

# **MOVIEGO WEB APP**

---

*ANALYSIS AND DESIGN DOCUMENT*

**Group 6**  
**Divina Gorospe | Enrique Paredes | Pawel Zajac**

## TABLE OF CONTENTS

<b>Table of Figures.....</b>	<b>6</b>
<b>1. Introduction.....</b>	<b>7</b>
1.1 Purpose .....	7
1.2 Scope .....	7
1.3 Overview/References.....	8
1.4 Definitions and Acronyms:.....	9
<b>2. System Architecture.....</b>	<b>10</b>
2.1 System Overview.....	10
2.2 Architectural Design.....	10
2.3 Decomposition Description .....	11
2.4 Design Rationale.....	12
<b>3. Human Interface Design.....</b>	<b>14</b>
3.1 Account.....	14
A. Registration .....	14
A1. Overview of User Interface.....	14
A2. Screen Images .....	14
A3. Screen Objects and Actions: .....	14
A4. Input and Output .....	14
A5. Actions .....	15
A6. Pre and Post Conditions .....	15
A7. Validation.....	15
A8. Different actors cases need to be included (client and administrator) .....	16
B. Login.....	17
B1. Overview Of User Interface .....	17
B2. Screen Images.....	17
B3. Screen Objects and Actions: .....	17
B4. Input and Output.....	17
B5. Actions .....	17
B6. Pre and Post Conditions .....	18
B7. Validation.....	18
B8. Different actors cases need to be included (client and administrator) .....	18
C. Change Password .....	19
C1. Overview Of User Interface .....	19
C2. Screen Images.....	19
C3. Screen Objects And Actions .....	19
C4. Input And Output .....	19
C5. Actions .....	19
C6. Pre and Post Conditions .....	20
C7. Validation.....	20
C8. Different actors cases need to be included (client and administrator) .....	20
3.2 Movie Facts .....	21
A. Select .....	21
A1. Overview of User Interface.....	21
A2. Screen Images .....	21

A3. Screen Objects and Actions .....	21
A4. Input and Output .....	21
A5. Actions .....	22
A6. Pre and Post Conditions .....	22
A7. Validation .....	22
A8. Different actors cases need to be included (client and administrator) .....	22
B. Sort .....	23
B1. Overview of User Interface .....	23
B3. Screen Objects and Actions: .....	23
B4. Input and Output .....	23
B5. Actions .....	24
B6. Pre and Post Conditions .....	24
B7. Validation .....	24
B8. Different actors cases need to be included (client and administrator) .....	24
C. Rate .....	25
C1. Overview of User Interface .....	25
C2. Screen Images .....	25
C3. Screen Objects and Actions: .....	25
C4. Input and Output .....	26
C5. Actions .....	26
C6. Pre and Post Conditions .....	26
C7. Validation .....	26
C8. Different actors cases need to be included (client and administrator) .....	27
3.3 Search .....	28
A. Create .....	28
A1. Overview of User Interface .....	28
A2. Screen Images .....	28
A3. Screen Objects and Actions: .....	28
A4. Input and Output .....	28
A5. Actions .....	28
A6. Pre and Post Conditions .....	28
A7. Validation .....	28
A8. Different actors cases need to be included (client and administrator) .....	29
B. Modify .....	30
B1. Overview of User Interface .....	30
B2. Screen Images .....	30
B3. Screen Objects and Actions: .....	30
B4. Input and Output .....	30
B5. Actions .....	30
B6. Pre and Post Conditions .....	30
B7. Validation .....	30
B8. Different actors cases need to be included (client and administrator) .....	31
C. Input .....	31
C1. Overview of User Interface .....	31
C2. Screen Images .....	31
C3. Screen Objects and Actions: .....	31

C4. Input and Output.....	31
C5. Actions .....	31
C6. Pre and Post Conditions .....	31
C7. Validation.....	31
C8. Different actors cases need to be included (client and administrator) .....	32
D. Select.....	32
D1. Overview of User Interface.....	32
D2. Screen Images .....	32
D3. Screen Objects and Actions: .....	32
D4. Input and Output .....	32
D5. Actions .....	32
D6. Pre and Post Conditions .....	32
D7. Validation.....	33
D8. Different actors cases need to be included (client and administrator) .....	33
3.4 Discussion Board.....	34
A. Post.....	35
A1. Overview of User Interface.....	35
A2. Screen Images .....	35
A3. Screen Objects and Actions: .....	35
A4. Input and Output .....	35
A5. Actions .....	36
A6. Pre and Post Conditions .....	36
A7. Validation.....	36
A8. Different actors cases need to be included (client and administrator) .....	36
B. Reply.....	37
B1. Overview of User Interface .....	37
B2. Screen Images .....	37
B3. Screen Objects and Actions: .....	37
B4. Input and Output.....	37
B5. Actions .....	38
B6. Pre and Post Conditions .....	38
B7. Validation .....	38
B8. Different actors cases need to be included (client and administrator) .....	38
C. Validate.....	39
C1. Overview of User Interface .....	39
C2. Screen Images .....	39
C3. Screen Objects and Actions: .....	39
C4. Input and Output.....	39
C5. Actions .....	40
C6. Pre and Post Conditions .....	40
C7. Validation.....	40
C8. Different actors cases need to be included (client and administrator) .....	40
<b>4. Component Design.....</b>	<b>41</b>
4.1 Account.....	41
A. UML Static Structure Class Diagram.....	41
B. Entities .....	41

4.2 Movie Facts .....	43
A. UML Static Structure Class Diagram.....	43
B. Entities .....	43
4.3 Search .....	45
A. UML Static Structure Class Diagram.....	45
B. Entities .....	45
4.4 Discussion Board.....	46
A. UML Static Structure Class Diagram.....	46
B. Entities .....	46
<b>5. Data Design.....</b>	<b>49</b>
5.1 Data Description .....	49
A. Registration / Authentication.....	49
B. Movie Facts .....	50
C. Search .....	51
D. Discussion Board.....	52
5.2 Database Dictionary / Schema.....	53
A. Registration / Authentication.....	53
B. Movie Facts .....	54
C. Search .....	55
D. Discussion Bard.....	56
5.3 Full Database Model.....	57
<b>6. Requirements Matrix.....</b>	<b>58</b>

## TABLE OF FIGURES

Figure 1 MG Architectural Design .....	10
Figure 2 Decomposition Description .....	11
Figure 3 Data Flow Diagram .....	12
Figure 4 Registration Screen Image.....	14
Figure 5 Login Screen Image.....	17
Figure 6 Change Password Screen Image.....	19
Figure 7 Select Screen Image .....	21
Figure 8 Sort Screen Image.....	23
Figure 9 Rate Screen Image.....	25
Figure 10 Discussion Board Screen Image (Home Page).....	34
Figure 11 Discussion Board Screen Image (Page) .....	34
Figure 12 Post Screen Image .....	35
Figure 13 Reply Screen Image.....	37
Figure 14 Validate Screen Image.....	39
Figure 15 Account UML Static Class Diagram.....	41
Figure 16 Movie Facts UML Static Class Diagram.....	43
Figure 17 Discussion Board UML Static Class Diagram .....	46
Figure 18 Registration/Authentication ERD Diagram.....	49
Figure 19 Movie Facts ERD Diagram .....	50
Figure 20 Discussion Board ERD Diagram.....	52

Module Assignment	Individual
Movie Facts	Enrique Paredes
Search	Pawel Zajac
Discussion Board	Divina Gorospe

## 1. INTRODUCTION

### 1.1 PURPOSE

[Enrique]

The purpose of the ADD (Analysis and Design Document) is to explain and illustrate how all features of MovieGo will be implemented. Screen mock-ups will be used to show stakeholders what the MovieGo website might look like and show them how the features work if implemented. The intended audience today is the stakeholder but also fellow software developers to showcase how these features will be grouped into classes and then proceed to building the final domain model.

### 1.2 SCOPE

[Enrique]

MovieGo system is designed to provide users movie information and to facilitate communication between users.

**Below is the list of the features of the website.**

- The website will have information about movies.
- Use of MySQL database to store content with multiple tables
- A website coded in HTML
- Use of PHP scripting language
- Users can register to become a member
- Non-member and members will have the capability to:
  1. Search for specific movies using any of these criteria: title, genre, actor name
  2. View movie information: synopsis, cast, awards, year of release, and genre
  3. Sort movies
- Members can participate in the Discussion Board
  1. Post a question/movie recommendation
  2. Reply to posts
- Users can quickly search for a specific movie based on these criteria's: movie title, actor's name. Users can also search by clicking on genre for a list of movies in that category.
- Admin will have the following capabilities:
  1. Add/update movie to the database
  2. Review posts: approve/reject
  3. Monitor the entire website

**Below is the list of the features that is out of scope:**

- The website will not have the capability to purchase movie tickets.
- The website will not contain movie trailers/videos.

## 1.3 OVERVIEW/REFERENCES

[Divina]

This document is grouped into seven main sections as outlined below:

### ***Section 1 Introduction***

This is the current section which outlines the documents purpose, scope, overview, and definitions and acronyms. This section introduces the overall structure of the document. The target audience for this section comprises of all stakeholders

### ***Section 2 System Architecture***

This section provides a high level description of the MovieGo webapp and all its modules and components. It represents the structure of data and program components that will be built in the webapp. It includes context diagram and description of the functionality. The target audience for this section comprises of Software Engineers, Programmers, and Database Designers. Although it is geared towards technical people, other non-technical stakeholders are encouraged to read this section.

### ***Section 3 Human Interface Design***

This section provides a detailed description and specification for each use case of each module. It includes an overview of the user interface (includes a mock-up of the web page/form) required inputs, the corresponding outputs, actions, pre and post conditions and validations on the inputs.

### ***Section 4 Component Design***

This section provides a more detailed description of the domain analysis presented in the SRS document. It defines the classes for each use case, their responsibilities, attributes, methods, and the relationships between them.

### ***Section 5 Data Design***

This section provides a detailed description of the data for each by utilizing an ER diagram (5.1). The ER diagram identifies the relationships and attributes in each module. The database dictionary/schema (5.2) describes the different tables that are in each module. The full database model (5.3) depicts the full ER diagram of the MovieGo web app.

### ***Section 6 Requirements Matrix***

This section provides a cross-reference that traces components and tables to the original requirements from the SRS document.



## 1.4 DEFINITIONS AND ACRONYMS:

[Enrique & Divina)

Term/Abbreviation	Definition
<b>ADD</b>	Analysis and Design Document
<b>Administrator (Admin)</b>	User with special privilege to manipulate website content
<b>Bootstrap</b>	Bootstrap is a free and open-source front-end web framework for designing websites and web applications. It contains HTML- and CSS-based design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions. Unlike many web frameworks, it concerns itself with front-end development only. <sup>1</sup>
<b>Database (DB)</b>	A database is an application that manages data and allows fast storage and retrieval of that data
<b>Dynamic Website</b>	Dynamic websites contain Web pages that are generated in real-time. These pages include Web scripting code, such as PHP or ASP. When a dynamic page is accessed, the code within the page is parsed on the Web server and the resulting HTML is sent to the client's Web browser.[1]
<b>ER Diagram</b>	An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is a component of data. In other words, ER diagrams illustrate the logical structure of databases. <sup>2</sup>
<b>HTML</b>	Hypertext Markup Language, a standardized system for tagging text files to achieve font, color, graphic, and hyperlink effects on World Wide Web pages.
<b>Member</b>	Registered user
<b>Non-registered user</b>	General users that is not registered with the MovieGo web app
<b>MovieGo (MG)</b>	Formal website name
<b>MySQL</b>	Open source database that will be used for this project
<b>Non-member</b>	General website user
<b>PHP</b>	Hypertext Preprocessor; scripting language that will be used for this project
<b>TBD</b>	To be determined
<b>Web Browser</b>	A web browser (commonly referred to as a browser) is a software application for retrieving, presenting and traversing information resources on the World Wide Web.

<sup>1</sup> [https://en.wikipedia.org/wiki/Bootstrap\\_\(front-end\\_framework\)](https://en.wikipedia.org/wiki/Bootstrap_(front-end_framework))

<sup>2</sup> <https://www.smartdraw.com/entity-relationship-diagram/>

## 2. SYSTEM ARCHITECTURE

### 2.1 SYSTEM OVERVIEW

[Enrique]

MovieGo is designed for moviegoers or stay at home viewers who wish to know more about the movies they love/want to watch. The system is broken down into four different modules which consist of account registration, getting movie facts, searching for titles, and posting into a discussion board. The modules are then broken down into different use cases which the user might encounter when on this module. The system will be designed to have three separate users. These users include non-registered users, registered users, and the admins. The admins for this system will consist of Enrique (Movie Facts), Pawel (Search), and Divina (Discussion Board).

### 2.2 ARCHITECTURAL DESIGN

[Divina]

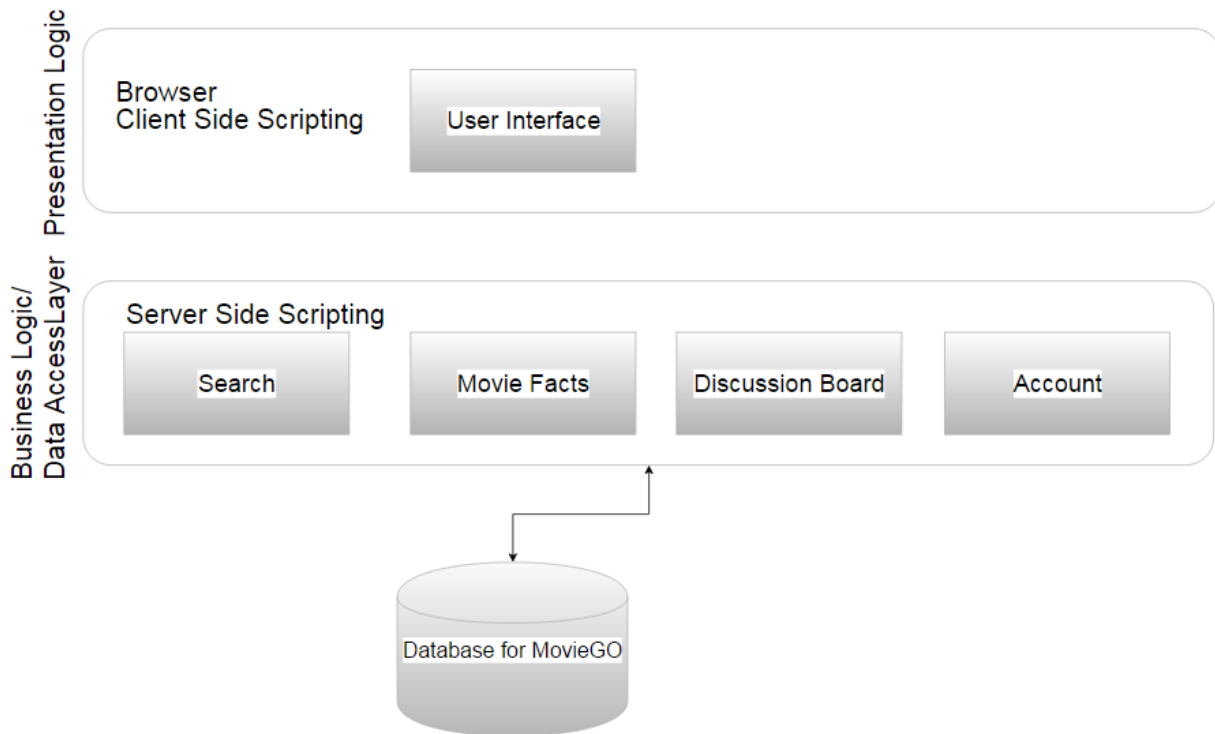


Figure 1 MG Architectural Design

The MovieGo web application will use a layered system architecture. At the top level is the User Interface where each user interacts with the web app. A common theme will be used throughout the site and will be accomplished by utilizing Bootstrap. This level will use client side scripting.

The second layer will comprise of the business logic and data access layer. This is where the different modules will reside. It will use server side scripting (PHP) to access the database for the web app. MovieGo's security will be employed by using sessions and cookies to access restricted sites. A user login will be required and will be checked in the database.

The project will utilize a single database that will store all the data and information for the MovieGo web application.

## 2.3 DECOMPOSITION DESCRIPTION

[Divina]

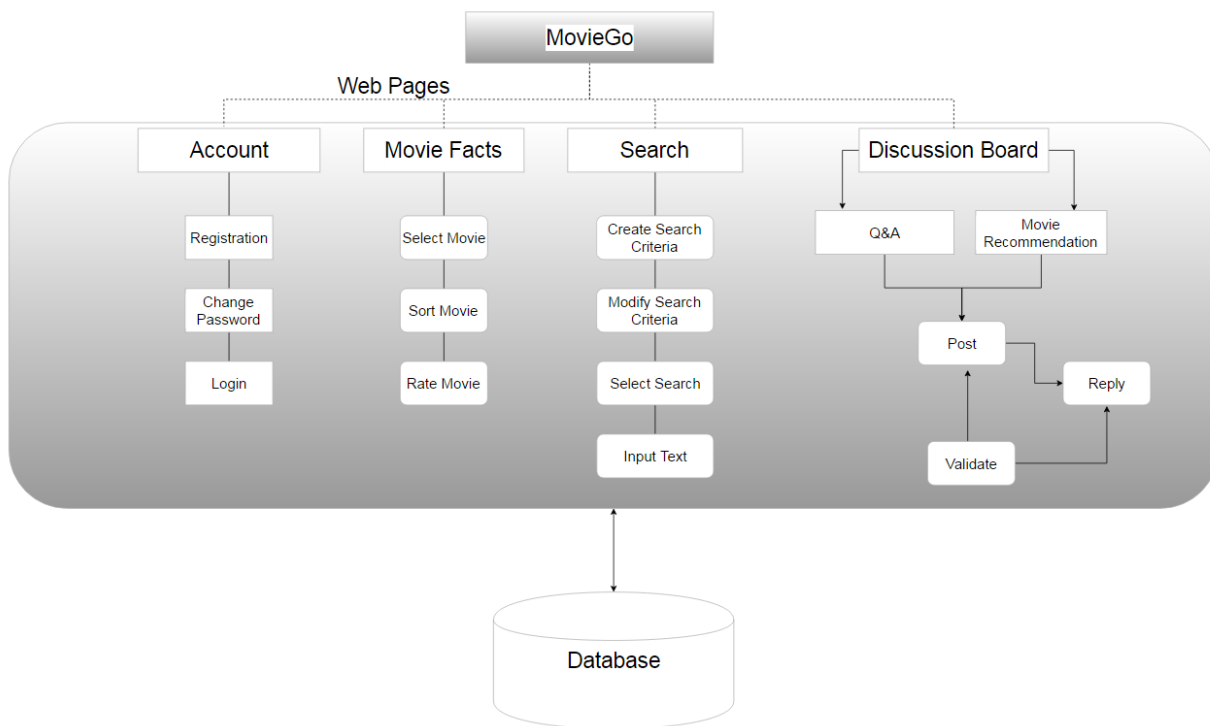
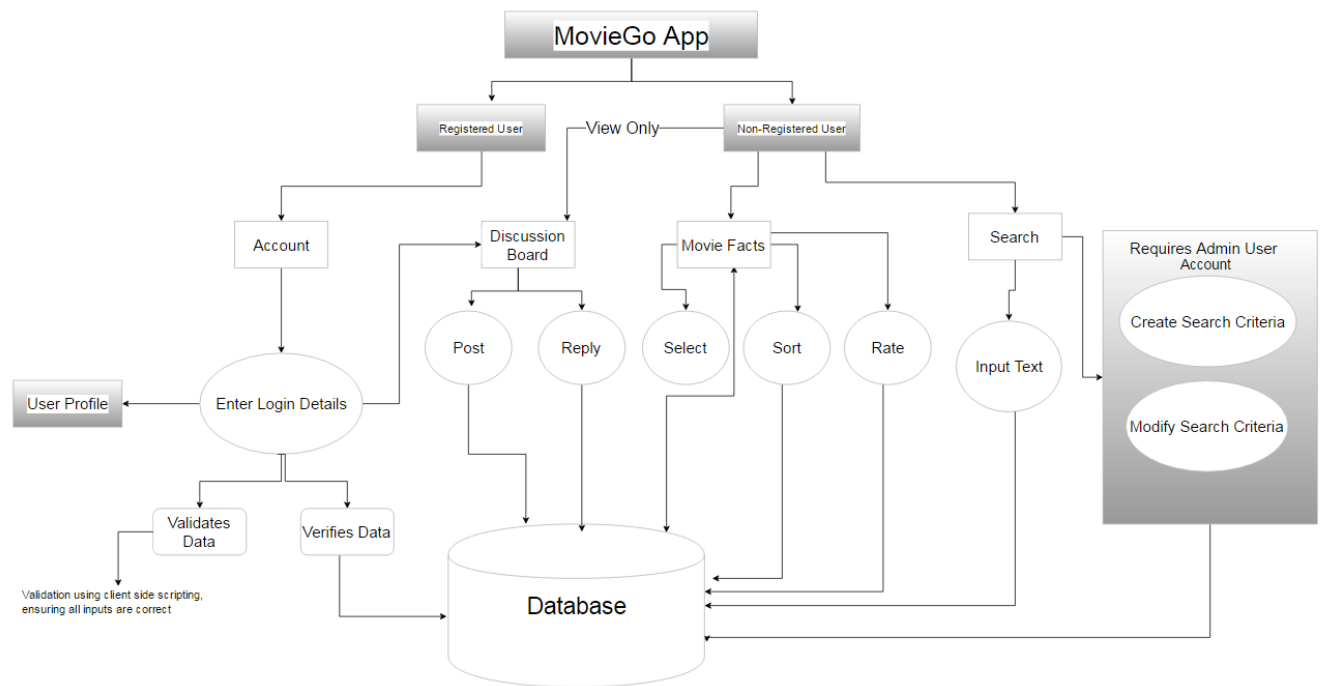


Figure 2 Decomposition Description

The above diagram depicts the decomposition of the four modules and their use cases below each module that will be used in the MovieGo app. The four main components: Account, Movie Facts, Search, and Discussion Board will each have a page or form that will require user inputs. All the modules will access the single database for this project.



**Figure 3 Data Flow Diagram**

The data flow diagram for the MovieGo web app is pictured above. Registered users will have access to their user profile and the Discussion Board module. Non-registered users will have access to the Movie Facts and Search Modules, and will only have a view capability of the Discussion Board module.

## 2.4 DESIGN RATIONALE

[Divina]

The Human Interface Design is the architecture design that is used in Section 3.1. MovieGo is a web application, thus, we have determined that using Interface Design will effectively communicate our design rationale. The MovieGo web app is a dynamic web application that will require user inputs, actions, validations, and will utilize form elements. This was considered in creating the screen layout for each use cases in the four modules that makes up the entire web app being developed.

The MG web app will abide by the Golden Rules of Interface Design:

1. Place the user in control – MG will use web form elements and each of those elements will perform the specific functions such as a button will be designed to look like a button and will function as a button.
2. Reduce the user's memory load – MG will use breadcrumbs to help users know which page they are on and will also enable users to navigate through the pages easily. Form elements will clearly define what inputs are required.

3. Make the interface consistent – MG web app will be utilizing bootstrap to create a consistent theme on all its web pages. The color scheme for the web app will consists of white text, black/gray background.

### 3. HUMAN INTERFACE DESIGN

#### 3.1 ACCOUNT

##### A. REGISTRATION

##### A1. OVERVIEW OF USER INTERFACE

The registration page will allow new users to register with the system. The user must enter all fields before submitting the form.

##### A2. SCREEN IMAGES

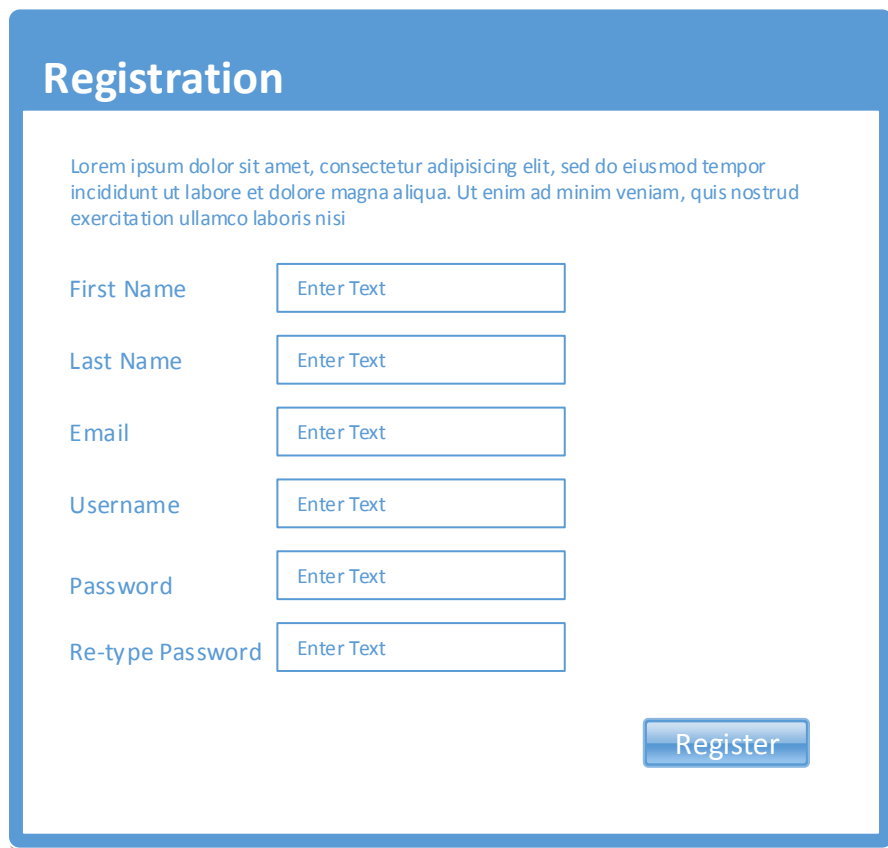
A screenshot of a web registration form titled "Registration". The form is enclosed in a blue border. At the top, there is a blue header bar with the title "Registration" in white. Below the header, there is a paragraph of placeholder text: "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi". Below the text, there are six input fields, each with a label to its left and a text box containing the placeholder "Enter Text". The labels are "First Name", "Last Name", "Email", "Username", "Password", and "Re-type Password". At the bottom right of the form, there is a blue button with the text "Register" in white.

Figure 4 Registration Screen Image

##### A3. SCREEN OBJECTS AND ACTIONS:

N/A

##### A4. INPUT AND OUTPUT

<i>Input</i>	
<b>First Name</b>	String representing the first name of user. Minimum of 1 and maximum of 50 characters.

<b>Last Name</b>	String representing last name of user. Min 1 and maximum 50 characters
<b>Email:</b>	User's email address used for communication and verification. Should match standard email pattern "name@server.com". Minimum of 10 characters, maximum of 50 characters.
<b>Username</b>	string representing the username can be a minimum of 6 characters and a maximum length of 20.
<b>Password</b>	Also a string. Must be a minimum of 8 characters and a maximum of 20. Must contain on capital letter and one number.
<b>Re-enter Password</b>	matches Password field. Used only for verification of password.
<i><b>Output</b></i>	
	Upon successful creation of user, the system will display a welcome message or a warning message. In addition, if the data entered is not valid, the system will show a list of the fields with invalid data in red font.
<i><b>Possible Messages</b></i>	
Successful	"Welcome [First Name] [Last Name]. Thank you for registering with us!"
Failed	"There is an error with your information. Please review the fields highlighted in red and re-enter the information"

---

## A5. ACTIONS

Upon submitting the form, the system will validate that all required fields have input, and that each field matches the required data in the required format.

- If the validation is successful, the system will run a query to store the user's information in the database.
- If the validation failed, the system will abort transaction and return an error message for each invalid field.

---

## A6. PRE AND POST CONDITIONS

<i><b>Pre-conditions</b></i>	
	User has not registered with the system already. The system should validate that the email and username are unique within the stored user values.
<i><b>Post-Conditions</b></i>	
	User record is stored successfully. A user session is created.

---

## A7. VALIDATION

The following validations will be performed on the submitted data:

- **First Name, Last Name:** fields must be at least the minimum specified length and no more than the maximum length.
- **Username:** must be unique
- **Email:** must be unique and must meet standard email address format

- **Password:** Must meet standard strong password policy. The password must be stored using a secure mechanism that will prevent unauthorized access. All passwords will be hashed and stored as such in the database.

---

## **A8. DIFFERENT ACTORS CASES NEED TO BE INCLUDED (CLIENT AND ADMINISTRATOR)**

Client



---

## B. LOGIN

### B1. OVERVIEW OF USER INTERFACE

The login page will allow students/faculty to access the system's features. They can also reset their password or unlock their account.

### B2. SCREEN IMAGES

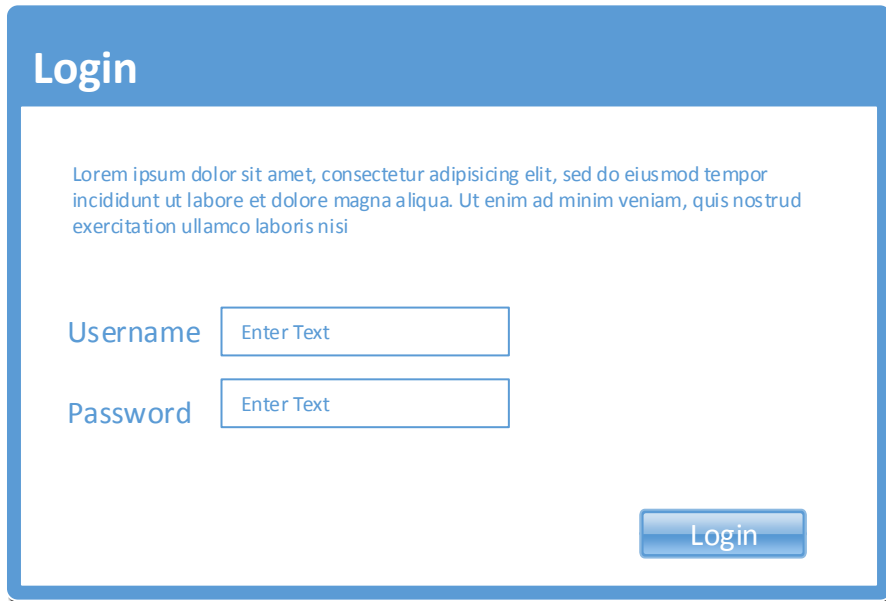


Figure 5 Login Screen Image

### B3. SCREEN OBJECTS AND ACTIONS:

### B4. INPUT AND OUTPUT

<i>Input</i>	
<b>Username</b>	string representing the username can be a minimum of 6 characters and a maximum length of 20.
<b>Password</b>	Also a string. Must be a minimum of 8 characters and a maximum of 20. Must contain one capital letter and one number.
<i>Output</i>	
	A session is created for the successful logged in user to access information granted to the specific user role. Also a message welcoming the user or an error (depending on the credentials provided)

### B5. ACTIONS

Once the user has entered their user name and password, the “Login” button will prompt the site to authenticate the user. Their role will be determined and they will be granted access to the

modules available to that class of user. Based on the assigned role, the user might be re-routed to the faculty page or the student page.

---

## B6. PRE AND POST CONDITIONS

<i>Pre-conditions</i>	
	Faculty and students both require a valid user name and password, the user name will be assigned upon acceptance or employment to the institution.
<i>Post-Conditions</i>	
	Successful entry of credentials will result in a portal page. Unsuccessful entry of credentials will result in an error page, where the user is required to re-submit credentials.

---

## B7. VALIDATION

The following validations will be performed on the submitted data:

Both the user name and password will be validated by comparing them to the specific user credentials stored in the database.

**Password Security:** The password must be stored using a secure mechanism that will prevent unauthorized access. All passwords will be hashed and stored as such in the database.

---

## B8. DIFFERENT ACTORS CASES NEED TO BE INCLUDED (CLIENT AND ADMINISTRATOR)

---

## C. CHANGE PASSWORD

### C1. OVERVIEW OF USER INTERFACE

The Change Password page will allow the user to change their password by entering the current one, and the new password.

---

### C2. SCREEN IMAGES

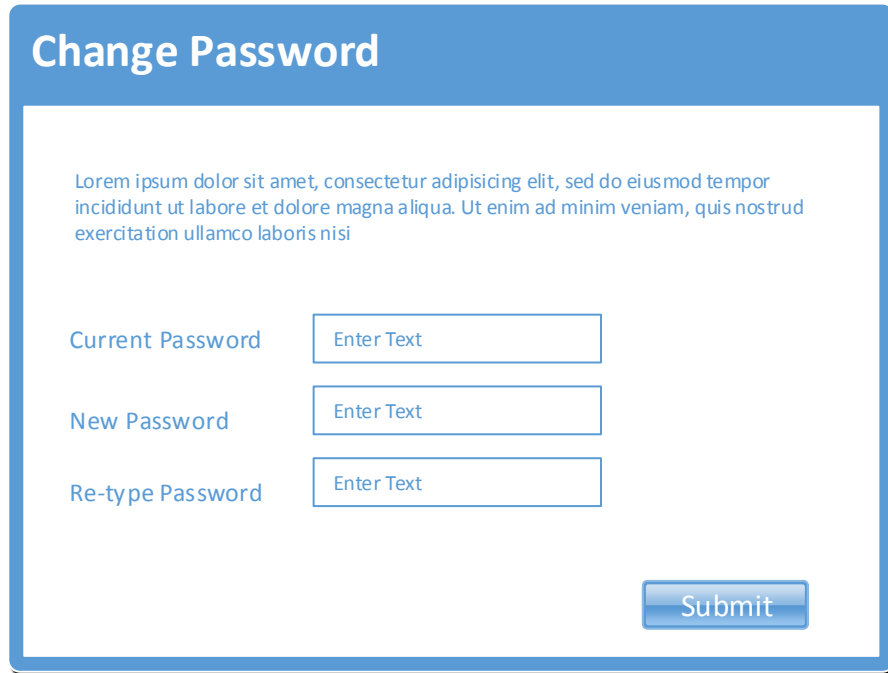
A screenshot of a web form titled "Change Password". The form has a blue header bar with the title. Below the header, there is a paragraph of placeholder text: "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi". Below the text, there are three input fields labeled "Current Password", "New Password", and "Re-type Password". Each field has a placeholder text "Enter Text". At the bottom right of the form, there is a blue "Submit" button.

Figure 6 Change Password Screen Image

---

### C3. SCREEN OBJECTS AND ACTIONS

---

#### C4. INPUT AND OUTPUT

<i>Input</i>	
<b>Current Password</b>	String. Must be a minimum of 8 characters and a maximum of 20. Must contain on capital letter and one number.
<b>New Password</b>	Also a string. Must be a minimum of 8 characters and a maximum of 20. Must contain on capital letter and one number.
<b>Re-type Password</b>	Same requirements as New Password
<i>Output</i>	
	If the user enter the correct “current password”, the system will store the new password and return a “Successfully change of password” message

---

### C5. ACTIONS

Upon submission of the form, the system will validate the user has entered the correct password. To do so, the system will use the current session username and retrieve the current password from the database. The “Current Password” entered will be verified against the stored password and if successful, it will also validate both “New Password” and “Re-type Password” values are identical.

If all validation passes, then the system will update the user account with the new password entered.

---

## C6. PRE AND POST CONDITIONS

<i>Pre-conditions</i>	
	User is already registered and the account is active.
<i>Post-Conditions</i>	
	Successful transaction will update the user’s password to the specified new values.

---

## C7. VALIDATION

The following validations will be performed on the submitted data:

Current password will be validated against the stored password. New password will be validated against the “Re-type password” value, which should match. Also, new password must meet strong password policy.

**Password Security:** The password must be stored using a secure mechanism that will prevent unauthorized access. All passwords will be hashed and stored as such in the database.

---

## C8. DIFFERENT ACTORS CASES NEED TO BE INCLUDED (CLIENT AND ADMINISTRATOR)

Client

## 3.2 MOVIE FACTS

### A. SELECT

#### A1. OVERVIEW OF USER INTERFACE

The select page will show the user a list of movies populated by the MovieGo web app. From here, the user may click on any movie and see information about the movie. No input fields are required for this portion of the feature.

#### A2. SCREEN IMAGES

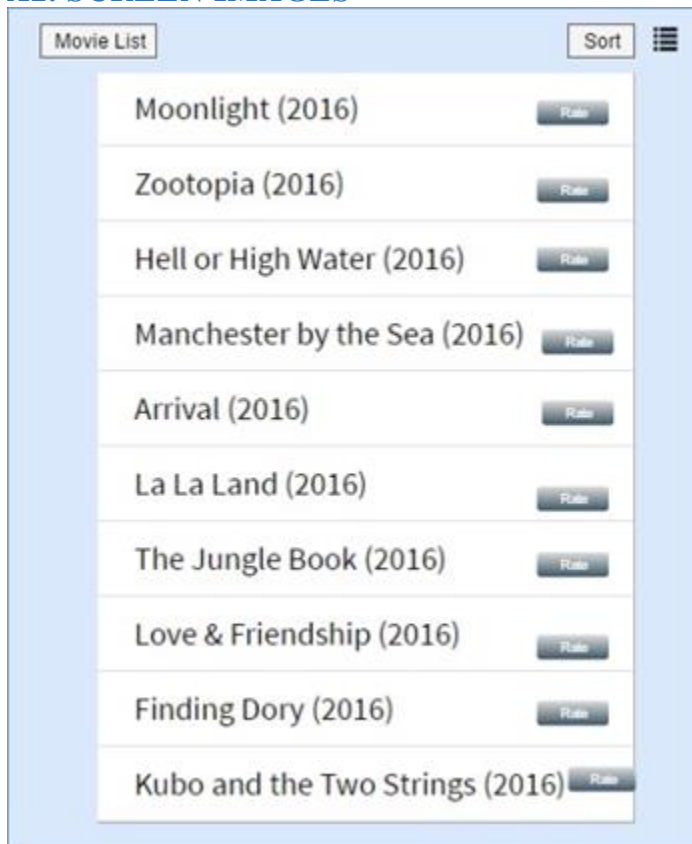


Figure 7 Select Screen Image

#### A3. SCREEN OBJECTS AND ACTIONS

The first screen will have movie titles, rate buttons, and the sort button as its objects. For this use case the only object that will have focus is the movie title. Once the user clicks on a title, they will be taken to a new page that will display to them information regarding the movie. \*Rate and sort buttons will be discussed below\*

#### A4. INPUT AND OUTPUT

<i>Input</i>	
<b>Movie Title</b>	From the list of titles given to the user, they may choose a title they wish

	to get more info on.
<b>Output</b>	
<b>Movie Facts</b>	A new page will populate the screen and show the user information such as year of release, main actor, US total gross, etc.
<b>Possible messages</b>	
<b>N/A</b>	This use case will only allow users to select from a pre-existing list so no errors may occur up to this point.

## A5. ACTIONS

The system will look for the information requested by the client in the DB.

- If the information is found, the user will be taken to a new screen where they will see their movie results
- If information is not found, the system will return a message letting the user know that MovieGo is currently working on resolving the issue.

## A6. PRE AND POST CONDITIONS

<b>Pre-conditions</b>	
	No pre-conditions exist as the user does not have to be registered to view and select different movie titles
<b>Post-Conditions</b>	
	If the user selects the correct movie, then they successfully completed the first use case which allows them to select a movie title.

## A7. VALIDATION

The following validations will be performed on the submitted data:

The selected title must be from the list provided by MovieGo. If the title is found in the database, then the information about the selected title will be given to the user.

## A8. DIFFERENT ACTORS CASES NEED TO BE INCLUDED (CLIENT AND ADMINISTRATOR)

Client: The client in this case will use MovieGo and select a title they wish to know more about.

Administrator: Will populate the database with movie titles that the user may select from.

---

## B. SORT

### B1. OVERVIEW OF USER INTERFACE

A user will first see a movie list implemented by the admins of MovieGo. The user will see the option to sort the list by year of movie, rating, genre, etc. Only one sort criteria are allowed per list. Once the criteria are entered, a new list will appear with the sorting being completed. The user may then re-sort the list by a different sort criteria if they wish.

#### B2. Screen Images

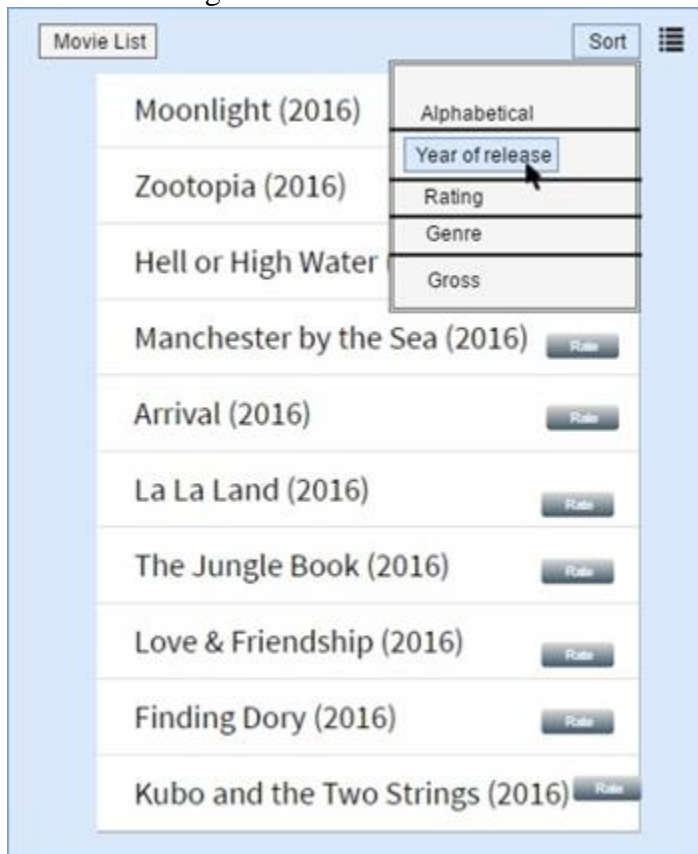


Figure 8 Sort Screen Image

---

### B3. SCREEN OBJECTS AND ACTIONS:

The object's in this screen, like stated before, include the movie titles, rate buttons, and the sort button. This time the sort button is selected and a new side menu becomes visible. Required action by the user at this point is to select a sort criteria to which MovieGo will populate a new list with the specified criteria.

---

### B4. INPUT AND OUTPUT

<i>Input</i>	
<b>Sort Criteria</b>	The user is required to enter sort criteria so MovieGo can populate a new

	list. Once the criteria are chosen, MovieGo will find the query in the database which satisfies this criterion and display the results to the user.
<b>Output</b>	
<b>Sorted List</b>	The new movie list will appear with the criteria that the user chose. This list can then be re-sorted if the user chooses to.
<b>Possible messages</b>	
<b>N/A</b>	A message will not appear as a user is entering criteria from a predefined list that the admins of MovieGo created.

## B5. ACTIONS

- The system will fetch the sort criteria that the user has selected
- Once the sort criteria is found, the user will be taken to a new page where they will see the results they were looking for
- Users may re-sort the list if they please by choosing a different sort criteria

## B6. PRE AND POST CONDITIONS

<b>Pre-conditions</b>	
	User must select criteria to which it will sort the list or else to website will sort the list by default which is year of release.
<b>Post-Conditions</b>	
	Users will be taken to a new screen where they will see the movie list sorted to their liking,

## B7. VALIDATION

The following validations will be performed on the submitted data:

The selected criteria must be from the list provided by MovieGo. If the sorted criteria are found in the database, then the web application will return a new page with the list sorted.

## B8. DIFFERENT ACTORS CASES NEED TO BE INCLUDED (CLIENT AND ADMINISTRATOR)

Client: The client has the option to sort a movie list with criteria given to them by MovieGo.

Admin: The administrator will create queries which will sort the movie lists into different groups. Once a user selects criteria, the proper query will be called and a new page will appear with the newly sorted list.



## C. RATE

### C1. OVERVIEW OF USER INTERFACE

The rate use case will take a user to a new page where they can rate the movie based on how much they liked/disliked the movie title. A text field will appear and the user may enter any number from 0-100. This number will represent a percentage which will then be factored into the current rating to provide a brand-new rating.

### C2. SCREEN IMAGES

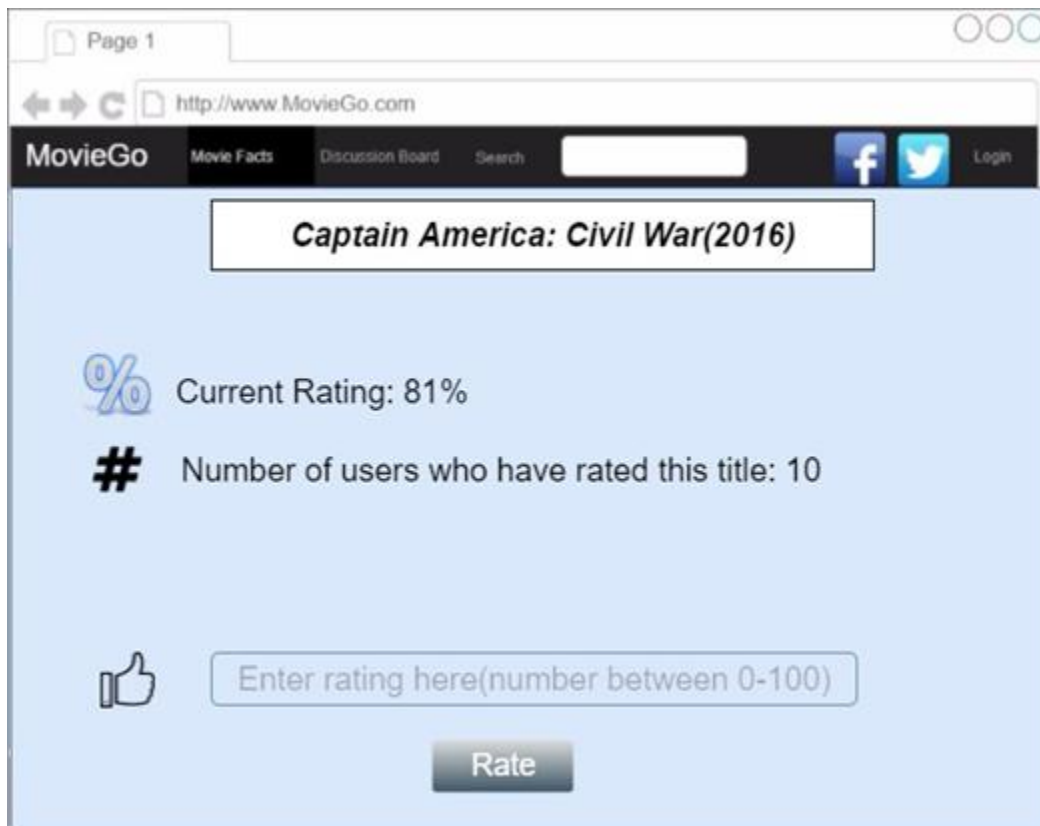


Figure 9 Rate Screen Image

### C3. SCREEN OBJECTS AND ACTIONS:

The two objects that are important in the rate use case is the blank text box and the rate button. The user will be asked to enter a rating they wish the movie deserves and then proceed to press the rate button. Once the button is pressed, the web app will display a thank you message and update the current rating to a new rating which includes the user's rating.

---

## C4. INPUT AND OUTPUT

<b>Input</b>	
<b>Movie title</b>	The user should pick a movie from a list that is provided to them by MovieGo.
<b>Rating</b>	The rating will be based on what the user enters in the textbox. The rating must be a number and all other inputs will be invalid.
<b>Rate Button</b>	Once the user enters the rating the rate button should be pressed in order to update the current rating to the new one.
<b>Output</b>	
<b>Rating</b>	If the user enters all the correct info, then the new rating will be displayed onto the screen
<b>Invalid Window</b>	The input must be a number and if anything, else is entered, the user will get an error message.
<b>Thank you, window</b>	When criteria is met, MovieGo will display a message thanking the user for rating the movie title they selected.
<b>Possible messages</b>	
<b>Error/Thank You</b>	Depending on the user input, the user will either receive a thank you message or an error message.

---

## C5. ACTIONS

- The system will check to make sure that the input received is a number.
- If another character is entered, the user will see an error message window and will be required to try rating the movie title again.
- If the input is correct, the system will add the user rating to the rating currently residing in the DB.
- User will be able to see the new rating which includes their input

---

## C6. PRE AND POST CONDITIONS

<b>Pre-conditions</b>	
	The user must enter a number into the textbox. If it is left blank or if any other characters are entered, the user will not have access to rate the movie. All users must also be members in order to use this feature.
<b>Post-Conditions</b>	
	Once all conditions are met then the user will have successfully rated a movie title.

---

## C7. VALIDATION

The following validations will be performed on the submitted data:

The input received in the textbox will be checked to make sure that the input is a number. A number is required because it may be added to the total of previous ratings.

---

## **C8. DIFFERENT ACTORS CASES NEED TO BE INCLUDED (CLIENT AND ADMINISTRATOR)**

Client: Will have the power to rate any movie they like. The business rule they would have to follow is they are only allowed to rate a movie title once. This is done to keep the integrity of MovieGo in a positive note.

Administrator: The admin will come up with an algorithm that will allow the users rating to be added on to the current rating of the movie. The admin will also create a validation which checks to make sure a user isn't rating the movie more than once and checks to make the user is a registered member.

### 3.3 SEARCH

#### A. CREATE

##### A1. OVERVIEW OF USER INTERFACE

The create page will allow creating search criteria's by adding specific keywords and tags that will be used in searching the webpage.

##### A2. SCREEN IMAGES

##### A3. SCREEN OBJECTS AND ACTIONS:

##### A4. INPUT AND OUTPUT

<i>Input</i>	
<i>Output</i>	
<i>Possible Messages</i>	

##### A5. ACTIONS

##### A6. PRE AND POST CONDITIONS

<i>Pre-conditions</i>	
<i>Post-Conditions</i>	

##### A7. VALIDATION

The following validations will be performed on the submitted data:

---

## **A8. DIFFERENT ACTORS CASES NEED TO BE INCLUDED (CLIENT AND ADMINISTRATOR)**

---

## B. MODIFY

---

### B1. OVERVIEW OF USER INTERFACE

---

### B2. SCREEN IMAGES

---

### B3. SCREEN OBJECTS AND ACTIONS:

---

### B4. INPUT AND OUTPUT

<i>Input</i>	
<i>Output</i>	
<i>Possible Messages</i>	

---

### B5. ACTIONS

---

### B6. PRE AND POST CONDITIONS

<i>Pre-conditions</i>	
<i>Post-Conditions</i>	

---

### B7. VALIDATION

The following validations will be performed on the submitted data:

---

## B8. DIFFERENT ACTORS CASES NEED TO BE INCLUDED (CLIENT AND ADMINISTRATOR)

---

### C. INPUT

---

#### C1. OVERVIEW OF USER INTERFACE

---

#### C2. SCREEN IMAGES

---

#### C3. SCREEN OBJECTS AND ACTIONS:

---

#### C4. INPUT AND OUTPUT

<i>Input</i>	
<i>Output</i>	
<i>Possible Messages</i>	

---

#### C5. ACTIONS

---

#### C6. PRE AND POST CONDITIONS

<i>Pre-conditions</i>	
<i>Post-Conditions</i>	

---

#### C7. VALIDATION

The following validations will be performed on the submitted data:

---

## C8. DIFFERENT ACTORS CASES NEED TO BE INCLUDED (CLIENT AND ADMINISTRATOR)

---

### D. SELECT

---

#### D1. OVERVIEW OF USER INTERFACE

---

#### D2. SCREEN IMAGES

---

#### D3. SCREEN OBJECTS AND ACTIONS:

---

#### D4. INPUT AND OUTPUT

<i>Input</i>	
<i>Output</i>	
<i>Possible Messages</i>	

---

#### D5. ACTIONS

---

#### D6. PRE AND POST CONDITIONS

<i>Pre-conditions</i>	
<i>Post-Conditions</i>	



---

## **D7. VALIDATION**

The following validations will be performed on the submitted data:

---

## **D8. DIFFERENT ACTORS CASES NEED TO BE INCLUDED (CLIENT AND ADMINISTRATOR)**

### 3.4 DISCUSSION BOARD

The Discussion Board Module will enable registered users to create a post or reply to posts. This module will be accessible via the navigation bar. Users will be able to access the pages for the two categories: Q&A and Movie, from the drop-down menu (pictured below). As noted on the SRS, any user can view the discussion board, but only registered users can create a post, or reply to a post. Users who attempt to reply or create a post will be prompted to log-in.

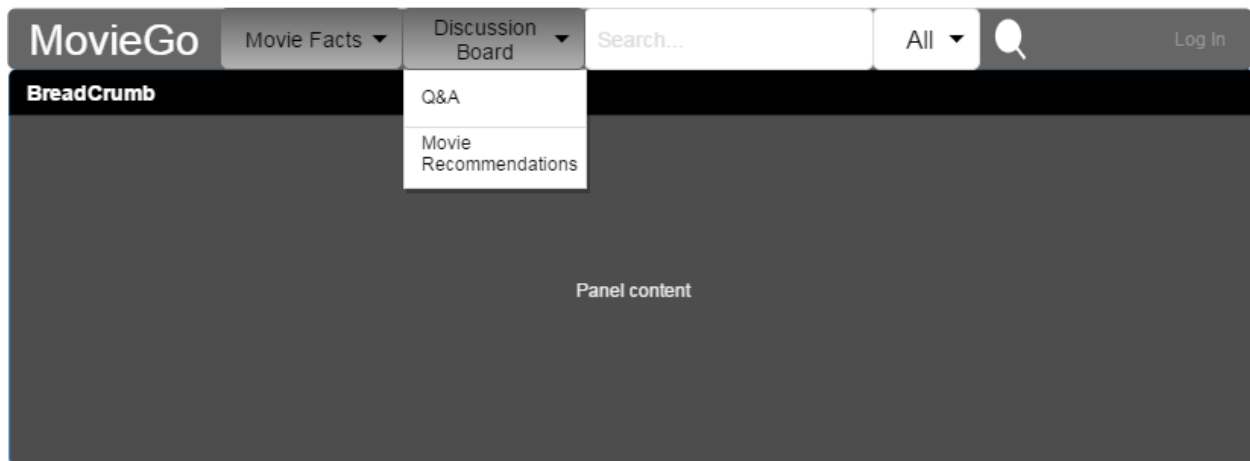


Figure 10 Discussion Board Screen Image (Home Page)

Once the user chooses a category, they will be redirected to the appropriate page. A mock-up is pictured below.

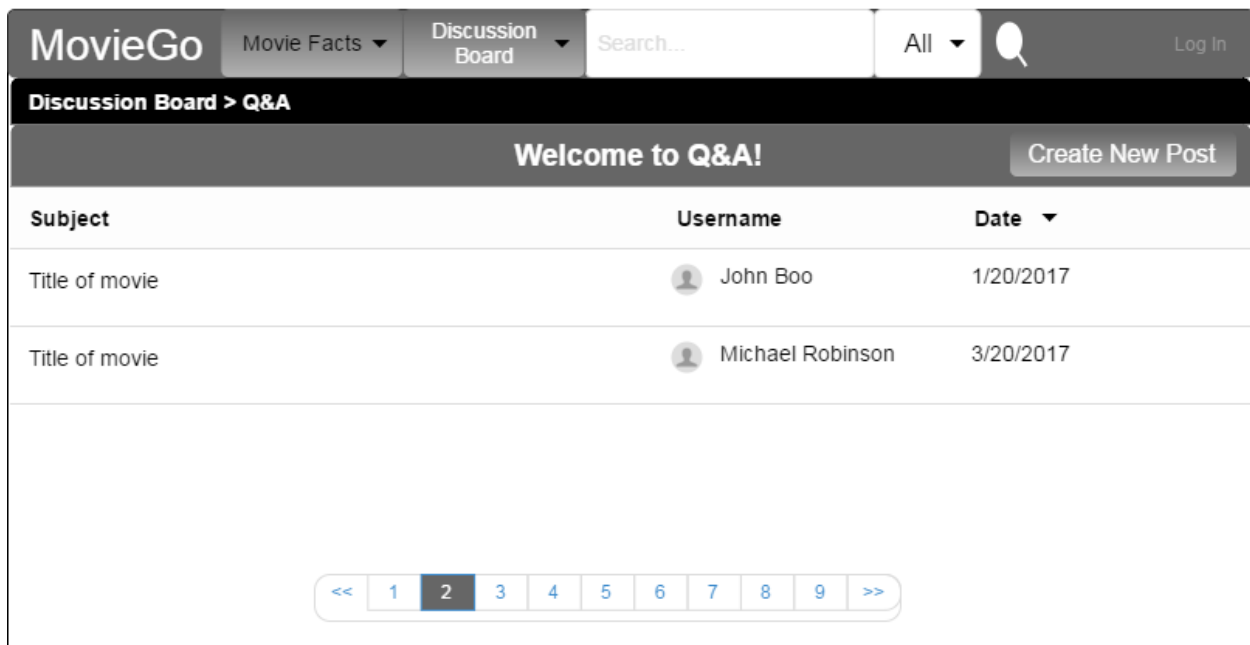


Figure 11 Discussion Board Screen Image (Page)

---

## A. POST

### A1. OVERVIEW OF USER INTERFACE

The post use case for the Discussion Board module will allow registered users to submit a post. All fields are mandatory and cannot be left blank.

---

### A2. SCREEN IMAGES



Figure 12 Post Screen Image

---

### A3. SCREEN OBJECTS AND ACTIONS:

The post form will have the following:

- One-line text field – text field for user to type in the Subject (title of post)
- Multi-line text field – text field for user to type in the body of the post
- Button – Submit
- Button - Close

---

### A4. INPUT AND OUTPUT

<i>Input</i>	
Subject	String representing the title of the post Can be a minimum of 1 character to 50 characters Cannot be left blank
Body	String representing the users' message to be posted. Can be a minimum of 1 character to 200 characters Cannot be left blank.
<i>Output</i>	
	Once the user clicks the submit button, the system will validate the fields. If any of

	the fields are left blank, the system will highlight those fields and show a red text “Field cannot be left blank.” beside the invalid field. Once validated, the post will appear in the discussion board.
<b>Possible Messages</b>	
Successful	Your post has been submitted.
Failed	Please correct the highlighted errors.

## A5. ACTIONS

- The system will validate the two required text fields.
- If the validation is successful, the system will run a query to store the user’s post in the database.
- If the validation failed, the system will abort transaction and return an error message for each invalid field.
- An Administrator will review the post. If it complies with the rules, the post will remain visible in the discussion board. If it is in violation of the rules, the post will be deleted and the user will be informed via email regarding the violation.

## A6. PRE AND POST CONDITIONS

<b>Pre-Conditions</b>	
	User must be logged in.
<b>Post-Conditions</b>	
	User post is stored successfully in the database and will be visible in the discussion board page.

## A7. VALIDATION

The following validations will be performed on the submitted data:

- Two text fields – must conform to the minimum/maximum length.

## A8. DIFFERENT ACTORS CASES NEED TO BE INCLUDED (CLIENT AND ADMINISTRATOR)

Client – The post use case only needs the Client’s input to execute the actions. Validation is done in the client side.

---

## B. REPLY

### B1. OVERVIEW OF USER INTERFACE

The reply use case for the Discussion Board module will allow registered users to reply to a post. All fields are mandatory and cannot be left blank.

---

### B2. SCREEN IMAGES

The screenshot shows the MovieGo website's Discussion Board Q&A section. At the top, there's a navigation bar with 'MovieGo', 'Movie Facts', 'Discussion Board', a search bar, and a 'Log In' link. Below this is a breadcrumb 'Discussion Board > Q&A' and a 'Welcome to Q&A!' banner. The main content area features a 'Sample Main Post' with a title, a date 'Mon Mar 20 2017 09:55:20', and a paragraph of placeholder text. Below the post is a 'Post comment' form with a large text input area and two buttons: 'Reply' and 'Cancel'.

Figure 13 Reply Screen Image

---

### B3. SCREEN OBJECTS AND ACTIONS:

- Multi-line text field – field for user to type in their comments
- Button – Submit
- Button – Cancel

---

### B4. INPUT AND OUTPUT

<i>Input</i>	
Comment	String representing the users' message to be posted. Can be a minimum of 1 character to 200 characters Cannot be left blank.

<b>Output</b>	
	Once the user clicks the submit button, the system will validate the fields. If the text field is blank, the system will highlight that field and show a red text “Field cannot be left blank.” beside the invalid field. Once validated, the post will appear in the discussion board.
<b>Possible Messages</b>	
Successful	Your post has been submitted.
Failed	Please correct the highlighted errors.

## B5. ACTIONS

- The system will validate the one required text field.
- If the validation is successful, the system will run a query to store the user’s post in the database.
- If the validation fails, the system will abort transaction and return an error message for each invalid field.
- An Administrator will review the post. If it complies with the rules, the post will remain visible in the discussion board. If it is in violation of the rules, the post will be deleted and the user will be informed via email regarding the violation.

## B6. PRE AND POST CONDITIONS

<b>Pre-conditions</b>	
	The user is logged-in. There is a parent post where the user would reply.
<b>Post-Conditions</b>	
	User post is stored successfully in the database and will be visible in the discussion board page, indented under the parent post.

## B7. VALIDATION

The following validations will be performed on the submitted data:

- One text field – must conform to the minimum/maximum length.

## B8. DIFFERENT ACTORS CASES NEED TO BE INCLUDED (CLIENT AND ADMINISTRATOR)

Client – The reply use case only needs the Client’s input to execute the actions. Validation is done in the client side.

---

## C. VALIDATE

### C1. OVERVIEW OF USER INTERFACE

The validate use case for the Discussion Board module is responsible for approving and rejecting posts submitted by the members. An administrator will review each post and determine whether it violates any of the policies.

---

### C2. SCREEN IMAGES

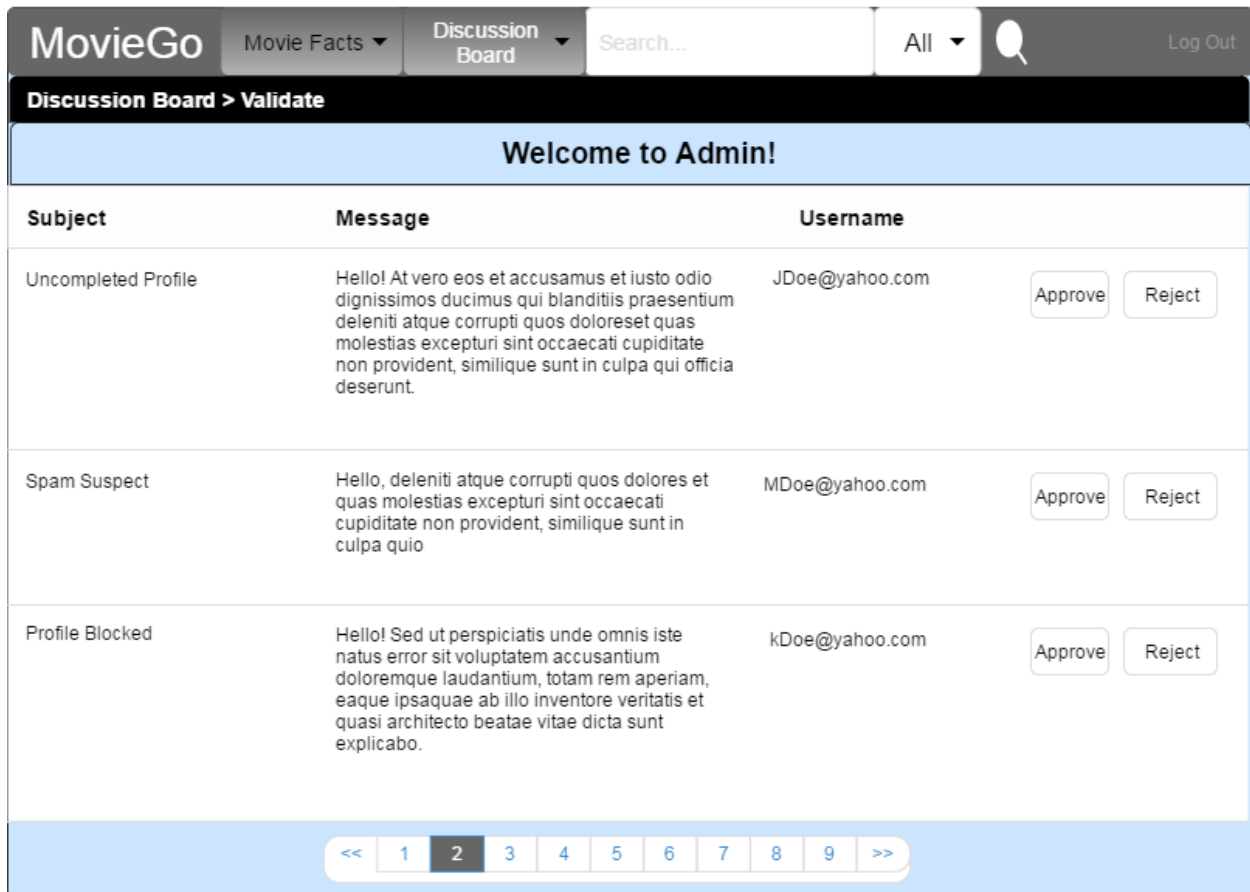


Figure 14 Validate Screen Image

---

### C3. SCREEN OBJECTS AND ACTIONS:

- Button - Approve
- Button - Reject

---

### C4. INPUT AND OUTPUT

Input	
User Posts	These are the posts submitted by users
Query	Admin will run a query ( <i>TBD</i> ) that will check for any posts that violates the policies.

Approve	Button that the admin can click to approve the post
Reject	Button that the admin can click to reject the post
<b>Output</b>	
Approved	If the post is approved, the post will remain visible in the discussion board and the database.
Rejected	If the post is rejected, the post will be deleted in the discussion board and the database. An email will be sent to the user outlining the violation and corrective action.
<b>Possible Messages</b>	
	If the query does not find a post that violates the policies, it will return a message "None found" Otherwise, it will display all the posts that have violated the policies.

---

## C5. ACTIONS

- The system will display all posts that require validation.
- After an admin validates the post, the database will be updated.

---

## C6. PRE AND POST CONDITIONS

<b>Pre-conditions</b>	
	A user with admin privileges is successfully logged in.
<b>Post-Conditions</b>	
	An updated discussion board will reflect the posts that have been validated by the admins.

---

## C7. VALIDATION

The following validations will be performed on the submitted data:

- The user's post will be validated by the admin by running a query (*TBD*).

---

## C8. DIFFERENT ACTORS CASES NEED TO BE INCLUDED (CLIENT AND ADMINISTRATOR)

Administrator – The validate use case requires admin privileges to execute its functions. An admin will review the submitted posts and reply to posts of each user. The user interface for this use case will have a different color scheme (blue background) to differentiate it from the user view of the Discussion Board.



## 4. COMPONENT DESIGN

This section describes all the classes needed for each use case in the four modules. A UML static structure class diagram and entities table will provide the attributes, relationships, and responsibilities of each class.

### 4.1 ACCOUNT

#### A. UML STATIC STRUCTURE CLASS DIAGRAM

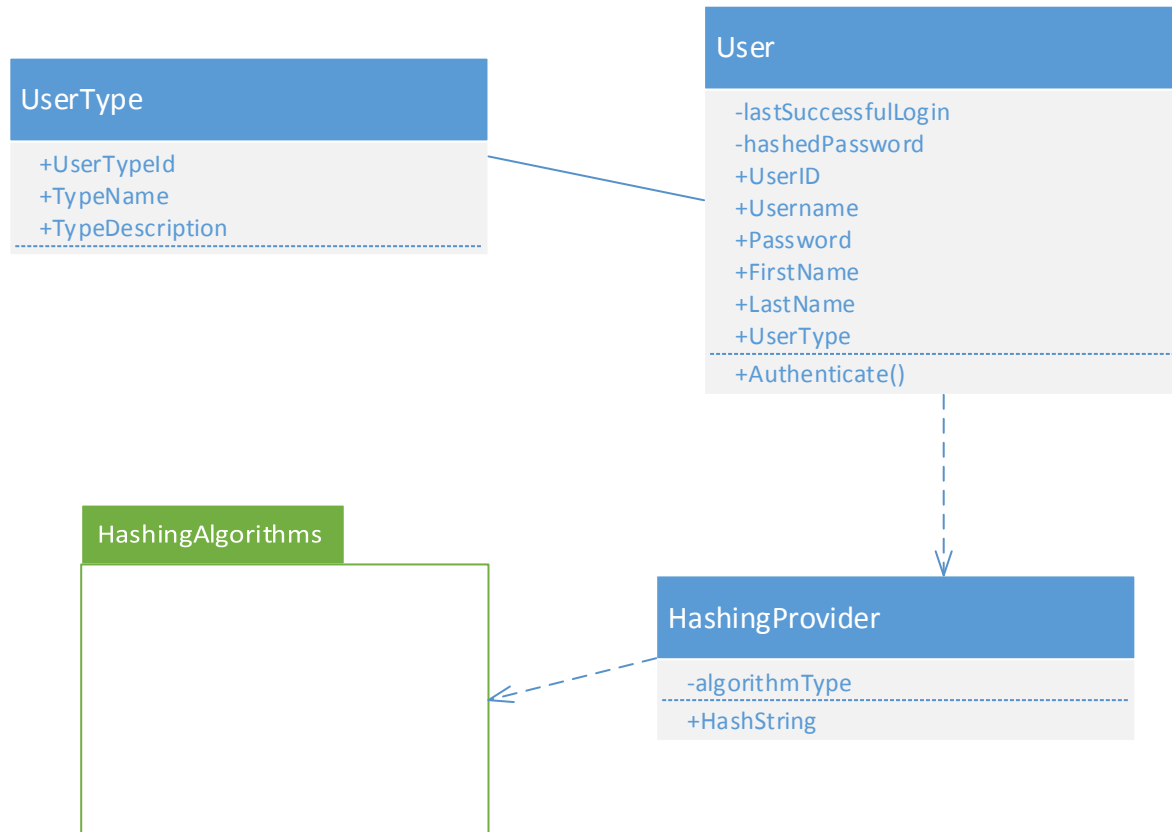


Figure 15 Account UML Static Class Diagram

#### B. ENTITIES

<i>Class: User</i>	
The User class represents both customer and staff entities. It is the base class for all user information. This class is responsible for managing the user personal information as well as defining the formatting of names.	
<i>Attributes/Fields</i>	<i>Description</i>
User ID	Unique identifier for a single User instance
FirstName	The first name of the user
LastName	The last name of the user

Email	A string representing a valid email address in the form of [account]@[domain].[suffix]
Username	A unique string chosen by the user to gain access to the system
Password	A string representing the password selected by the user to gain access to the Web site. It should be no less than 8 characters
UserType	An association to the UserType entity. It defines the type of the current user instance
<b>Relationships</b>	
UserType	Each user has a one to one relationship with a type entity.
HashingProvider	The User object will depend on the HashingProvider entity, which will hash the passwords for the current user object.

#### ***Class: UserType***

Entity that identifies one of the possible types of users. Possible options might be “customer”, “admin”, “guest”

<b><i>Attributes/Fields</i></b>	<b><i>Description</i></b>
UserTypeID	Unique identifier for a single UserType instance
TypeName	A string identifier for a single UserType instance
TypeDescription	A string identifier for a single UserType instance that provides a description for readers of the database.
<b>Relationships</b>	
User	Each UserType has a 1 to 1 relationship with User

#### ***Class: HashingProvider***

This object will be responsible for hashing the user passwords using one of the standard algorithm. The “provider” will depend on a third party library for the hashing algorithm. HashingAlgorithms: This package is TBD based on hashing requirements.

<b><i>Attributes/Fields</i></b>	<b><i>Description</i></b>
algorithmType	TBD
HashString	TBD
<b>Relationships</b>	
User	Each Hashing Provider has a 1 to 1 relationship with User

## 4.2 MOVIE FACTS

### A. UML STATIC STRUCTURE CLASS DIAGRAM



Figure 16 Movie Facts UML Static Class Diagram

### B. ENTITIES

#### *Movie Titles:*

This class will hold information regarding movie titles provided by MovieGo. Information will be stored in this entity and users will be allowed to view info stored by clicking on a movie title. The entity will include movie title, year, user rating, etc.

#### *Movie Title Attributes /Fields*

<b>Movie_id</b>	Will be used by the database to search and find movie titles that the user may request.
<b>Movie_title</b>	The attribute that holds the name of the movie which the user would like to get information on.
<b>release_year</b>	The year that the selected movie title was released.
<b>Total_gross</b>	How much money the selected movie made in the box office
<b>Main_actor</b>	The main actor in the selected movie
<b>Num_awards</b>	How many awards the movie currently has
<b>User_rating</b>	The rating of the movie based on users ratings

#### *Relationships:*

A movie title or many movie titles may be rated by a user. The movie title id will be carried over the rating entity and from there users may rate movies based on a percentage scale. The business rules stated at the beginning of the document state that users may only rate the movie once due to keeping the integrity of MovieGo and its affiliates.

### *Sort:*

This entity will oversee holding information on different sorting's available on MovieGo. This information can then be used to produce results for sorting's users wish to view.

#### *Sort Attributes / Fields*

<b>Sort_id</b>	Each sorting criteria will be unique and will be issued an id number.
<b>Sorting_name</b>	This attribute will hold the name of the sorting criteria.

### *Relationships*

The sort entity will be its own class and fetch queries that return movie lists with the correct sort. This entity will not require any movie information and will just return a list with movie titles.

### *Rate:*

This entity will hold the rating of the movie and will also hold the movie id so each movie may have a unique rating. This entity may also have a function which allow new users to rate movies and then see their input be updated with the movie rating.

#### *Rate Attributes / Fields*

<b>Movie_id</b>	A unique id which corresponds to a single movie title
<b>User_rating</b>	This will be a number which reflects the current rating of the selected movie

### 4.3 SEARCH

#### A. UML STATIC STRUCTURE CLASS DIAGRAM

#### B. ENTITIES

<i>Class:</i>	
<i>Attributes/Fields</i>	<i>Description</i>
<i>Relationships</i>	

## 4.4 DISCUSSION BOARD

### A. UML STATIC STRUCTURE CLASS DIAGRAM

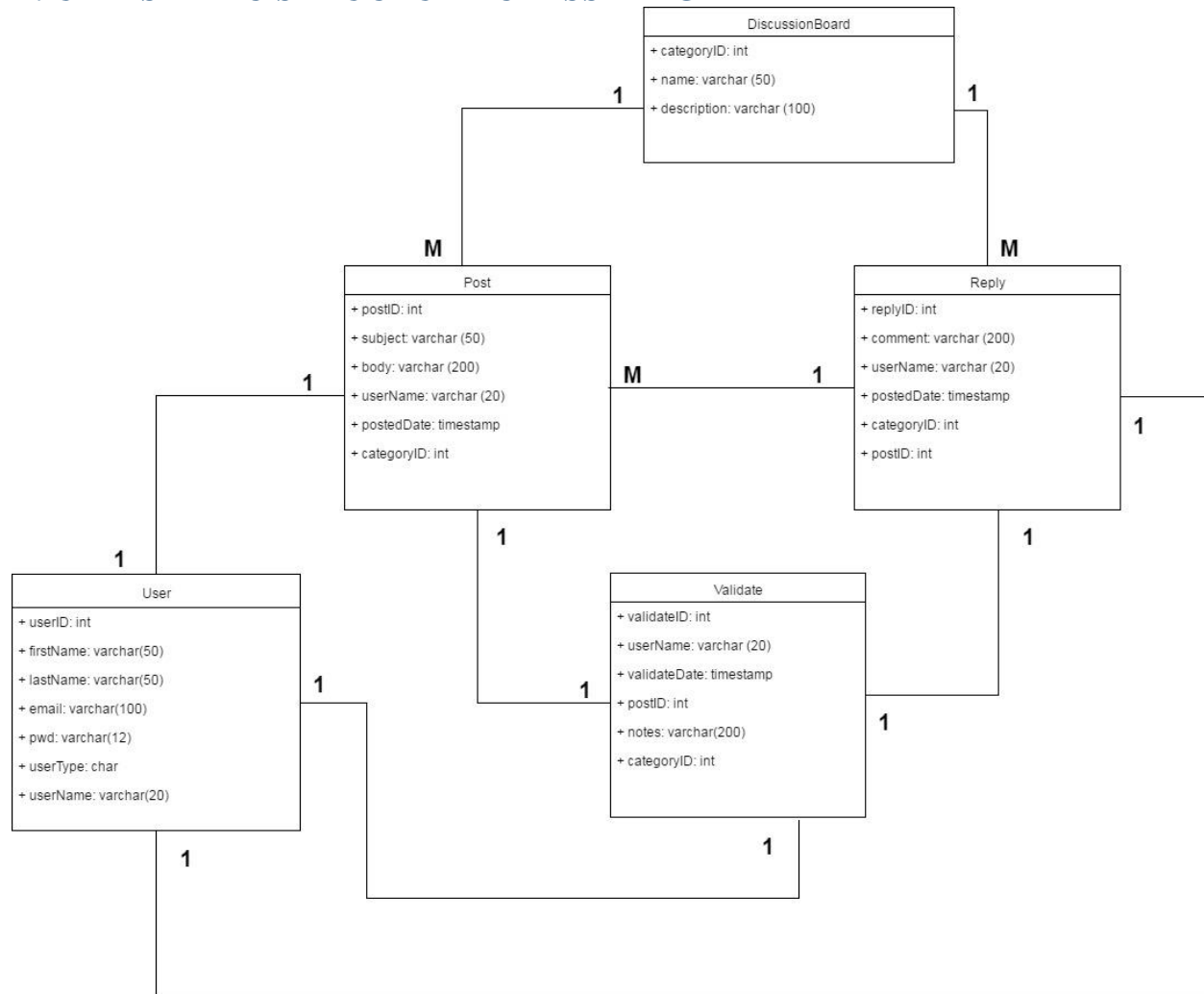


Figure 17 Discussion Board UML Static Class Diagram

### B. ENTITIES

*Class: User*

[See Account UML](#)

*Class: Discussion Board*

This is the super class for the Discussion Board module.

Entity that identifies the two possible categories: "Q&A" & "Movie Recommendation".

Attributes/Fields	Description
categoryID	Unique number identifier for a single category instance
name	A unique string representation of the name of the categoryID

description	Detailed description of the category to be used as a reference for other viewers/users of the database ie DB designers, DB admin etc.
<b>Relationships</b>	
Post	Each post can only be under one category A category can have many posts
Reply	Each reply can only be under one category A category can have many reply posts

#### **Class: Post**

Entity that is responsible for managing posts

<b>Attributes/Fields</b>	<b>Description</b>
postID	A unique number identifier for a single post instance, created by the database
subject	A string representing the title of the post, created by the user
body	A string representing the body of the post, created by the user
userName	A string representing the title of the post, created by the user
postedDate	Date type representing the creation date of the post
categoryID	Unique number identifier for a single category instance
<b>Relationships</b>	
Discussion Board	Each post can only be under one category A category can have many posts
Reply	A post can have many reply posts
User	Each post can only be under one username

#### **Class: Reply**

Entity that is responsible for managing replies to posts

<b>Attributes/Fields</b>	<b>Description</b>
replyID	A unique number identifier for a single post instance, created by the database
comment	A string representing the title of the post, created by the user
userName	A string representing the title of the post, created by the user
postedDate	Date type representing the creation date of the post
categoryID	Unique number identifier for a single category instance
postID	Number identifier for the parent post
<b>Relationships</b>	
Discussion Board	Each post can only be under one category A category can have many posts
Reply	A post can have many reply posts
User	Each post can only be under one username

#### **Class: Validate**

Entity that is responsible for validating posts and replies to posts

<b>Attributes/Fields</b>	<b>Description</b>
validateID	A unique number identifier for a single validate instance, created by the

	database
userName	A string representing the title of the post, created by the user
validateDate	Date type representing the creation date of the post
categoryID	Unique number identifier for a single category instance
postID	Number identifier for the post being validated
notes	String representing comments from validators
<i>Relationships</i>	
Discussion	Each post can only be under one category
Board	A category can have many posts
Reply	A post can have many reply posts
User	Each post can only be under one username



## 5. DATA DESIGN

This section provides a detailed description of the data to be represented in the database. A full database model is provided in Section 5.3.

### 5.1 DATA DESCRIPTION

#### A. REGISTRATION / AUTHENTICATION

From the below diagram, we identified a relationship between a user and his/her type. The possible types will be Administrator and Customer. The relationship “IsType” does not include any specific attributes.

Another relationship for the user is the Game Reviews. In this particular relationship, we have unique attributes that are not part of the user and neither are they part of the game. Those attributes are only related to the user review of a game. For example, the Review Description and the When Reviewed attributes are not part of the user, but they do describe the relationship of the user and game through a review.

Please include in the diagram a numeric quantity on the relationships.

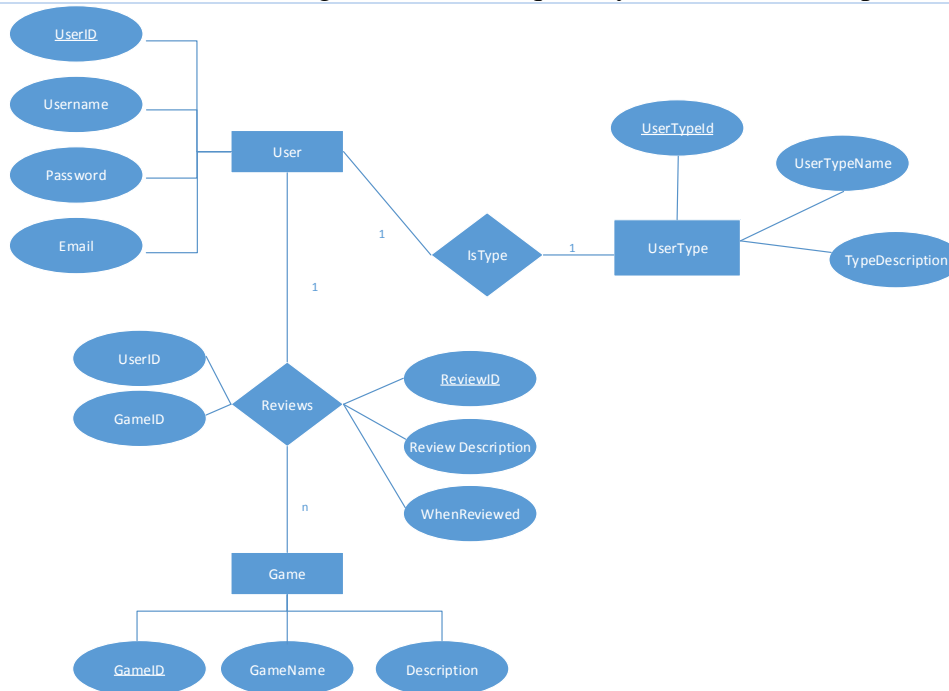


Figure 18 Registration/Authentication ERD Diagram

---

## B. MOVIE FACTS

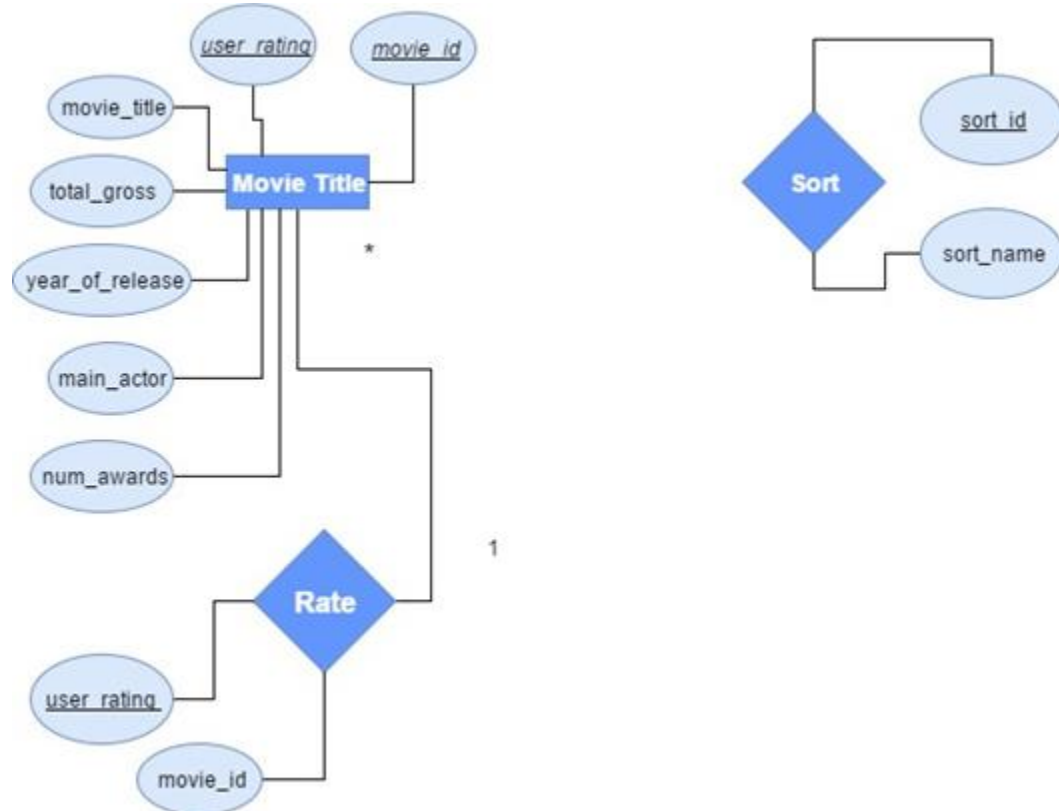


Figure 19 Movie Facts ERD Diagram

The diagram above illustrates the different relationships in the movie facts module. The sort entity is left alone because the sort will occur when a user hits on a html button. The sort entity will carry the sort name and the sort id to show users the different types of sort options they have to their disposal. The movie title entity will be home to the list of all movie titles produced by MovieGo. The DB will have attributes such as movie title, year of release, and total gross income made from the movie. The rate entity will hold the rating for the selected movie and each rating is unique to one movie. A movie may have one and only one rating provided by users. Ratings can be issued to one or many movies in the move list, provided by MovieGo.

---

**C. SEARCH**


## D. DISCUSSION BOARD

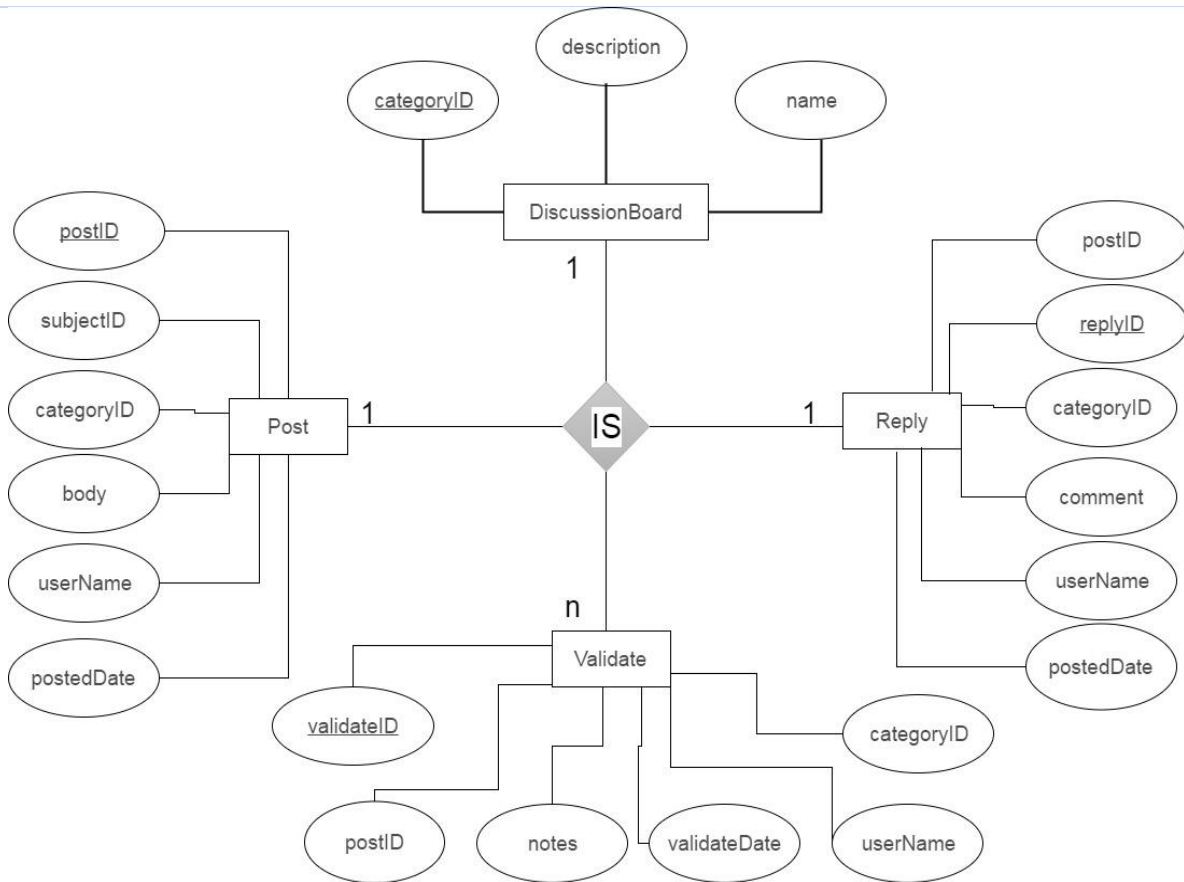


Figure 20 Discussion Board ERD Diagram

The above diagram depicts the relationship between the different entities of the Discussion Board module: DiscussionBoard, Post, Reply, and Validate.

The Post, Reply, and Validate entities all derive from the DiscussionBoard entity and will have categoryID.

The Reply entity will have the postID which is derived from the Post entity. This will identify the parent post to which the reply belongs to.

The Validate entity will have the postID which will identify which posts/replies are being validated.

All of the entities will have a username attribute which is derived from the User entity shown in the data description for the [Registration/Authentication](#).

## 5.2 DATABASE DICTIONARY / SCHEMA

### A. REGISTRATION / AUTHENTICATION

The following tables are defined for the Registration/Authentication module

*Table: User*

<i>Entity</i>	<i>Type</i>	<i>Value</i>	<i>Description</i>
<b>user_id</b>	int	Not null	PK. Unique identifier of a user.
<b>user_type_id</b>	int	Not null	FK. Relationship key to the User_Types table
<b>user_pw</b>	varchar(50)	Not null	User's password HASH
<b>first_name</b>	varchar(30)	Not null	User's first name
<b>last_name</b>	varchar(30)	Not null	User's last name
<b>DOB</b>	Date	Not null	User's date of birth
<b>Phone</b>	varchar(10)	Not null	User's phone number
<b>Email</b>	varchar(50)	Not null	User's E-mail address
<b>Address</b>	varchar(50)	Not null	User's home address
<b>City</b>	varchar(20)	Not null	City
<b>State</b>	varchar(2)	Not null	State
<b>Zip_code</b>	varchar(5)	Not null	Zip code

*Table: UserType*

<i>Entity</i>	<i>Type</i>	<i>Value</i>	<i>Description</i>
<b>user_type_id</b>	INT	Not Null	PK. Unique identifier of a user.
<b>type_name</b>	varchar(50)	Not null	The name of the user type
<b>type_description</b>	varchar(50)	Not null	Description of the user type and its uses

---

## B. MOVIE FACTS

The following tables are defined for the Movie Facts Module

<i>Movie Titles Table</i>			
<i>Entity</i>	<i>Type</i>	<i>Value</i>	<i>Description</i>
<b>Movie_id</b>	INT	Not Null	Id for unique movie title (PK)
<b>Movie_title</b>	VARCHAR(80)	Not Null	The title of the movie
<b>Year_of_release</b>	DATE	Not Null	The year that the selected movie was released
<b>Main_actor</b>	VARCHAR(60)	Not Null	The name of the actor that starred in the movie
<b>Total_gross</b>	INT	Not Null	Amount of income made from the movie
<b>Num_awards</b>	INT	Not Null	Number of awards won
<b>User_rating</b>	INT	Not Null	The rating of the movie based on user feedback (FK)

<i>Sort Table</i>			
<i>Entity</i>	<i>Type</i>	<i>Value</i>	<i>Description</i>
<b>Sort_id</b>	INT	Not Null	Id for unique sorting technique chosen by user (PK)
<b>Sort_name</b>	VARCHAR(20)	Not Null	Name of the selected sorting technique

<i>Rate Table</i>			
<i>Entity</i>	<i>Type</i>	<i>Value</i>	<i>Description</i>
<b>User_rating</b>	INT	Not Null	Rating given to selected movie by the user (PK)
<b>Movie_id</b>	INT	Not Null	The unique id given to the selected movie title (FK)

---

## C. SEARCH

The following tables are defined for the Search module

*Table:*

<i>Entity</i>	<i>Type</i>	<i>Value</i>	<i>Description</i>

---

## D. DISCUSSION BARD

The following tables are defined for the Discussion Board module.

<i>Table: Discussion Board</i>			
<i>Entity</i>	<i>Type</i>	<i>Value</i>	<i>Description</i>
<b>categoryID</b>	int	Not null	PK. Unique identifier for each instance of category
<b>name</b>	varchar(50)	Not null	Name of the category (Q&A and Movie Recommendation)
<b>description</b>	varchar(200)	Not null	Describes the name of the category in detail.

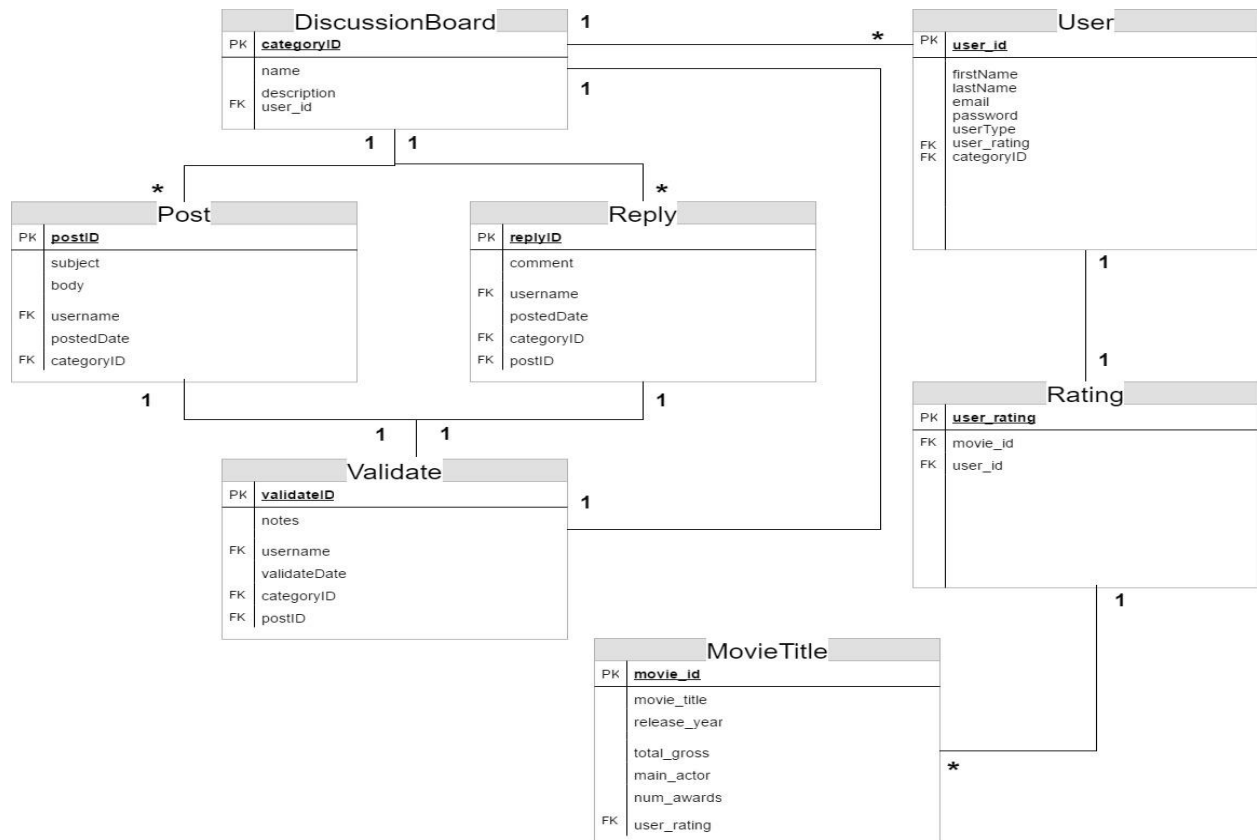
<i>Table: Post</i>			
<i>Entity</i>	<i>Type</i>	<i>Value</i>	<i>Description</i>
<b>postID</b>	int	Not null	PK. Unique identifier for each instance of post automatically generated by the database
<b>subject</b>	varchar(50)	Not null	User provided subject of the post
<b>body</b>	varchar(200)	Not null	User provided body of the post
<b>username</b>	varchar(20)	Not null	Unique name of user used in logging in FK. Relationship Key to User table
<b>postedDate</b>	timestamp	Not null	Date the post was created
<b>categoryID</b>	int	Not null	FK. Relationship Key to DiscussionBoard table

<i>Table: Reply</i>			
<i>Entity</i>	<i>Type</i>	<i>Value</i>	<i>Description</i>
<b>replyID</b>	int	Not null	PK. Unique identifier for each instance of post automatically generated by the database
<b>comment</b>	varchar(200)	Not null	User provided comment for the post
<b>username</b>	varchar(20)	Not null	Unique name of user used in logging in FK. Relationship Key to User table
<b>postedDate</b>	timestamp	Not null	Date the post was created
<b>categoryID</b>	int	Not null	FK. Relationship key to DiscussionBoard table
<b>postID</b>	int	Not null	FK. Relationship key to Post table

<i>Table: Validate</i>			
<i>Entity</i>	<i>Type</i>	<i>Value</i>	<i>Description</i>
<b>validateID</b>	int	Not null	PK. Unique identifier for each instance of validate automatically generated by the database
<b>notes</b>	varchar(200)	Not null	Admin notes on validation
<b>username</b>	varchar(20)	Not null	Unique name of admin validating the post FK. Relationship Key to User table
<b>validateDate</b>	timestamp	Not null	Date the post was created
<b>categoryID</b>	int	Not null	FK. Relationship key to DiscussionBoard table
<b>postID</b>	int	Not null	FK. Relationship key to Post table



## 5.3 FULL DATABASE MODEL



## 6. REQUIREMENTS MATRIX

<i>Module</i>	<i>Use Case</i>	<i>Design Component</i>
<b>Registration</b>		
	Authentication	User Interface: Section 3.1 A2 ERD Diagram: Section 5.1 A Component Model: Section 4.1 DB Schema: Section 5.2 A
	Change Password	User Interface: Section 3.1 B2 ERD Diagram Section 5.1 A Component Model: Section 4.1 DB Schema: Section 5.2 A
<b>Movie Facts</b>		
	Select Movie	User Interface: Movie List Section 3.2 A2 ERD Diagram Section 5.1 B Component Model: Movie Facts Section 4.2 A DB Schema: Section 5.2 B
	Sort Movie	User Interface: Movie List w/ sort selected Section 3.2 B2 ERD Diagram Section 5.1 B Component Model: Movie Facts Section 4.2 A DB Schema: Section 5.2 B
	Rate	User Interface: Section 3.2 C2 Section 3.2 C2 ERD Diagram: Section 5.1 B Component Model: Movie Facts Section 4.2 A Component Model: Section 4.2 DB Schema: Section 5.2 B
<b>Search</b>		
<b>Discussion Board</b>		
	Post	User Interface: Section 3.4 A2 ERD Diagram: Section 5.1 D Component Model: Section 4.4 DB Schema: Section 5.2 D
	Reply	User Interface: Section 3.4 B2 ERD Diagram: Section 5.1 D Component Model: Section 4.4 DB Schema: Section 5.2 D
	Validate	User Interface: Section 3.4 C2 ERD Diagram: Section 5.1 D Component Model: Section 4.4 DB Schema: Section 5.2 D

