# COURSE PROJECT

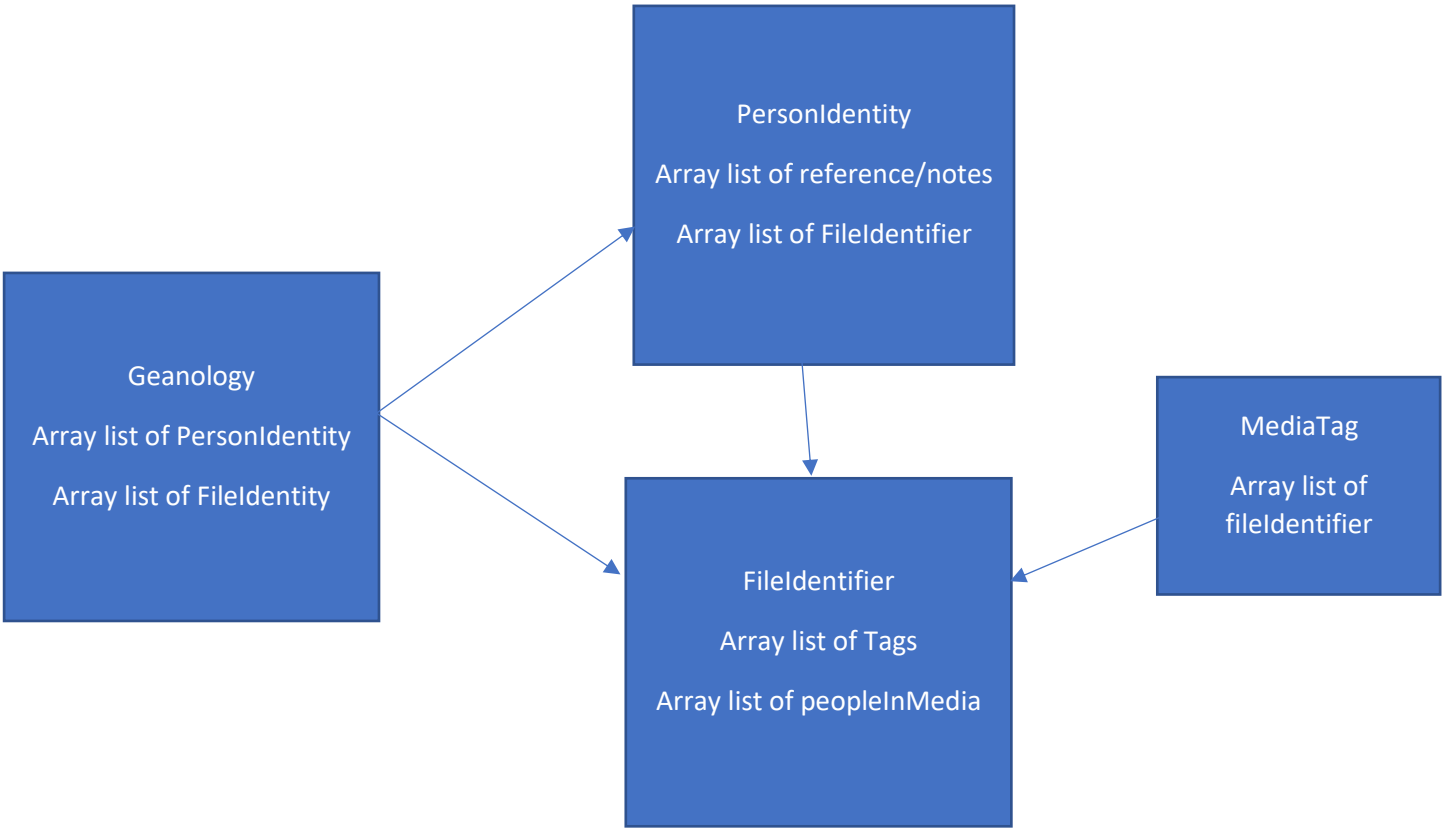
26th NOV 2021

## MILESTONE 3: External documentation of data structures, code design, key algorithms.

## Overview:

Genealogy is a class where we have a family relation between individuals and associated media files and tags and help answer question about how a person is related to another person and get more data on pictures and tags a person is present in.

## Main Class Diagram:




**Brief about each class:**

Genealogy is the main class that hold everything to gather it contains both person identity and identifier which and build relation and performs data operations through various functions and reports them back to the user as well.

Person Identity is a class that represents an instance of an individual and store all his attributes like name, gender, occupation, date and location of birth and death.

File identifier is a class which represents a media file which has a list of tags, location, date, place, city, province, country and people in media.

Media Tag is a class that represent tags and store an array list of File Identifiers.

## Data structures and their relation to each other:

Genealogy is a class that holds **Array list of <personIdentity>**, **Array list of <Fileidentifier>**. To manage and perform and build relations between each other.

Each **<personIdentity>** in turn stores all the attributes of a person along with **Array list of <Strings>** for reference and media files, Array list of **<FileIdentifiers>** to store the media that they are present in, Array list of **<PersonIdentity>** to store children and parent relations.

Each **<FileIdentifier>** will have Array List of **<PersonIdentity>** and Array list of **<MediaTags>** to store the tags associated with the media files, along with the attributes like file location, date, place, city, province, country.

Media Tag will have Array List of **<File Identifier>** to keep track of the media files associated with the tags.

## Code design and algorithm:

With each class separated with its functionality and storing its own attributes Genealogy encapsulates all the functionality of file Identifier and person Identity class and handles all the functionality of inserting the it to the data base.

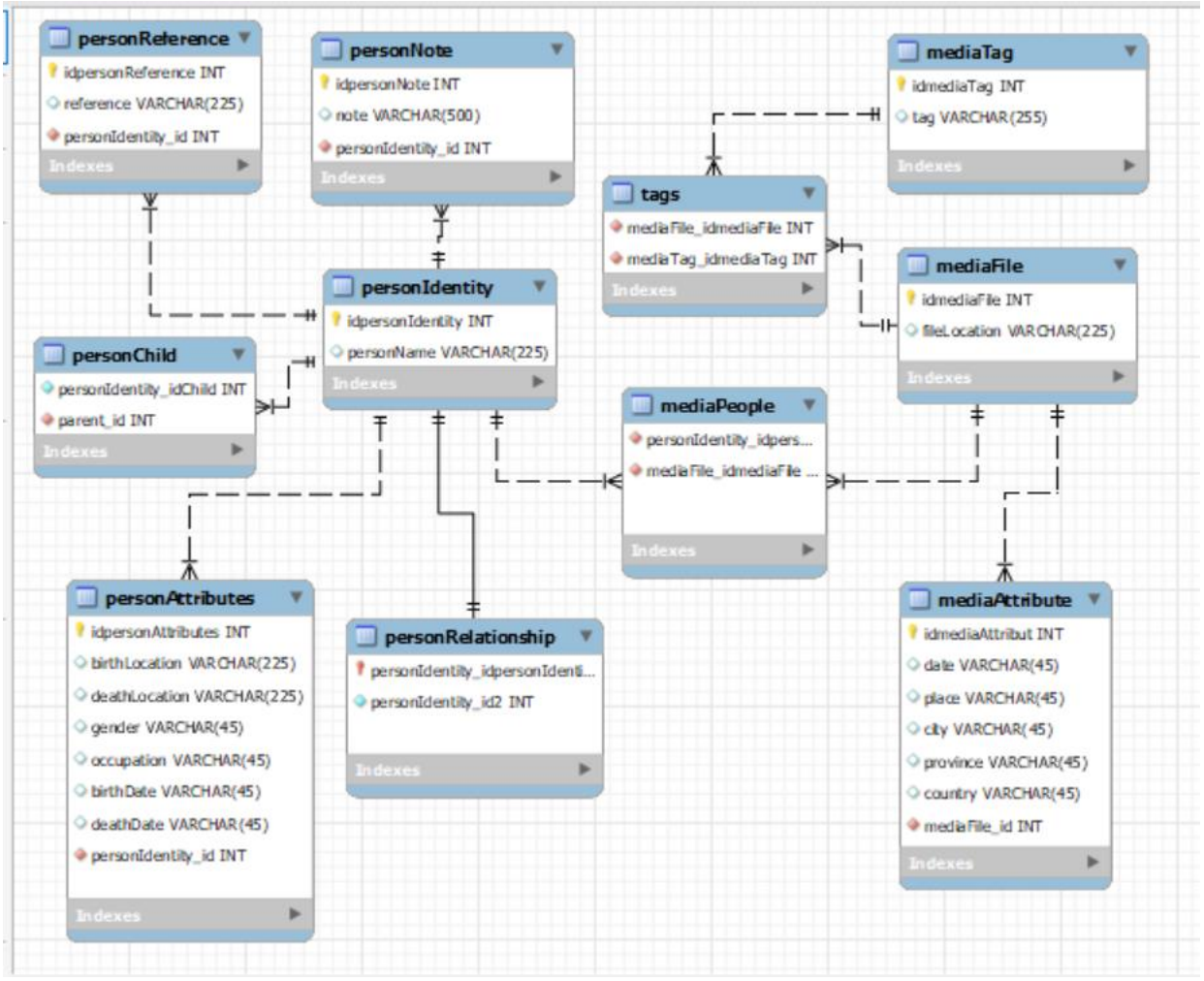
Each person will hold information and his direct relations with others in the geology and both parent and children for easy traversal on both sides of the tree.

File identifier hold all the people associated and tags in it and people in media along with its attributes to identify all the info, in turn the media files have all the tags and attributes.

Media tag class stores all the files with a particular tag.

We can traverse a family tree in turn to get the data processing to find out the relation between 2 individuals and in turn store all the data in the data base, and update them as they change.

## DATABASE ER DIAGRAM:




All the data is store in the data base and updated as we go on.

There are 12 tables normalized to an extent to implement the best data design principles and to store data of people and associated media files and store attributes and relation between them.

**Details about table specification:**

*Table:* **mediaattribute**

*Columns:*

- idmediaAttribut int PK
- date varchar (45)
- place varchar (45)
- city varchar (45)
- province varchar (45)
- country varchar (45)
- mediaFile_id int

*Table:* **mediafile**

*Columns:*

- idmediaFile int AI PK
- fileLocation varchar (225)

*Table:* **mediapeople**

*Columns:*

- personIdentity_idpersonIdentity int
- mediaFile_idmediaFile int

*Table:* **mediatag**

*Columns:*

- idmediaTag int AI PK
- tag varchar (255)

*Table:* **personattributes**

Columns:

- idpersonAttributes int AI PK
- birthLocation varchar (225)
- deathLocation varchar (225)
- gender varchar (45)

- occupation varchar (45)
- birthDate varchar (45)
- deathDate varchar (45)
- personIdentity_id int

*Table:* **personchild**

*Columns:*

- personIdentity_idChild int
- parent_id int

*Table:* **personidentity**

*Columns:*

- idpersonIdentity int AI PK
- personName varchar (225)

*Table:* **personnote**

*Columns:*

- idpersonNote int AI PK
- note varchar (500)
- personIdentity_id int

*Table:* **personreference**

*Columns:*

- idpersonReference int AI PK
- reference varchar (225)
- personIdentity_id int

*Table:* **personrelationship**

*Columns:*

- personIdentity_idpersonIdentity int PK
- personIdentity_id2 int

*Table:* **tags**

*Columns:*

- mediaFile_idmediaFile int
- mediaTag_idmediaTag int

## white box test cases

**only for reporting**

- **PersonIdentity findPerson (String name)**
  - Pass the same name as that of a person that exists in the data base
  - Add when there is no data in the data base
  - Add when there is lots of people in the data base
  - Try and add if there is no connection to the data base

- **FileIdentifier findMediaFile (String name)**
  - Add media when no file exists in the data base
  - Add media when there are lots of data in the data base.
  - Add when there is no connection in the data base.
  - Add where there is a connection with the media file already present

- **String find Name( Person Identity id )**
  - When there is no person present in the data base
  - When the person exists in the data base
  - When a lot of people exits in the data base
  - When null is passed

- **String findMediaFile( File Identifier field )**
  - When there is no file present in the data base
  - When the file exists in the data base
  - When a lot of files that exits in the data base
  - When null is passed into the column.

- **BiologicalRelation findRelation( PersonIdentity person1, PesonIdentity person2 )**

- When there is no relation between people
- When there is relation between 2 people
- When either one of then does not exist
- When there is a dissolution before

- **Set<PersonIdentity> descendents( PersonIdentity person, Integer generations )**
  - If there are no decedents to a person
  - If there is more than their level descendant's
  - If there are many levels of descendants.
  - If 0 is passed as an integer
  - If null is passed to the person

- **Set<PersonIdentity> ancestores(Person Identity person, Integer generations)**
  - If the person doesn't have any ancestors
  - If the person has more than 3 levels of ancestors
  - If the person is not present in the data base
  - If the 0 is passed inside the integers

- **List<String> notesAndReferences( PersonIdentity person )**
  - Get notes and reference when a person doesn't have any
  - When a person is not present in data base
  - If a person is present but doesn't have any reference and only notes
  - If a person had only reference.

- **Set<FileIdentifier> findMediaByTag(String tag , String startDate, String endDate)**
  - When tag does not exist in the data base
  - When tag exists and contains no data
  - When tag is associated with lot of media files

- **Set<FileIdentifier> findMediaByLocation(String location, String startDate, String endDate)**
  - When location is not associated with any media files
  - When null is passed for dates
  - When there are n number of files that exists with a particular location

- **List<FileIdentifier> findIndividualsMedia(Set<PersonIdentity> people, String startDate, String endDate)**
  - Where people do no exists in the data base
  - Where people don't have any media files associated with them
  - When there are a lot of media files associated with them
  - When null is passed as start date and end date
  - When null is passed as end date and a start date is given

- **List<FileIdentifier> findBiologicalFamilyMedia(PersonIdentity person)**
  - Where people don't have any relations defined
  - When no media exists for them
  - When a person doesn't exist in the data base
  - When a person exists and biological family media does not exist
  - When may media are associated with biological family