Advanced Programming 2016/17 Project Report
Title: Movie Management
Student: Elisa Vogl
StudentID: 14721

1. Description

The application helps managing a movie library. The user can add,update and delete movies, search them or import data from other files.
The application can only be used if the person has an user account and the useraccount can only be created by the admin or by a person which is already registered.
Additionally, the user can delete his own user account with the password and username.
The data is persistently stored in an DAT file for the users as well as for the movies. Every user registered has access to all the functions. As soon as somebody has done a function, the user can see the changed movie collection.

2. Utilization of the programming techniques

I used the seven programming techniques as follows.

Technique	Description
1. Junit Test Cases and Test Suite	**UserReaderTest.java:** Line 31-45: I created tests to check if the created user in the file reads it correctly in an ArrayList of username and password Line 47-59: I created a test to check is the user has been added successfully to the User file Line 61-78: I created tests to check is the username has been found successfully in the file or not. Line 82-99: I checked if username and password has been found together successfully in one line and checked also the opposite. Lines 101-116:I created tests to check if a user has entered the correct username credentials for the registration. Lines 119-132:I created tests to check if the user has entered the correct password credentials for the registration. Lines 134-140:I created test to check if the encryption of the password goes well and is encrypted successfully. **MovieReaderTest.java:** Line 31-44: : I created tests to check if the movies from the file are read successfully in an ArrayList of movies in title, genre, year, director and oscar Line 46-61: I created tests to check if a movie will be found succesfully in a file and the opposite also. Line 65-85: I created tests to check if the movie has been created successfully in the file and arraylist **AllTests.java** TestSuite for all JUnitTests
2. Logging	I created for the different classes different Logging Files, this means in case there would be an error of Level.SEVERE it would write the logging in the class file where the error has occurred. The creation of the Logging files can be found in every class where a file is needed. The most important things like FileNotFoundException or IOException for the files of Users and

	Data, have the level SEVERE because in this case it would be unable to start the application or doing the things which have to be done. The Level.INFO has been created in case the user did not enter correct Information in the textfields or something else which refers to the user. **LogInPanel.java** line 111: Logging for EmptyFieldExcecption line 140: Logging for EmptyFieldException **MovieReader.java:** **Method boolean findinFile (line 86)** Line 125: Logging FileNotFoundException Line 128: Logging IOException Line 136: Logging IOException for closing reader Line 139: Logging for IO Exception that reader is not open **Method public void write (line118)** Line 159: Logging IOException Line 165: Logging Printwriter not open **Method void add (line 141)** Line 185: Logging IOException Line 193: Finally Logging for IOException Line 196: Logging for finally closing PrintWriter **MenuPanel.java:** Line 226: Logging for RegexNumberException Line 236: Logging for EmptyFieldException Line 326: Logging RegexNumerException Line 335: Logging EmptyFieldException Line 532: Logging for FileNotFoundException LIne 535: Logging for NumberFormatException Line 542: Logging for IOException Line 550: Logging for ArrayIndexOutOfBoundsException **RegisterPanel.java:** **ImageIO.read** Line 108: Logging IOException Line 165: Logging for NoImageException Line 184: Logging for IOException Line 190: Logging in HeadlessException Line 193: EmptyFieldException Logging **ReaderWriterFactory.java** Line 48: Logging for IOException for getReaderWriterInterface Line 55: Logging for NumberFormatException Line 57: Logging IOException **UserReader.java:** **Method void add (line 103)** Line 113: Logging in IOException Line 119: Logging for IOException in Finally Line 122: Logging for Finally **Method boolean findinFile (line 135)** Line 152: Logging in FileNotFoundException Line 154: Logging in IOException **Method boolean findUsernamePassword (line 173)** Line 188:Logging in FileNotFoundException Line 190: Logging in IOException

	Line 195: Logging in IOException in Finally **Method String encrypt (line 209)** Line 223: Logging in NoSuchAlgorithmException **Method write(line 275)** line 285: Logging for IOException Line 291: Logging for finally **ProfilePanel.java** **Method ImageIcon uploadImage(line 115)** Line 140: Logging IO Exception
3. Exception throwing	I have created for all the files which have to be read or written exceptions that the program of Movie Collection will not close during the execution. Additionally, I have created three custom exceptions which are described below **LogInPanel.java** line 106-112: try and catch for EmptyFieldException Line 137-143: EmptyFieldException with try and catch **MenuPanel.java** Line 74-86: try and catch for Logging File creation Line 322-327: try and catch for RegexNumberException line 332-339: throws new EmptyFieldException between try and catch line 323: throw new RegexNumberException line 318-322: try and catch for RegexNumberException line 502-551: Try and catch for importing file that the application will not terminate, JOptionPane for different situations and loggings **MovieReader.java** line 69: throws NumberFormatException, IO Exception line 110-140: try and catch and finally for different Exceptions and checking if the file has already been closed or is closed now line 151-167: try and catch + finally for checking if writer is already closed or is closed now line 178-197: try and catch + finally for checking if writer is already closed or now **ReaderWriterFactory.java** line 17-28: try and catch for the Logging File line 45-49 try and catch for taking the UserReader in the Factory Pattern line 52-58: try and catch for taking the MovieReader in the FactoryPattern line 102-109: try and catch for IOException if the image cannot be read line 162-169: NoImageException if the picture is null, try and catch Line 181-182: catching IOException for writing the image Line 187-189: catching HeadlessException Line 190-195: catching EmptyFieldException throwing if fields are empty **RegisterPanel.java** line 52: throwing IOException for taking object of UserReader line 53-64: try and catch for creating file Line 102-109 try and catch for reading the image Line 159-169 try and catch for NoImageException Line 183-197 catch for saving the image **UserReader.java**

	Line 39-50: try and catch for creating Logging File Line 61: throws IOException Line 106-´125 : try and catch for adding a user to the file and finally for checking if printwriter is open or no Line 138-155: Try and catch for finding user in the file Line 176-196: try and catch if username and password has been found line 212-223: try and catch for encrypting the password line 277-294: try and catch for writing the ArrayList to the file and finally for checking if file closed **RegexNumberException.java** Own created exception for testing that oscartext and yeartext have a number and if not there is thrown the exception and the information of JOptionPane is showns MenuPanel.java Line 322-327: try and catch, Line 323 Line 322-327: try and catch for RegexNumberException **NoImageException.java** Own created exception for throwing if the user has not uploaded any image when registering **RegisterPanel.java** Line 175: Own created exception throwing if user has forgotten to upload an picture there will be thrown the NoImage Exception and the user will be remembered to upload a picture **EmptyFieldException.java** created for checking if the textfields are empty DeletePanel.java line 118 throws new EmptyFieldException LogInPanel.java line 102 throws new EmptyFieldException between try and catch MenuPanel.java line 232 throws new EmptyFieldException with try and catch
4. data I/O (files/JDBC/web)	I have used **BufferedReader together with FileReader and PrintWriter together with BufferedWriter and FileWriter**. **MovieReader.java** Line 69-90: Method for reading from the movie file and add the contents to Arraylist (BufferedReader together with FileReader) Line 107-143: Method for reading and checking in the movie file if the String exists (BufferedReader together with FileReader) Line 148-169: Method for writing the whole ArrayList to the file (PrintWriter,BufferedWriter,FileWriter) Line 175-199: Method for writing a movie from ArrayList to the file (PrintWriter,BufferedWriter, FileWriter) **UserReader.java** Line 61-81: Method for reading from the users file and adding the contents to ArrayList User (BufferedReader, FileReader) Line 103-125: Method for adding a user to the file (PrintWriter,BufferedWriter,FileWriter) Line 135-158: Method for finding a String in File and therefore reading it (BufferedReader,FileReader) Line 173-199: Method for finding the whole line in a File (BufferedReader,FileReader) Line 275-292: Writing with PrintWriter the whole ArrayList to the file
5. Generics and Collections	I have used an **ArrayList** for all the users registered at the

	MovieCollection as well as an **ArrayList** for the movies which are saved in the file. Moreover, there is an interface for the different ArrayLists to write the whole ArrayList in the file (method can be found in UserReader as well as MovieReader), update the ArrayList which is found in the MenuPanel.java, delete a movie or user from the ArrayList which is found also in the MenuPanel.java and DeletePanel.java and additionally, adding a new user or movie to the ArrayList which can be found in the class MenuPanel.java and DeletePanel.java. All methods are without any cast.
6. JavaDoc documentation	I have created a Javadoc documentation which is saved in the folder log. Above mentioned I have described the classes and methods including the parameters and return. **LogInPanel.java** description for the class description for the method removeUser(String username) with the parameter **MainFrame.java** description for the class description for changeToDelete() description for changeToRegister() description for changeToProfile() description for changeToLogin () description for changetoMenu() description for createMenu() **MenuPanel.java** description for the class description for the method search() description for the method InsertFileDatatoJTable description for the method addRowforMovie() **MovieReader.java** description for the class description for getList () description for the Method ArrayList<Movies>readFile() description for method findinFile (String find) description for method write() description for method add() description for method openMovies() **Movies.java** description for the class description for method getYear() description for the method getOscar() description for the method getTitle() description for the method getGenre() description for the method getDirector() description for the method String toString() description for the public static class Builder creates a new Movie with all the necessary parameters **ProfilePanel.java** description for the class description for the method uploadImage ()

	description for the method changeUserName and for the parameters String name description for the method resize() with explanation of all the parameters which are included **ReaderWriterFactory.java** description for the class explains the class getReaderWriterInterface with their parameter and the return method **RegisterPanel.java** description for the class description for the method addnewUser with the parameters explained **RegisterPanelTest.java** description for the class **User.java** description for the class description for the method getUsername() description for the method getPassword() description for the method String toString () **UserReader.java** description for the class description for the method readFile() and what is returned description for the method getList() description for the method add() and their parameter of the Writer print description of the method findinFile with their parameters of the String searchedString description of the method findUsernamePassword with all the parameters which are needed (password and username) as well as description of the return description of the method encrypt with the parameter of the unencrypted String and what should be returned description of the method checkUsername with the parameter and the return method description of the method checkPassword with the paramenter of the CharSequence password and what should be returned description for the method openUsers with the return and also what exception is thrown description for the method write()
7. Design Patterns	**I used the BuilderPattern in the following class:** **Movies.java** created a public static Builder class for taking the object for the other classes and creating the object of the movie much easier by the method createMovies **I used the SingletonPattern in the following class:** **MainFrame.java** I have created a object in the class to call it from different classes and always using the same new Object for not instantiating in every class another new Object. Used in the following classes for calling the method: **LogInPanel.java** **MenuPanel.java**

	ProfilePanel.java
	RegisterPanel.java
	
	I used the Factory Pattern for the classes UserReader and MovieReader:
	For taking the FactoryPattern I created the class ReaderWriterInterface for implementing the interface in the class UserReader as well as MovieReader. Additionally, the ReaderWriterFactory class for comparing both classes and taking the method which is needed for the different classes.
	Following methods are used for the interface:
	`public boolean findinFile(String linetoremove);`
	`public void add (Writer writer);`
	`public <E> ArrayList<E> getList();`
	`public void write();`
	Used in the following classes:
	LoginPanel.java
	MenuPanel.java
	RegisterPanel.java