PG FSD implement OOPS using Java with Data Structures and Beyond course capstone project

Ahmed Rakan Aldhafeeri

Title: Command Line File Management System

Introduction:

      The Command Line File Management System is a software developed in Java that allows users to upload, delete and search files from the command line. The system uses various Java concepts and design patterns such as Singleton, Factory and Strategy to provide efficient and reliable file management functionality.

Architecture:

The command line file management system will be developed using java programming language with object-oriented concepts such as inheritance, encapsulation, and polymorphism. The main component of the command-line tool are the following:

- User interface component: This component will handle user input and display the output on the command line interface.
- Business Logic Component: This component will handle user input and display output on the command line interface.
- Data Access Component: This component will handle data storage and retrieval dynamically and securely in such a way it can be local storage or cloud storage like google cloud storage or Amazon S3.

Functional Requirements:
1. Upload files: Users can upload files to the system by specifying the file path and name.
2. Delete files: Users can delete files from the system by specifying the file name.
3. Search files: Users can search for files in the system by specifying a keyword or file name.
4. Navigation: User can close the current execution context and return to the main context.

Non-functional Requirements:
1. Performance: The system should be able to handle large numbers of files efficiently.
2. Security: The system should ensure that only authorized users can access, upload, delete or search for files.
3. Reliability: The system should be reliable and ensure that no data is lost during file operations.
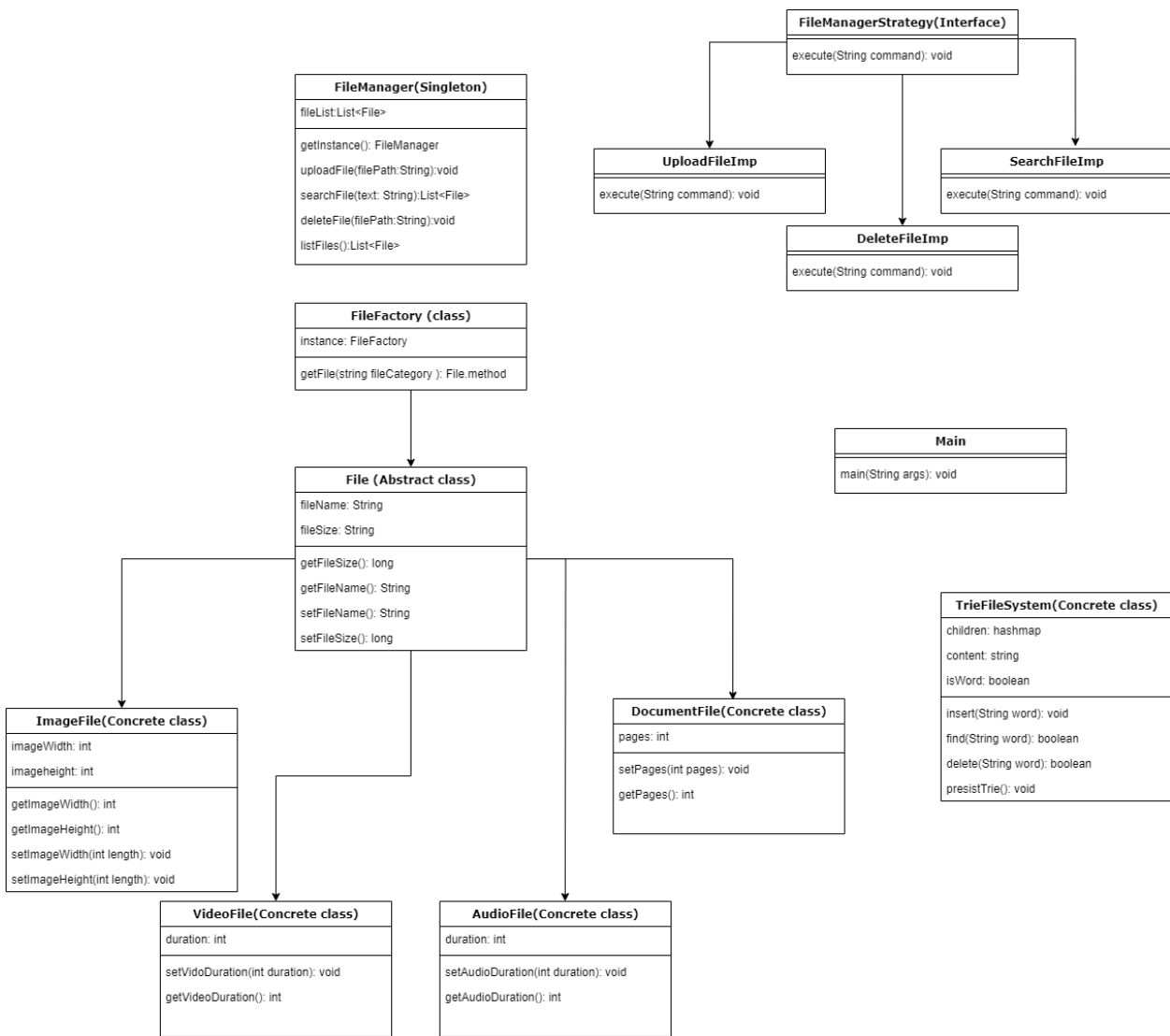
Design Patterns:
1. Singleton Pattern: The Singleton pattern is used to ensure that only one instance of the FileManager class is created throughout the lifetime of the application.
2. Factory Pattern: The Factory pattern is used to create different types of File objects based on user input and file extension, the files are distributed via multiple file categories such as:
   - Image : JPEG, JPG, PNG< GIF, SVG, TIFF, TIF
   - Document: PDF, DOC, DOCX, HTML, HTM, XLS, XLSX, TXT, PTT, PPTX, ODP, KEY
   - Videos: MP4, AVI, MOV, FLV, AVCHD
   - Audio files: M4A, MP3, WAV
3. Strategy Pattern: The Strategy pattern is used to turn the user text input into an invoked command run by the program.

Search Algorithm and Data structure:
   - A trie is a tree-like data structure that stores strings, where each node represents a character in the string.
   - The root node represents an empty string, and each child node represents a character in the string.
   - The path from the root to a leaf node represents a complete string of a stored file.
   - The trie is presented to YMAL file, in which it is used to keep state of the stored files, once program is rerun the trie is uploaded for maximum performance.

Class Diagrams:

FileManagerStrategy(Interface)
execute(String command): void

FileManager(Singleton)
fileList:List<File>

getInstance(): FileManager
uploadFile(filePath:String):void
searchFile(text: String):List<File>
deleteFile(filePath:String):void
listFiles():List<File>

UploadFileImp
execute(String command): void

SearchFileImp
execute(String command): void

DeleteFileImp
execute(String command): void

FileFactory (class)
instance: FileFactory

getFile(string fileCategory): File.method

Main
main(String args): void

File (Abstract class)
fileName: String
fileSize: String

getFileSize(): long
getFileName(): String
setFileName(): String
setFileSize(): long

TrieFileSystem(Concrete class)
children: hashmap
content: string
isWord: boolean

insert(String word): void
find(String word): boolean
delete(String word): boolean
presistTrie(): void

ImageFile(Concrete class)
imageWidth: int
imageheight: int

getImageWidth(): int
getImageHeight(): int
setImageWidth(int length): void
setImageHeight(int length): void

DocumentFile(Concrete class)
pages: int

setPages(int pages): void
getPages(): int

VideoFile(Concrete class)
duration: int

setVidoDuration(int duration): void
getVideoDuration(): int

AudioFile(Concrete class)
duration: int

setAudioDuration(int duration): void
getAudioDuration(): int

Scrum sprints:

ID	Sprint	Tasks	Duration (hours)
1	Design the Command Line File Management System	- Software Design document. - Diagrams	2
2	Project setup and architecture	- Create a project in an IDE. - Setup version control system. - Create a basic file structure for the project.	1
3	Implementation	All implemented classes will go through few cycles of (red-green-refactor) ● Strategy pattern classes ● Factory pattern classes ● FileManagerClass ● FileClass ● Main class	5-10
4	Integration testing	Test the entire application	**1-3**
5	Acceptace testing	Manual testing of the application	**1-3**