Cinema Application

Analysis and Design Document

Student: Deac Dan

**Group: 30233**

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| <5/04/2017> | <1.0> |  | Deac Dan |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

I. Project Specification 4

II. Elaboration – Iteration 1.1 4

1. Domain Model 4

2. Architectural Design 4

2.1 Conceptual Architecture 4

2.2 Package Design 4

2.3 Component and Deployment Diagrams 4

III. Elaboration – Iteration 1.2 4

1. Design Model 4

1.1 Dynamic Behavior 4

1.2 Class Design 4

2. Data Model 4

3. Unit Testing 4

IV. Elaboration – Iteration 2 4

1. Architectural Design Refinement 4

2. Design Model Refinement 4

V. Construction and Transition 5

1. System Testing 5

2. Future improvements 5

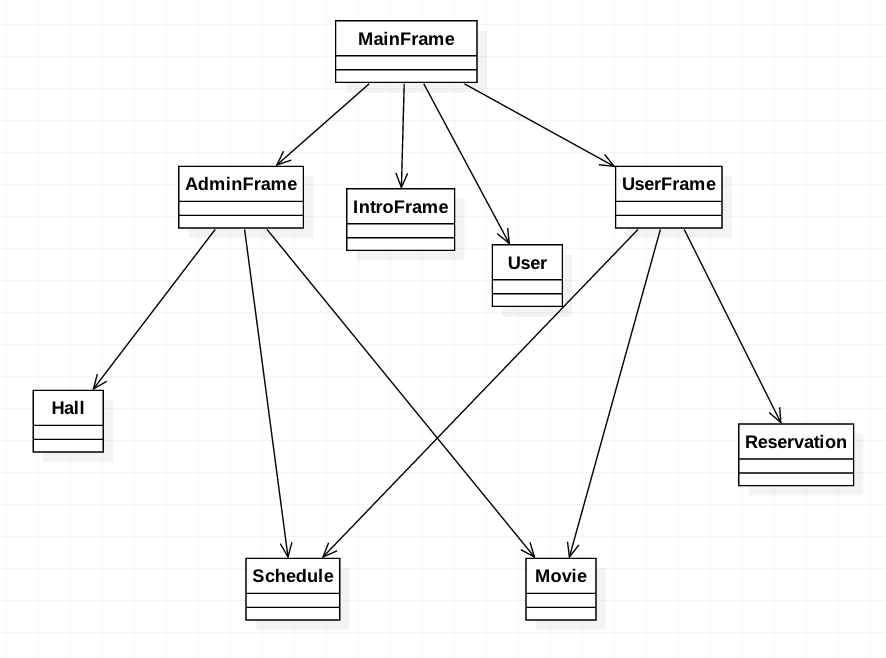
VI. Bibliography 5

# Project Specification

My project will be about a cinema application. There will be two main actors: the administrator and the regular user. The administrator will be able to make CRUD operations on movies, halls and schedule. The regular user could be any movie enthusiastic. He will be able to login, register, watch movie details and make reservations.

# Elaboration – Iteration 1.1

# Domain Model



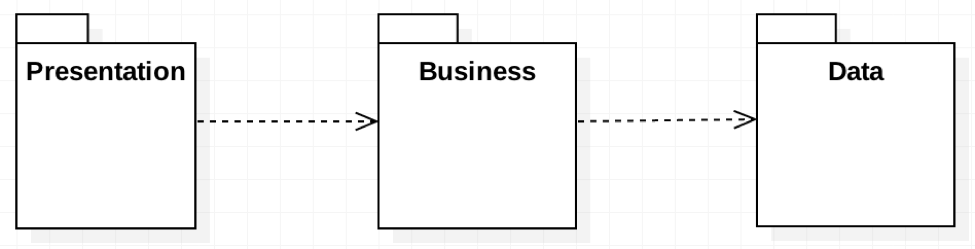
# Architectural Design

## Conceptual Architecture

The architectural pattern of my application will be the Layered Architectural Pattern. The Layered Architectural Pattern will be divided into three layers: Presentation, Business and Data Source. The Presentation Layer will contain the GUI, the Business Layer will be the linker layer between Presentation and Data Source. Here the information will be transformed and passed forward to another layer. The Data Source Layer will be the part of the project that will be directly linked to the database and will execute queries.

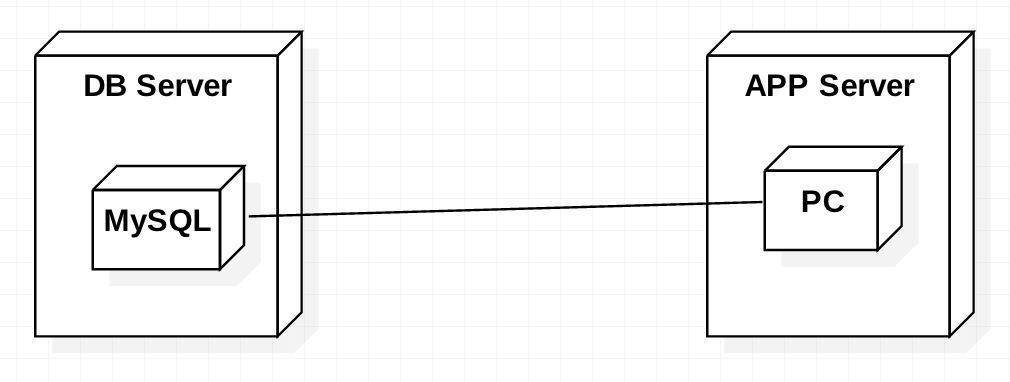
## Package Design

The package diagram:

**

## Component and Deployment Diagrams

The deployment diagram:



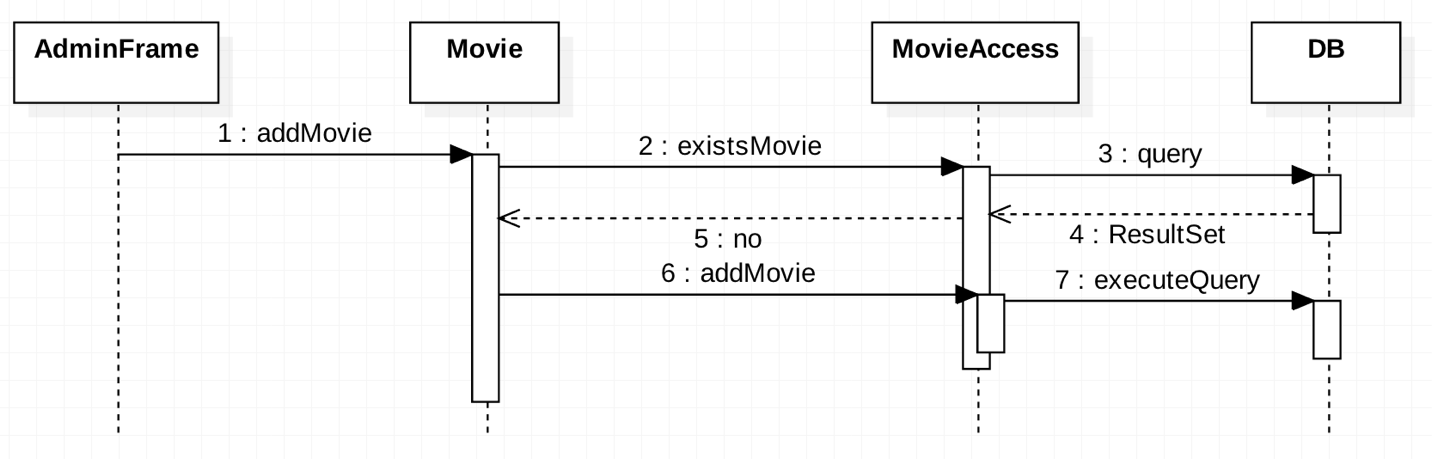
# 

# Elaboration – Iteration 1.2

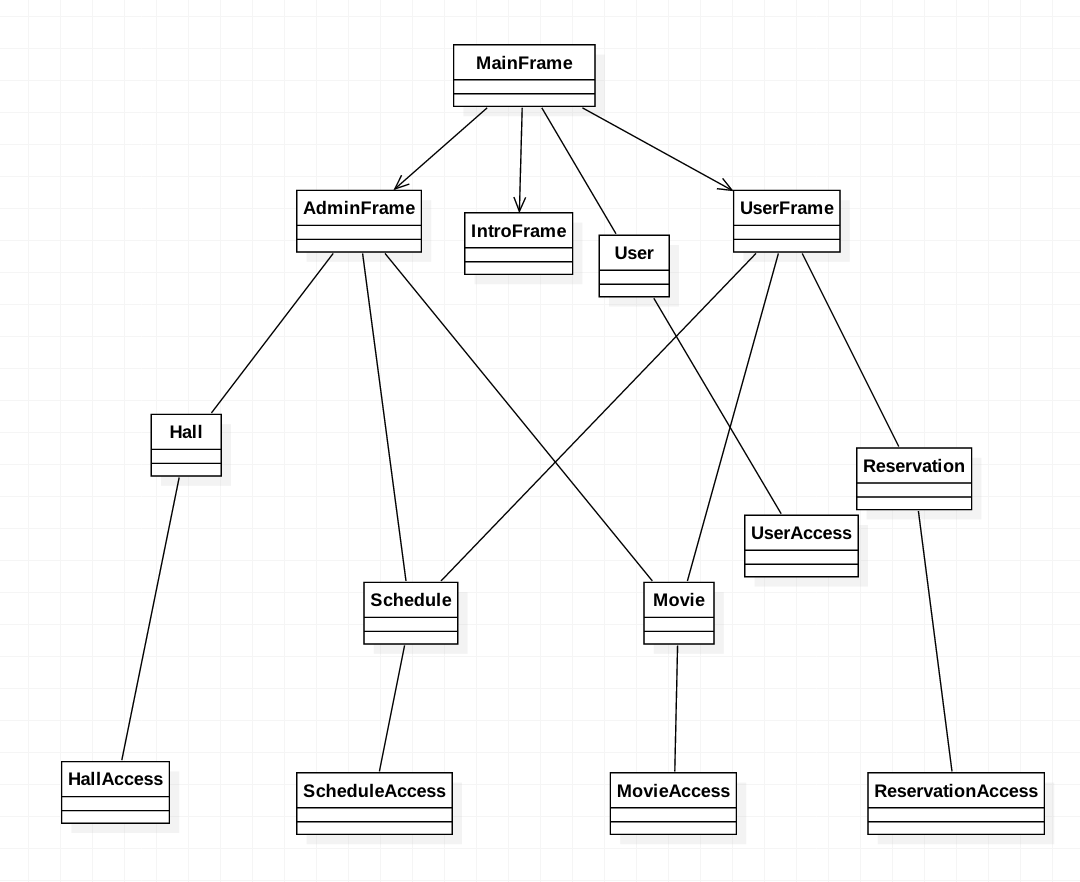
# Design Model

## Dynamic Behavior

The sequence diagram for the "add a movie" scenario is presented in the figure below:



## Class Design



# Data Model

The data model will be composed of 5 tables into the database. There will be a table for the movies that will contain information like movieID, name, director, imdb rank and a link to the image. The halls table will contain information like hallID and capacity. The user table will contain information like username, password and email. The schedule table will contain information like scheduleID, movie name, hallID, left capacity and hours. The reservation table will contain information like id, scheduleID, userID and number of places.

# Elaboration – Iteration 2

# Architectural Design Refinement

The architectural pattern that I've used is Layered Architectural Pattern. It is divided into three main layers: Presentation (user interface), Business (the transition layer between Presentation and Data layer) and the Data layer that executes queries over the database. As design patterns I chose the Filter Pattern and the Transfer Object Pattern.

Filter pattern or Criteria pattern is a design pattern that enables developers to filter a set of objects using different criteria and chaining them in a decoupled way through logical operations. This type of design pattern comes under structural pattern as this pattern combines multiple criteria to obtain single criteria. I have used this pattern to display different ResultSets as a table.

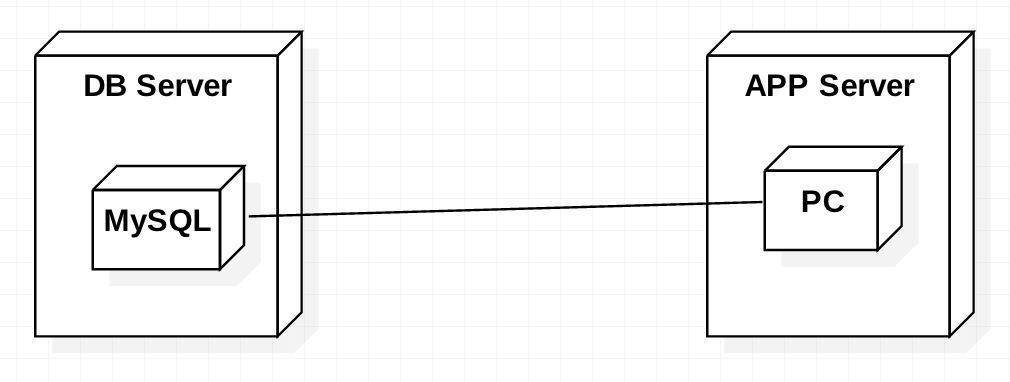
The Transfer Object pattern is used when we want to pass data with multiple attributes in one shot. Transfer object is also known as Value Object.

Following are the entities of this type of design pattern.

* Business Object - Business Service fills the Transfer Object with data.
* Transfer Object - Simple POJO having methods to set/get attributes only.
* Client - Client either requests or sends the Transfer Object to Business Object.

This pattern was used for the email sending feature.

The deployment diagram:



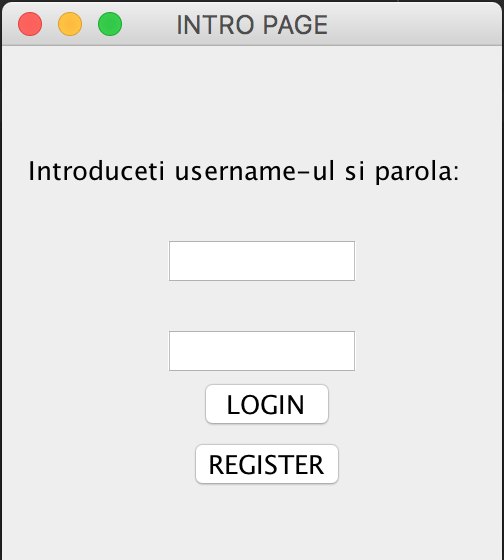
# Design Model Refinement

## 

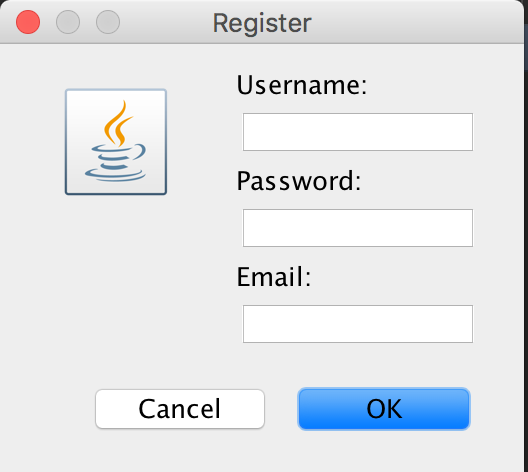
# Construction and Transition

# System Testing

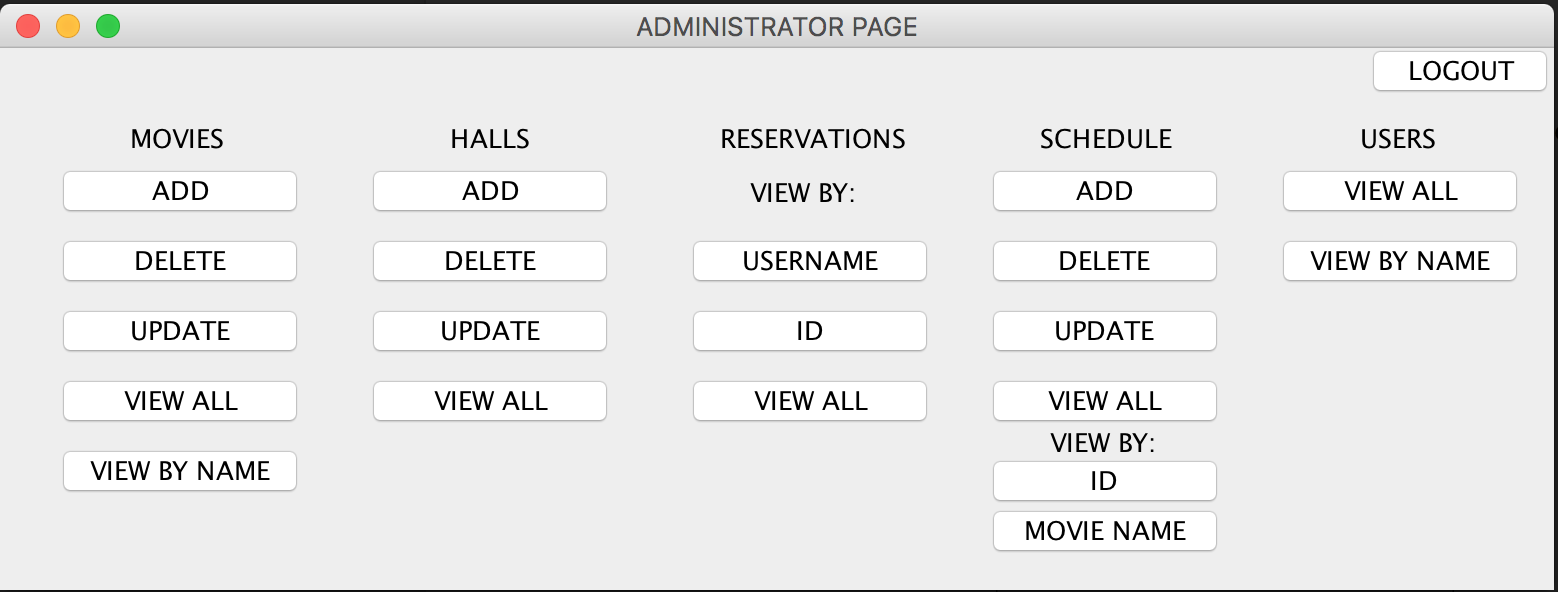
After we run the application the login page will pop-up:



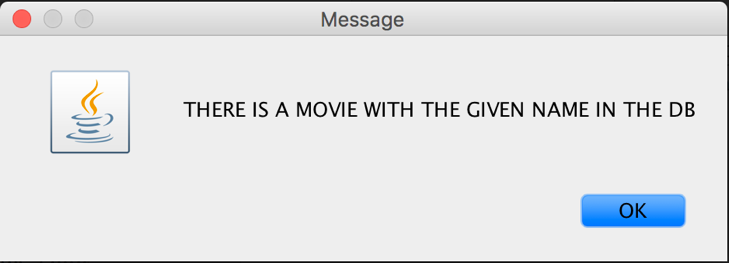
The user types his credentials and depending on his type the admin page or the employee page will pop-up. If the username or the password are wrong a message dialog will be showed. If the user doesn't have an account he can create one by clicking the REGISTER button:



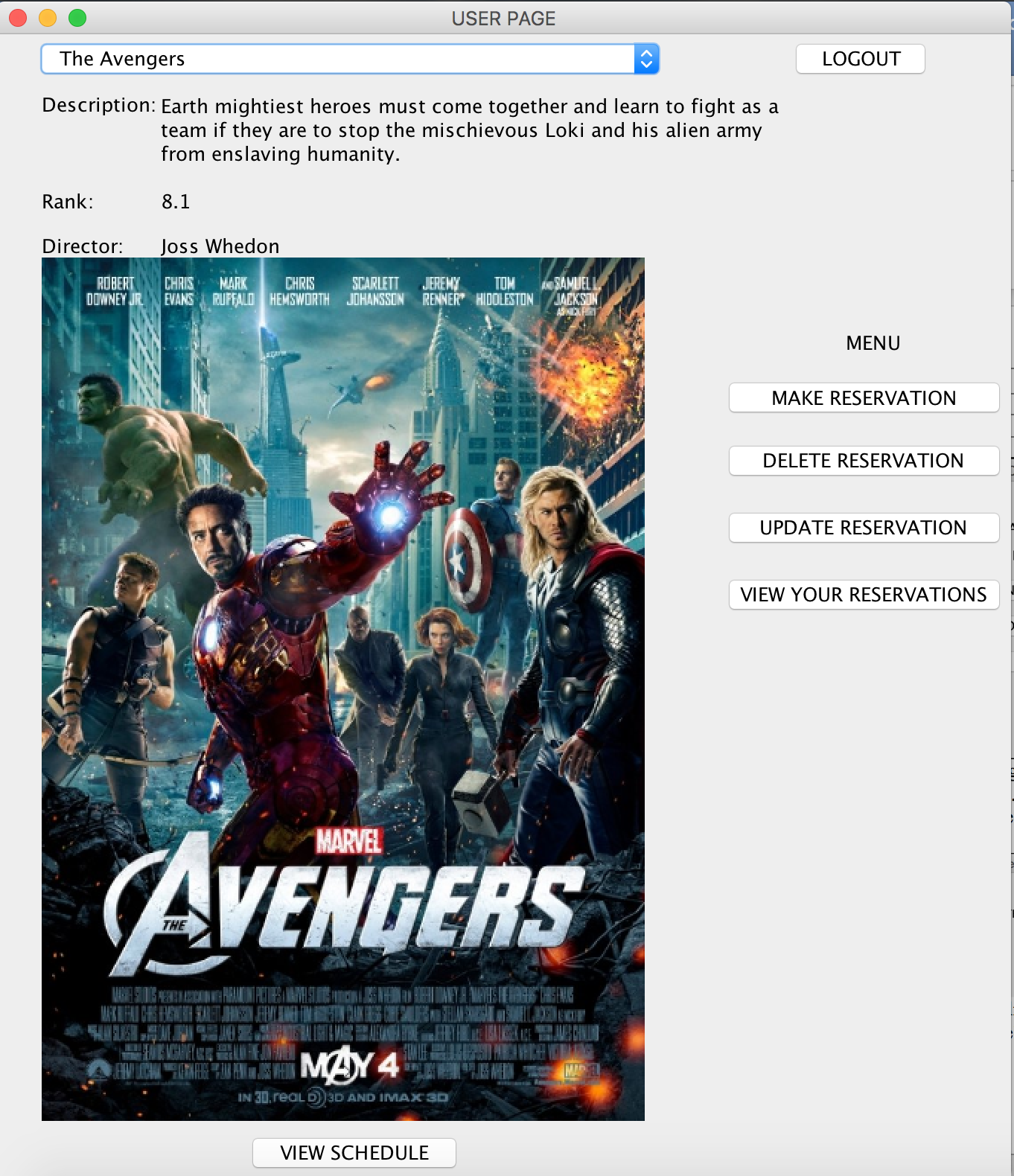
The administrator could do CRUD operations on movies, halls and schedule and view information about reservations and users.



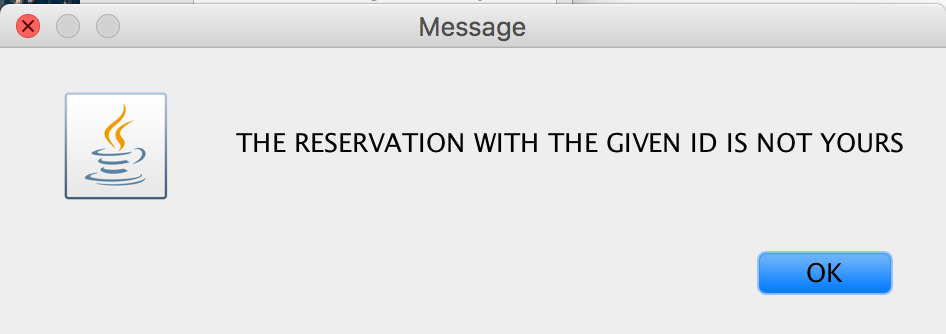
When adding a new movie an error message will be showed if the administrator tries to add a movie that already is in database (same title). Error messages will be showed as well when the administrator will try to add halls and schedule or will try to update or remove entries which are not existing in the database.



For the regular users the page will look like in the picture below, and they can view details about movies, view the schedule for the selected movie and make CRUD operations on reservations.



If a user tries to update or delete a reservation that does not belong to him an error message will be showed:



When the user makes a new reservation, updates or deletes an existent reservation he will be notified by email that is associated to the account.



# Future improvements

As future improvements I could modify the reservation process, so that the user can choose the seats, watch the movie trailer and choose the type of ticket he wants to reserve (normal, students, pupils) and the date. Also the application could be improved by extending the list of cinemas so that the user can choose the closest one.

# Bibliography

1. Patterns of Enterprise Application Architecture, By Martin Fowler, David Rice, Matthew Foemmel, Edward Hieatt, Robert Mee, Randy Stafford
2. Software Architecture Patterns, By Mark Richards
3. <https://www.codeproject.com/Articles/654670/Layered-Application-Design-Pattern>
4. <https://dev.mysql.com/doc/refman/5.7/en/tutorial.html>
5. <https://www.tutorialspoint.com/design_pattern/filter_pattern.htm>
6. <https://www.tutorialspoint.com/design_pattern/transfer_object_pattern.htm>
7. <https://www.mkyong.com/java/javamail-api-sending-email-via-gmail-smtp-example/>