package com.project.lockedme;

import java.io.IOException;

import java.util.Scanner;

public class Menus {

Scanner scan = new Scanner(System.in);

AllOperations dao = new AllOperations();

public void introScreen() {

System.out.println();

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

System.out.println("\* DEVELOPED BY VINEET MULIK \*");

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

System.out.println("\* LOCKEDME.COM \*");

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

System.out.println("\* Directory: " + Main.path +" \*");

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

System.out.println("\n\n");

}

public void exitScreen() {

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

System.out.println("\* \*");

System.out.println("\* THANK YOU FOR VISITING LOCKEDME.COM \*");

System.out.println("\* \*");

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

System.out.println("\n\n");

}

public void mainMenuOptions() {

System.out.println("=====================================");

System.out.println("| MAIN MENU |");

System.out.println("=====================================");

System.out.println("| Select any one of the following: |");

System.out.println("| 1 - List All Files |");

System.out.println("| 2 - More Options |");

System.out.println("| 3 - Exit |");

System.out.println("=====================================");

System.out.println("Enter your choice : ");

}

public void subMenuOptions() {

System.out.println("=====================================");

System.out.println("| SUB MENU |");

System.out.println("=====================================");

System.out.println("| Select any one of the following: |");

System.out.println("| 1 - Add a file |");

System.out.println("| 2 - Delete a file |");

System.out.println("| 3 - Search a file |");

System.out.println("| 4 - Go Back |");

System.out.println("=====================================");

System.out.println("Enter your choice : ");

}

public void mainMenu() {

int choice = 0;

char decision = 0;

do {

mainMenuOptions();

try {

choice = Integer.parseInt(scan.nextLine());

} catch (NumberFormatException e) {

System.out.println("\nInvalid Input \nValid Input Integers:(1-3)\n");

mainMenu();

}

switch (choice) {

case 1:

System.out.println();

try {

dao.listAllFiles(Main.path);

}catch(NullPointerException e) {

System.out.println(e.getMessage());

}catch(IllegalArgumentException e) {

System.out.println(e.getMessage());

}catch(Exception e) {

System.out.println(e.getMessage());

}

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

break;

case 2:

System.out.println();

subMenu();

break;

case 3:

System.out.println("\n Are you sure you want to exit ? ");

System.out.println(" (Y) ==> Yes (N) ==> No ");

decision = scan.nextLine().toUpperCase().charAt(0);

if(decision == 'Y') {

System.out.println("\n");

exitScreen();

System.exit(1);

}else if(decision == 'N') {

System.out.println("\n");

mainMenu();

}else {

System.out.println("\nInvalid Input \nValid Inputs :(Y/N)\n");

mainMenu();

}

default:

System.out.println("\nInvalid Input \nValid Input Integers:(1-3)\n");

mainMenu();

}

}while(true);

}

public void subMenu() {

String file = null;

String fileName = null;

int choice = 0;

do {

subMenuOptions();

try {

choice = Integer.parseInt(scan.nextLine());

} catch (NumberFormatException e) {

System.out.println("Invalid Input \nValid Input Integers:(1-4)");

subMenu();

}

switch (choice) {

case 1:

System.out.println("\n==> Adding a File...");

System.out.println("Please enter a file name : ");

file = scan.nextLine();

fileName = file.trim();

try {

dao.createNewFile(Main.path, fileName);

}catch(NullPointerException e) {

System.out.println(e.getMessage());

}catch(IOException e) {

System.out.println("Error occurred while adding file..");

System.out.println("Please try again...");

}catch(Exception e) {

System.out.println("Error occurred while adding file..");

System.out.println("Please try again...");

}

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\n");

break;

case 2:

System.out.println("\n==> Deleting a File...");

System.out.println("Please enter a file name to Delete : ");

file = scan.nextLine();

fileName = file.trim();

try {

dao.deleteFile(Main.path, fileName);

}catch(NullPointerException e) {

System.out.println(e.getMessage());

}catch(IOException e) {

System.out.println("Error occurred while Deleting File..");

System.out.println("Please try again...");

}catch(Exception e) {

System.out.println("Error occurred while Deleting File..");

System.out.println("Please try again...");

}

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

break;

case 3:

System.out.println("\n==> Searching a File...");

System.out.println("Please enter a file name to Search : ");

file = scan.nextLine();

fileName = file.trim();

try {

dao.searchFile(Main.path, fileName);

}catch(NullPointerException e) {

System.out.println(e.getMessage());

}catch(IllegalArgumentException e) {

System.out.println(e.getMessage());

}catch(Exception e) {

System.out.println(e.getMessage());

}

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

break;

case 4: mainMenu();

break;

default:

System.out.println("Invalid Input \nValid Input Integers:(1-4)");

subMenu();

}

file = null;

fileName = null;

}while(true);

}

}

**package** com.project.lockedme;

**import** java.io.File;

**import** java.io.IOException;

**import** java.util.Arrays;

**import** java.util.Set;

**import** java.util.TreeSet;

**import** java.util.regex.Matcher;

**import** java.util.regex.Pattern;

**public** **class** AllOperations {

**public** **void** listAllFiles(String path) {

**if** (path == **null** || path.isEmpty())

**throw** **new** NullPointerException("Path cannot be Empty or null");

File dir = **new** File(path);

**if**(!dir.exists())

**throw** **new** IllegalArgumentException("Path does not exist");

**if**(dir.isFile())

**throw** **new** IllegalArgumentException("The given path is a file. A directory is expected.");

String [] files = dir.list();

System.***out***.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*");

**if**(files != **null** && files.length > 0) {

Set<String>filesList = **new** TreeSet<String>(Arrays.*asList*(files));

System.***out***.println("The Files in "+ dir.getAbsolutePath() + " are: \n");

**for**(String file1:filesList) {

System.***out***.println(file1);

}

System.***out***.println("\nTotal Number of files: "+ filesList.size());

}**else** {

System.***out***.println("Directory is Empty");

}

}

**public** **void** createNewFile(String path , String fileName) **throws** IOException {

**if** (path == **null** || path.isEmpty())

**throw** **new** NullPointerException("Path cannot be Empty or null");

**if** (fileName == **null** || fileName.isEmpty())

**throw** **new** NullPointerException("File Name cannot be Empty or null");

File newFile = **new** File(path + File.***separator*** + fileName);

**boolean** createFile = newFile.createNewFile();

**if** (createFile) {

System.***out***.println("\nFile Successfully Created: " + newFile.getAbsolutePath());

}**else** **if**(!createFile) {

System.***out***.println("\nFile Already Exist.. Please try again." );

}

}

**public** **void** deleteFile(String path , String fileName) **throws** IOException {

**if** (path == **null** || path.isEmpty())

**throw** **new** NullPointerException("Path cannot be Empty or null");

**if** (fileName == **null** || fileName.isEmpty())

**throw** **new** NullPointerException("File Name cannot be Empty or null");

File newFile = **new** File(path + File.***separator*** + fileName);

**boolean** deleteFile = newFile.delete();

**if** (deleteFile) {

System.***out***.println("\nFile deleted Successfully");

}**else** {

System.***out***.println("\nFile Not Found.. Please try again." );

}

}

**public** **void** searchFile(String path , String fileName){

**if** (path == **null** || path.isEmpty())

**throw** **new** NullPointerException("Path cannot be Empty or null");

**if** (fileName == **null** || fileName.isEmpty())

**throw** **new** NullPointerException("File Name cannot be Empty or null");

File dir = **new** File(path);

**if**(!dir.exists())

**throw** **new** IllegalArgumentException("Path does not exist");

**if**(dir.isFile())

**throw** **new** IllegalArgumentException("The given path is a file. A directory is expected.");

String [] fileList = dir.list();

**boolean** flag = **false**;

Pattern pat = Pattern.*compile*(fileName);

**if**(fileList != **null** && fileList.length > 0) {

**for**(String file:fileList) {

Matcher mat = pat.matcher(file);

**if**(mat.matches()) {

System.***out***.println("File Found at location: " + dir.getAbsolutePath());

flag = **true**;

**break**;

}

}

}

**if**(flag == **false**)

System.***out***.println("File Not Found.. Please try again.");

}

}