Rushi Samirbhai Patel RS397441@dal.ca
B00886157

Final Project

Genealogy Media Integration System

Milestone 4

Overview:

For this milestone, I am submitting my partial implementation of GMIS. As discussed in my previous milestone submission, I have designed my database and the code in a similar way which is explained briefly in the below sections:

Schema and Database Implementation:

I have created a schema 'finalproject' in my localhost and have used this schema to create tables which are used in GMIS. For the database, I have created seven tables which are : FamilyTree, MediaArchive, ReferencesOfIndividual, NotesOfIndividual, Children, MediaPeople, and MediaTag. The FamilyTree table stores details of all the individuals added in the system along with their personal details. MediaArchive table stores all the details of a file media. Since a person can have multiple references, ReferencesOfIndivual table would store all the references of the individual. Similarly, NotesOfIndividual table would store all the notes of the individual. I have created the Children table to store and keep track of all the children of a person. For the media database, I have created MediaPeople table to store all the individuals in the media since there can be multiple individuals in a media. Likewise, a media can have multiple tags hence I have created MediaTag table to record all the tags for the media.

Once the schema and design of my database was ready, I connected the database to my java project using JDBC.

Code Implementation:

Once the database was connected, I started my implementation of GMIS as my final project and managed to build the PersonIdentity and FileIdentifier classes. I also completed to manage the family tree and media archive and tested if the data is getting stored in my database as expected. Till now I have not deviated from the plan and logic that I created in my previous milestone.

Rushi Samirbhai Patel RS397441@dal.ca
B00886157

As part of the code design, I have created two interfaces one for each class (FileIdentifier and PersonIdentity class) and implemented all of the methods to record or add data into the system in a class that is implementing the interface.

I have created a Database Util class which deals with the connection of the database with my java code. As discussed in my previous milestone, I have used singleton code design which will make sure that only one instance of connection is created for the code. I have segregated methods for different sql commands which can be globally used in the system.

Files and External Data:

I have used 8 java files till now.

1. DBConnectionUtil.java - The class is used to create instances and connect the java code to the database.
2. FileIdentifier.java - This class is used to identify files which has private fields of media objects.
3. FileIdentifierUtil.java - This class is used for the actual operations to record data into the database
4. Main.java - This class is the main class where we can call all the methods for GMIS.
5. Media.java - This is the interface for the FileIdentifierUtil class.
6. Person.java - This is the interface for the PersonIdentityUtil class.
7. PersonIdentity.java - This class is used to identify people as objects which has private fields of People objects.
8. PersonIdentityUtil.java - This class is used for the actual operations to record data into the database.

There are no external data used for this project as of now.

Data Structures Used:

I have only used Maps and List as abstract data types till now in my code. I have used HashMap and ArrayList as data structures.
HashMaps are used to add attributes of people and media into the database as key value pairs while ArrayList is used to store the people and media as well as person names and media and media file names in the system.

Next step:
Exception class. Genealogy class,Reporting functions and logic implementation, Follow SOLID principles and add proper inline comments.
Next steps include:

Rushi Samirbhai Patel RS397441@dal.ca
B00886157

- Implement Geneology class based on the algorithm and logic mentioned in my previous milestone.
- Implement reporting functions and logic implementation.
- Create a custom exception class to throw exceptions for error conditions.
- Refactor code and apply SOLID principles as much as possible.
- Add proper inline comments for better readability.

SQL Queries to replicate my schema and database:

/*Creating Schema for the project*/
CREATE DATABASE `finalproject` /*!40100 DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci */ /*!80016 DEFAULT ENCRYPTION='N' */;

use finalproject;

/*Creating family tree database*/
create table FamilyTree (id int,individualName varchar(90) not null,dateOfBirth date,locationOfBirth varchar(90), dateOfDeath date, locationOfDeath varchar(90), gender varchar(90), occupation varchar(90), referencesToSourceMaterial varchar(90), notes varchar(90), partner varchar(90), dissolution varchar(90));

/*Creating family tree database*/
create table MediaArchive (id int, fileName varchar(90) not null, dateOfMedia date, location varchar(90), tags varchar(90), individualsInPicture varchar(90));

/*Creating table to map references of individual*/
create table ReferencesOfIndividual (id int, individualName varchar(90), referencesToSourceMaterial varchar(90));

/*Creating table to map notes of individual*/
create table NotesOfIndividual (id int, individualName varchar(90), notes varchar(90));

/*Creating table to record child*/
create table Children (id int, parent varchar(90), child varchar(90));

/*Creating table to record peopleInMedia*/
create table MediaPeople (id int, fileName varchar(90), individualName varchar(90));

/*Creating table to record media tag*/
create table MediaTag (id int, fileName varchar(90), tag varchar(90));