When relation is added for the nth time between any partner 1 and partner 2.

Control flow

- Should record partnership between person 1 and person 2 and vice-versa inside the person_relations table and mark the status as Active.

Data flow

- Check if persons map gets updated after all the changes are updated inside the database.

Boolean recordDissolution (PersonIdentity partner1, PersonIdentity partner2)

Input Validation

- When null is passed in either of partner 1 or partner 2 or in both. Should throw Illegal argument exception.
- When a partner 1 or partner 2 that does not exist is passed. Should return false.
- When both the partner 1 and partner 2 are same. Should throw Illegal argument exception.
- When there is already a dissolution relation between partner 1 and partner 2. Should return true.
- When there is already a parent-child relation between partner 1 and partner 2. Should return false.
- When there is already a partner relation between partner 1 and partner 2. Should update relation and return true.
- When there is no partner relation between partner 1 and partner 2. Should return false.
- When there is partnership relationship with anyone. Should return false.

Boundary

- When relation is added for the first time between partner 1 and partner 2. Should add relation and return true.
- When relation is added for the nth time between any partner 1 and partner 2. Should add relation and return true.

Control flow

- If both partners are present. Should remove each other as partners.

Data flow

- Check if persons map gets updated after all the changes are updated inside the database.

2. Manage the media archive

FileIdentifier addMediaFile(String fileLocation)

Input Validation

- When empty or null is passed as a file location. Should throw Illegal argument exception.
- When location does not exist. Should throw Illegal Argument exception.
- When same file location is passed. Should return existing file identifier.
- When file with same name but different file location is passed. Should add and return file identifier.

Boundary

- When file location is added for the first time. Should add and return file identifier.
- When file location is added for the n time. Should add and return file identifier.
- 1 letter file is passed. Should add name.
- Long string passed. Should add name.

Boolean recordMediaAttributes(FileIdentifier fileIdentifier, Map<String, String> attributes)

Input Validation

- When null file Identifier is passed. Should throw Illegal argument exception.
- When null map is passed, or map size is 0 for attributes. Should throw Illegal argument exception.
- When attributes matching the name of the actual attributes are passed inside the map. Should record attributes and return true.
- When attributes with either key or value as null or blank is passed. Should throw Illegal argument exception.
- When pre-defined attributes are passed inside the map. Should add them to the media table.
- When new attributes or tag is passed. Should add them to the media_attributes table.
- When new attributes but already added before are passed existing values are updated except for tag where new entries are created every time inside media_attributes table.
- If same value is passed for tag for a file, returns false.
- When any of the attribute fails to get added, do not add any attribute to the database.
- If location attribute is passed and locations table has an entry for it, do not add it again use that value.

Boundary

- When 0 attribute is passed inside the Map. Should return false.
- When only 1 attribute is passed inside the Map. Should record attribute and return true.
- When all the attributes are passed inside the Map. Should record attribute and return true.

Control flow

- When a new attribute comes when it is not present inside database. Should add new attribute and its value.
- When one of the attributes is not saved inside database. Return false.

Boolean peopleInMedia(FileIdentifier fileIdentifier, List<PersonIdentity> people)

Input Validation

- When null file identifier or people list is passed. Should throw illegal argument exception.
- When update of 1 or more persons fails. Should not add any of the person for a given file.
- When all the people passed in the list of persons have been updated. Should add and return true.
- When file does not exist, no entries for the person goes inside the database.

Boundary

- When only 1 person is passed in the list.
- When n person is passed in the list.

Boolean tagMedia(FileIdentifier fileIdentifier, String tag)

Input Validation

- When null file identifier is passed. Should throw illegal argument exception.
- When null tag is passed. Should throw illegal argument exception.
- When tag with same value is added for a file. Should not add tag and return true.

Boundary

- When a tag is passed for the first time for a file.
- When a tag is passed for the n time for a file.

3. Reporting

PersonIdentity findPerson(String name)

Input Validation

- When null or empty name is passed. Should throw Illegal argument exception.
- When a name does not exist in the system. Should return null.
- When a name exists in the system. Should return person identity.

Boundary

- When a name is searched for the first time. Should return person identity.
- When same name is searched for n time. Should return person identity.

FileIdentifier findMediaFile(String name)

Input Validation

- When null or empty name is passed. Should throw Illegal argument exception.
- When a name does not exist in the system. Should return null.
- When a name exists in the system. Should return file identifier.

Boundary

- When a name is searched for the first time. Should return file identifier.
- When same name is searched for n time. Should return file identifier.

String findName(PersonIdentity id)

Input Validation

- When null is passed. Should throw Illegal argument exception.
- When person identity does not exist in the system. Should return null.
- When valid person identity is passed. Should return name.

Boundary

- When id is searched for the first time. Should return name.
- When same id is searched for n time. Should return name.

String findMediaFile(FileIdentifier fileId)

Input Validation

- When null is passed. Should throw Illegal argument exception.
- When file identifier does not exist in the system. Should return null.
- When valid field identifier is passed. Should return name.

Boundary

- When field is searched for the first time. Should return name.
- When same field is searched for n time. Should return name.

BiologicalRelation findRelation(PersonIdentity person1, PesonIdentity person2)

Input Validation

- When null is passed for either person1 or person 2 or both. Should throw Illegal argument exception.
- When 1 or both persons do not exist in the system. Should return null.
- When no relation exists between persons. Should return null.

- When relation exists between persons. Should return relation.

Control flow

- When all the persons are added, find the relation when 2 persons are siblings.
- Find the relation when 2 persons have more than 1 common ancestor, should return cousinship and degree of removal according to the lowest common ancestor.
- Find the relation between 2 persons, when 1 is ancestor of the another.
- Find the relation between 2 persons, when they don't have a common ancestor.

Set<PersonIdentity> descendants(PersonIdentity person, Integer generations)

Input Validation

- When null is passed as person. Should throw Illegal argument exception.
- When negative integer is passed as generations. Should throw Illegal argument exception.
- When generations are larger than actual level of generations. Should return all the descendants.

Boundary

- When 1 is passed as generations. Should return immediate descendants.
- When n is passed as generations. Should return n generation descendants.

Set<PersonIdentity> ancestors(PersonIdentity person, Integer generations)

Input Validation

- When null is passed as person. Should throw Illegal argument exception.
- When negative integer is passed as generations. Should throw Illegal argument exception.
- When generations are larger than actual level of generations. Should return all the ancestors.

Boundary

- When 1 is passed as generations. Should return immediate ancestors.
- When n is passed as generations. Should return n generation ancestors.

List<String> notesAndReferences(PersonIdentity person)

Input Validation

- When null is passed as person. Should throw Illegal argument exception.
- When person does not exist in the system. Should return null.
- When person exists in the system. Should return notes and references for the person.

Boundary

- When either of note or reference exists for a person. Should return existing notes or references for the person.
- When both notes and reference exist for a person. Should return notes and references for the person.

Set<FileIdentifier> findMediaByTag(String tag , String startDate, String endDate)

Input Validation

- When null or empty is passed as tag. Should throw Illegal argument exception.
- When start or end of date is ahead of current date. Should throw Illegal argument exception.
- When start date is greater than current date. Should throw Illegal argument exception.
- When null is passed as start or end date. Should return files for a tag.

Boundary

- When current date is 1st January, 0001. Should return files for existing individuals.
- When end date is current date. Should return files for existing individuals.

Set<FileIdentifier> findMediaByLocation(String location, String startDate, String endDate)

Input Validation

- When null or empty is passed as location. Should throw Illegal argument exception.
- When start or end of date is ahead of current date. Should throw Illegal argument exception.
- When start date is greater than current date. Should throw Illegal argument exception.
- When null is passed as start or end date. Should return all the files for a location.
- When location with existing database location but extra information is passed. Should return null.

Boundary

- When current date is 1st January, 0001. Should return files for existing individuals.
- When end date is current date. Should return files for existing individuals.

List<FileIdentifier> findIndividualsMedia(Set<PersonIdentity> people, String startDate, String endDate)

Input Validation

- When null or empty is passed as people. Should throw Illegal argument exception.
- When start or end of date is ahead of current date. Should throw Illegal argument exception.
- When start or end date is greater than current date. Should throw Illegal argument exception.
- When null is passed as start or end date. Should return all the files for existing individuals.
- If any of the individual does not exist, do not return its data but return for those who exists.

- Files are distinct and sorted according to their dates, then according to their names and then, no dates at the end.

Boundary

- When a set with only 1 person is passed. Should return files for that individual.
- When a set with n persons is passed. Should return files for existing individuals.
- When current date is 1st January, 0001. Should return files for existing individuals.
- When end date is current date. Should return files for existing individuals.

List<FileIdentifier> findBiologicalFamilyMedia(PersonIdentity person)

Input Validation

- When null is passed as person. Should throw Illegal argument exception.
- When a person does not exist in the system. Should throw exception.
- When a person exists but no immediate children present. Should return empty list.
- When a person exists in the database. Should return list of distinct files for immediate children in the order picture was taken and files without any date at the end of the list. If 2 media were taken on same date, return them according to the alphabetical order.

Boundary

- When a person with only 1 child exists in the database with multiple files. Should return files for that child in the list.
- When a person with only 1 child exists in the database with single file. Should return single file for that child in the list.