**Contents**

1. Defining the problem

2. Investigation and analysis

i) Interviews & Questionnaires  
 a) Interview Plan (administrator)  
 b) Interview (administrator)  
 c) Interview Plan (client relationship employee)  
 d) Interview (client Relationship Employee)  
 e) Employee (questionnaire)  
 ii) Analyzing the Present System  
 iii) Software & Hardware Specifications  
 a) Software Requirements  
 b) Hardware Requirements (minimum)

3. Nature of solution

i) Design specification  
 a) List of objectives  
 b) Designing the input and output  
 ii) Designing the Data  
 iii) Code Design  
 iv) Test Plan

4. Intended Benefits

i) Feasibility Status

5. Limits of the Scope of the Solution

6. Development

i) Data Structures  
 ii) Input and Output Effectiveness and Changes in Database  
 a) Logging In  
 b) Adding a New Item to the Program  
 c) Listing, modifying, or editing an item  
 d) Selling a product, renting a movie and pushing its deadline  
 e) Sales Report  
 f) Message/E-Mail to a Member  
 g) Help  
 h) Guest Panel

7. Programming

i) How to Use the ModerUI for the WPF Application  
 ii) Original Code  
 a) Logging-in (Administrator, Employee, Guest)  
 b) Adding a new item to the program  
 c) Listing/Modifying/Deleting an item  
 d) Selling an item  
 e) Rental process  
 f) Sales report  
 g) Email a member  
 h) Guest panel  
 i) Help  
 j) Logging out  
 iii) Creating the Setup for the Program

8. Testing

i) Alpha testing  
 a) Adding a product  
 b) Listing a movie  
 c) Modifying an item in the database  
 d) Deleting a movie  
 e) Authentication security  
 f) Selling a product  
 g) Rental process  
 ii) Beta testing

9. Installation

i) Planning the installation  
 ii) Training the staff  
 iii) Strategy for implementation

10. Systems Maintenance Documentation  
  
 i) Hardware and Software Specifications  
 a) Software Requirements  
 b) Hardware Requirements (minimum)  
 ii) Technologies used  
 iii) Original programming used in the new system

11. User Guide  
  
 i) Introduction  
 ii) Set-Up Guide  
 a) Logging In  
 b) Option 1: Adding something new to the program  
 c) Option 2: Showing a list of movies, games, members or employees  
 d) Option 3: Modifying or deleting one of the listed items (movies, games,  
 members, employees)  
 e) Selling a Product  
 f) Renting a Movie  
 g) Viewing Sales  
 h) Generating a PDF file of the sales  
 i) Guest  
 j) Help  
 k) Logging Out  
 iii) Troubleshooting Guide  
 a) Logging In  
 b) Adding something to the program  
 c) Modifying/Deleting something from the program  
 d) No sales are shown to the sales report of the administrator  
 e) Program crushes

12. Evaluation

i) Success in meeting the original objectives  
 ii) The user’s response to the system

13. Glossary

**1. Defining the problem**

“Video Store” is a private business created in 2000, which sells video games for different platforms (new or used) and rents and sells movies. Clients of this business can become members of this store to get a discount in their purchases and get informed for the new products of the store.

The owner of the “Video store” (client) presented to me (system analyst) his problem, which consists in computerizing some of the procedures and works done by the administrator and employees of this business. The problem was that all the above mentioned people did their job manually, keeping every information needed in hard copy without using a system of controlling these system. The client asked for computer software that functions as a stock control, calculates the brute income and can also be used from a terminal in the store from the clients as a content of the store products. According to our client this software should also have the function to add new employees, members, products and keep a record of every transaction made by the store and its clients.

Until now the employees of the store used papers to keep a record of every product of the store. These papers consisted of the games sold, movies sold/rent, all the members of the store, and the stock of each product. Then the administrator took all the bills of the products bought and sold and did the calculations himself to get a brute and net income.

In general, the owner of the store requested for a computer application that could do all the above mentioned procedures with a little use of a human (employee or administrator).

Some of the data that is going to be stored in the application software already exist in form of spreadsheets in the current PC that the owner of the store uses. Some of the data will be stored as string such as names, game or move title etc. Other type of data that is going to be used is integer for cost of products to calculate the income. Date/Time data may come in handy in case of the rental process.

**2. Investigation and analysis**

**i) Interviews & questionnaires**

After arranging a meeting with the client I prepared some questions on which I’m going to base most of this conversation. Of course there will also be follow up questions depending on the context of the conversation and the answers of the client.

**a) Interview plan (administrator)**

1. Would you like to have a computerized system to manage the workflow of business?
2. Can you give me a brief description of the business?
3. How many employees do you have for the moment?
4. Do all of your employees do the same thing or not?
5. How can we help each one of them?
6. When you hire a new employee, what information do you need?
7. How do you keep track of your sales?
8. Can the employees see each other sales?
9. Does it sound reasonable that every employee has its own account to keep track of his sales?
10. What data do you need for film and games?
11. What procedure do you follow for every new product to your store?
12. Where do you store used games?
13. How do you manage the rental process of a film?
14. Would you like to see a resultant of the sales at the end of the month?
15. How many computers do you have?
16. Now do you have any requirements for the date of software to be ready?

The following is the interview with the administrator.

**b) Interview (administrator)**

A: Would you like to have a computerized system to manage the work flow of business?

B: Yes. The current work that I do is very tiring, and this system would really help me to save some time and to manage some of the work.

A: Can you give me a brief description of the business?

B: Well, my business consists of a small store that sells movies and games and those who become members of our store can rent movies. We also buy used games and re-sell them.

A: Only members can rent your movies?

B: Yes, only members, because we need to control them. We have their mobile phone number and their e-mail address in any case.

A: And, when are these phone numbers and e-mail addresses stored?

B: Oh, we write them down in spreadsheets manually.

A: I’m guessing you want to email or message to those members automatically for any case?

B: Yeah, you guessed it right. We always email or message our members one day after their rental deadline has passed. This is kind of hard to text each one of them manually.

A: Well for that I’ll need the e-mail address and password and the phone number that you use for contacting the clients

B: Here, I’ll write it down for you. (e-mail: video.store2000@yahoo.com; password: 17May2000). This is the e-mail address and the password. The phone number is the one that we’ve contacted.

A: Ok. Since we started talking about members, can you tell me a little bit more about them?

B: Yeah, sure. Members are our special clients. A client comes to our store and he has to register to become a member. Becoming a member gives him some benefits because he can rent movies or earn discounts on our products.

A: Can you tell me a little bit more about the registering? What type of records do you need to save for the members?

B: The clients that want to register need to come with an ID or a passport. After we verify the validity of the document, we save their first and last name, their age and gender.

A: What about the rent? Are there any restrictions about the date?

B: Yes. The members should return the movies within 3 days of taking it from the store.

A: What about the amount of movies that a member can rent at a time? Is there any restriction about that?

B: Yes. The member can only rent five movies at a time.

A: You mentioned earlier about a discount for members? Can you talk a little bit more about that?

B: Of course. For every 10 films that a member buys he takes 1 free. Also if the client has been a member for more than 1 year he gets a 10% discount in every buy.

A: Does this discount apply to games too?

B: Yes, the same thing applies to games too.

A: Do you have anything else to add about the members?

B: No, not for the moment.

A: Ok, then let’s proceed to the next question. So, is this a personal business or do you have any employees?

B: No, it’s not a personal business. I also have other employees.

A: How many employees do you have for the moment?

B: There are 4 employees in total. They work in shifts two by two, morning and afternoon.

A: Do all of your employees do the same thing or not?

B: One of them works as a receptionists and the other helps the clients or regulates the shelves.

A: How can we help each one of them?

B: Well the receptionist works with the lists of movies and games. He keeps them in spreadsheets and modifies them manually. The other works with the clients.

A: Can the receptionist register new members?

B: Yes. He can also penalize members or contact them.

A: Does the morning receptionist and the afternoon receptionist have the right to access the same data?

B: I don’t understand the question.

A: Can the members registered by one of the receptionist be accessed by the other one?

B: Yes, of course. The same spreadsheet is used for all the members. I, too can access those members.

A: Ok, than. When you hire a new employee what information do you need?

B: Well their first and last name, the date of employment, age and of course the salary.

A: Do you have any penalties or bonuses for your employees?

B: No, no penalties but bonuses yes.

A: What exactly do you mean by bonus?

B: Well, for every 200 products sold by an employer he gets 10% of the wage as a bonus.

A: Ok, let us proceed to the next question. How do you keep track of your sales?

B: Like every other document, in spreadsheets. The employees know how to use the formulas of Microsoft Excel for any calculations needed.

A: Is it a little bit of hard, using all these spreadsheets?

B: Yes it is. Especially when the employees do not use them correctly. That’s why I’m asking for your help.

A: What about the sales, can the employees see each other sales?

B: No, every employee has its file saved by with a password.

A: Does it sound reasonable that every employee has its own account to keep track of his sales? An account like Facebook or G-mail.

B: Ok. It sounds great. But there’s one thing. I’d like to have the right of creating accounts, like an administrator.

A: No problem. We could also add you the right to modify and delete these accounts.

B: Yes, that would be great.

A: Anything else to add for the employees?

B: No, I don’t have anything for the moment. If I recall something I’ll contact you later.

A: Ok, then. Based on the description of your business, I heard you sell films and games? What data do you need for each product?

B: Well it’s the title, year and genre. I think that there’s more but I don’t remember for the moment.

A: It’s ok. We plan to talk to the employees for extra information.

B: Ok.

A: What procedure do you follow for every new product to your store?

B: After we buy the product, we add it to a spreadsheet with all its information.

A: Based on your description you also sold used games. You store those on a different spreadsheet?

B: No, there’s an extra field, used or not.

A: What about the rental process of a film? How do you manage the situation?

B: We keep another spreadsheet for this procedure. We write the name of the member, film title, date of taking the film and the date of bringing it back. Sometimes members pay extra money, which is 50 lek, to postpone the deadline. It’s a little bit of itchy situation because you have to follow all of the members that have rented a film and we have to contact them one day before the deadline.

A: How much is the price for renting a movie?

B: It’s always a fixed cost. You pay 100 lek and return the movie after 3 days.

A: Ok. Would you like to see a resultant of the sales at the end of the month?

B: Yes. I’d like the total turnover and the brute income.

A: I’ll see if it’s feasible. Ok, now, to the technical part. How many computers do you have?

B: Only one, at the receptionist.

A: What type of operating system do you use?

B: Windows 7

A: Do you have any printers?

B: Yes.

A: Alright then. I think I’ve gathered enough information about the software. Now do you have any requirements for the date of software to be ready?

B: As soon as possible, but I want it before the end of March or beginning of April.

A: I’ll try my best. If I need anything else I’ll contact you. Goodbye.

B: Ok, goodbye.

**c) Interview plan (*client relationship employee*)**

1. Alright. Can you give me a brief description of your job?
2. How do you mange to do the list of the movies?
3. Where do you get this information about the movies?
4. Do you think of any computer solution that is suitable for you?

**d) Interview (*client relationship employee*)**

A: Hi, I’m the system analyst hired by the owner of the store to create an application software of computerizing some of the procedures needed for this business.

B: Yes, I’ve heard about this news.

A: Do you have time for an interview?

B: Yes, of course. I have some time.

A: Alright. Can you give me a brief description of your job?

B: Well today I’m on the shift of the employee that helps the client. I tell them about new games or movies, games trending now etc. I also orient them about the position of the games in the store.

A: How do you mange to do the list of the movies?

B: I take the movies from our spreadsheets, do an order of them according to dates, and stick a printed copy on the wall. I also stick a printed copy of the top rated films.

A: Same thing about games?

B: Yeah, the same thing. But for the games also a print of a copy of the most trending games for the moment.

A: Where do you get this information about the movies?

B: Most of it is information from IMDb.

A: And for the games, where do you take information.

B: For games I use IGN as a source.

A: Do you think of any computer solution that is suitable for you?

B: My job is kind of hard. If there’s an automatic way of presenting the films and games, it would be easier. There’s also difficulty in making the lists of the top rated films and games.

**e) Employee (*questionnaire*)**

*1. What does your job consist?*

Putting new information on the computer, such as information about games, films, sales and members.

*2. What information do you keep for films?*

Title, category, release date, runtime, rating, film description, number in stock, price and times sold.

*3. What information do you keep for games?*

Title, category, release date, producer, rating, game description, number in stock new and used, compatibility and times sold.

*4. Sales information about films and games:*

Films: film sold, member who bought it, sale date, price and cost   
Games: game sold, member who bought it, sale date, price and cost.

*5. Information needed for rented films?*

Film rented, member who rented it, date of renting it, date of return, price.

*6. What information do you keep about members?*

First name, last name, age, gender, registration date, films bought, games bought.

*7. How do you contact the member?*

E-mail or through mobile phone.

*8. Opinions of computerizing your work.*

Using a friendly interface; having my own control panel of sales; If feasible the option to contact clients.

**ii) Analyzing the present system**

1. Manual system
2. Use of spreadsheets
3. Information needed for films stored manually (title, category, release date, runtime, number in stock and rental availability)
4. Information needed for games stored manually (title, category, release date, producer, number in stock, compatibility, number in stock of used games)
5. Every sale/rent data is stored and calculated manually (price, cost of products, rental process of films, calculation of net and brute income)
6. DFD description
7. Problems:

a) A lot of time needed to store all the data. b) Bad management of the data because they are stored in different files. c) Risk of file corruption and the loss of them.  
 d) Difficulty in noting the data.

1. Solution: proposal of a computerized system of data management of the business, who requires the completion of the following list of objectives.

**Data Flow Diagrams**

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. Often they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).

A DFD shows what kind of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of processes, or information about whether processes will operate in sequence or in parallel (which is shown on a flowchart).

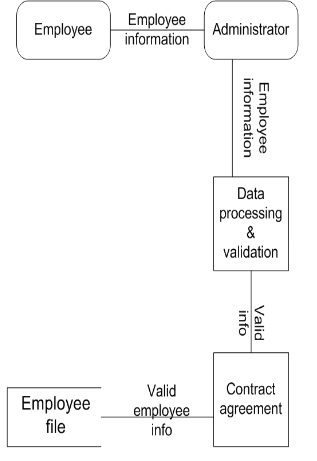
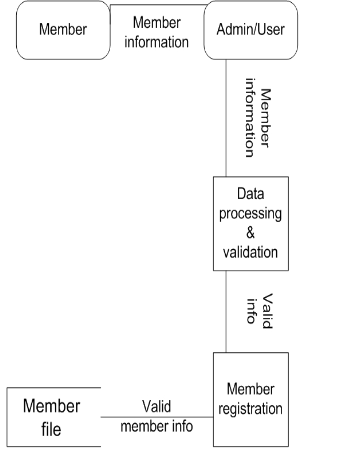
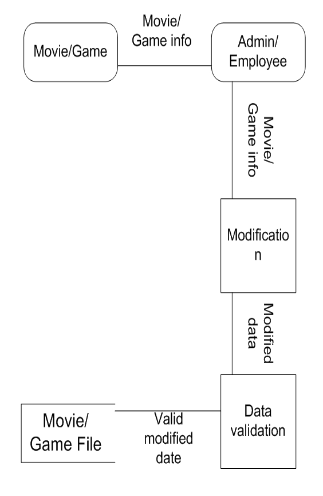
  

Fig. 2.1 Adding an employee Fig. 2.2 Adding a member Fig. 2.3 Adding a movie/game

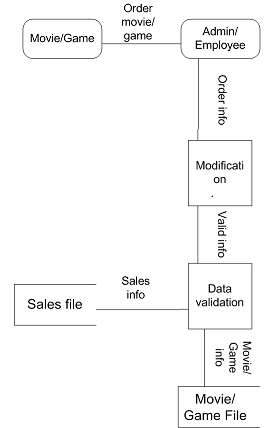
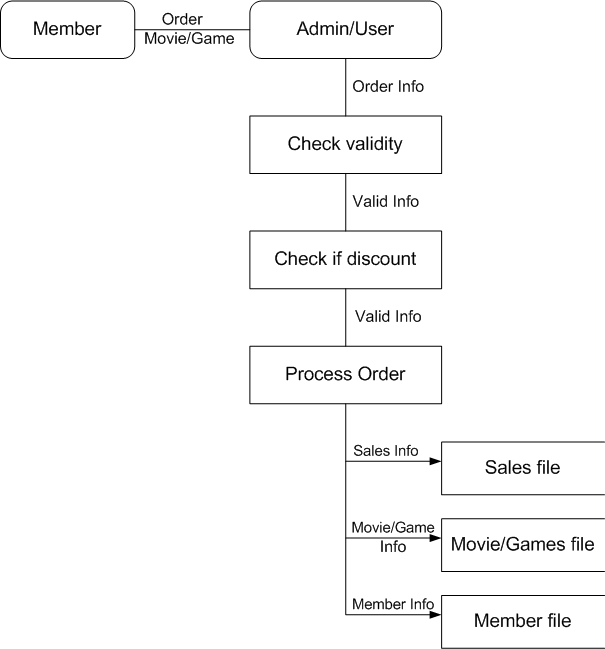
 

Fig. 2.4 Selling a product (not to a member) Fig 2.5 Selling a movie to a member

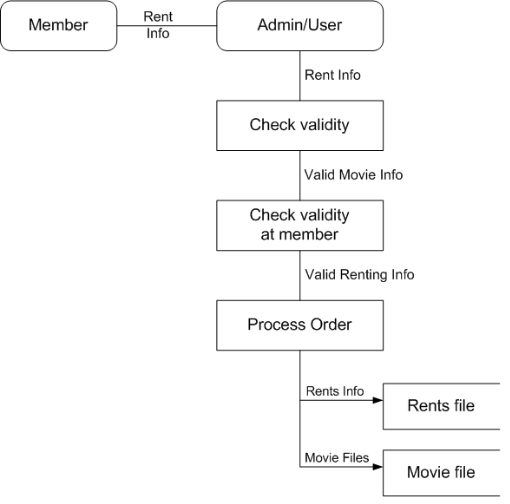
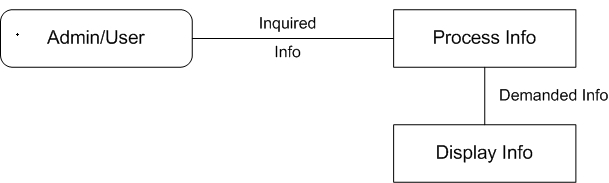
**** ****

Fig. 2.6 Rental process of a movie Fig. 2.7 Listing the demanded item

**iii) Software & hardware specifications**

**a) Software requirements:**

1. .NET Framework 3 (or later version) – the platform which allows the execution of the application

2. Microsoft SQL Server 2008 or later – RDBMS (Relational Database Management System) stores the data of the application in table format which are related to each other

3. Microsoft Windows Vista/7/8 – Operating system that serves as a platform for the execution of the application

**b) Hardware requirement (*minimum*):**

1. 2 PCs with:  
 a) Pentium 4; 3.0 GHz (or equivalent AMD processor) – to have a good user experience  
 b) 2 GB RAM DDRII – to have a good user experience  
 c) 500 GB HDD – to store the data   
 d) Network interface card – in order to connect to internet  
 e) Intel HD Graphics 4000 (or any other graphic card for better performance) (it’s optional)

2. 19” Flat Monitor x2 (touch: optional) -

3. Keyboard x2 - to input data

4. Mouse x2 - to navigate

5. Printer - to print hard copies of sales

6. CD-ROM – to install the program

7. Router or switch (wireless optional) – to connect the computers with each other and the internet.

**3. Nature of solution**

**i) Design specification**

**a) List of objectives:**

1. Realization of the graphical interface for the computerization of the work

2. Access of the application software in different levels form different users

3. Realization of an administrating panel with access only from the administrator with these functions:  
 a) Add, modify, delete employee  
 b) Add, modify, delete member  
 c) Add, modify, delete film/game  
 d) View all film/game sales  
 e) View all film rents  
 f) Accounting panel  
 g) Rent film  
 h) Sell film/game

4. Employee panel:  
 a) Add, modify, delete member  
 b) Add, modify, delete member  
 c) Add, modify, and delete film/game  
 d) Rent film  
 e) Sell film/game  
 f) View its sales

5. Member panel:  
 a) Top rated films/games  
 b) New films/games

**Use case diagrams**

A use case diagram at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well.

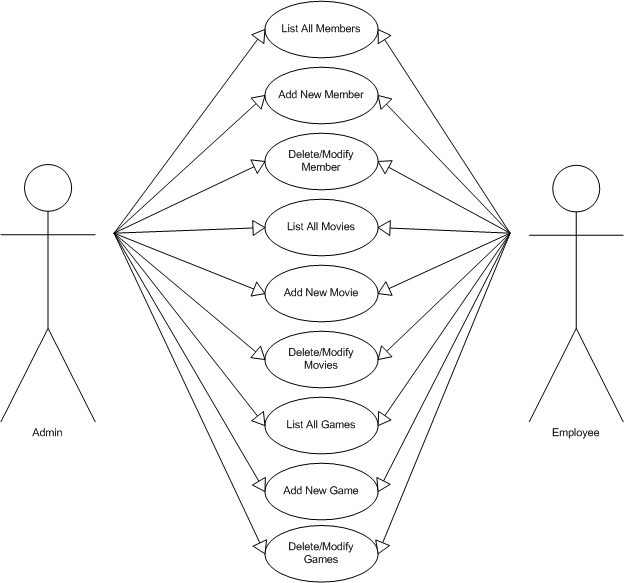


Fig. 3.2 Use case diagram for administrator and employee

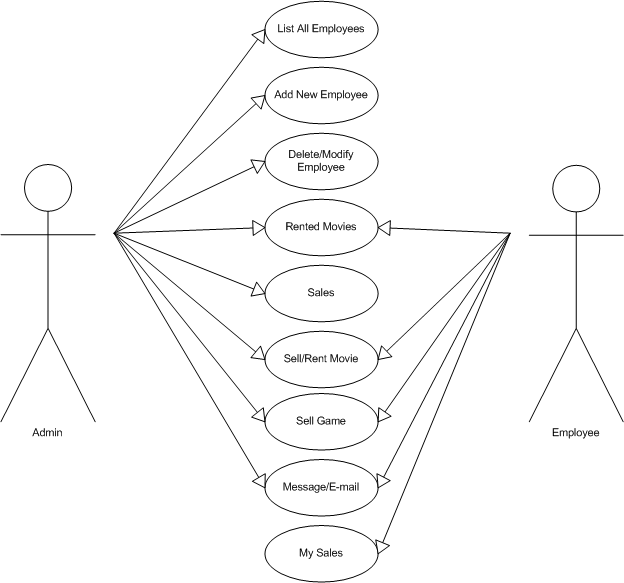
****

Fig. 3.3 Use case diagram for administrator and employee (continued)

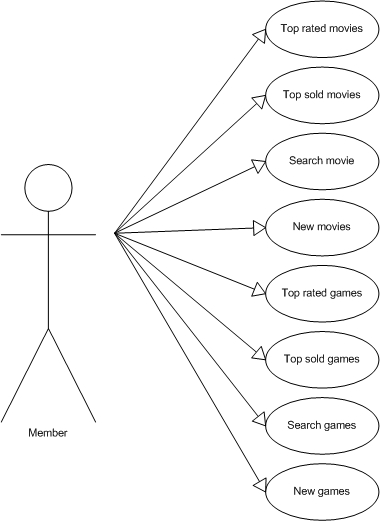
****

Fig. 3.4 Use case diagram for member

**b) Designing the input and output**

Based on the list of the objectives there are four essential functions (add, edit, delete, list) that this application software will provide to the user. As far as it concerns to the ‘add’ function this will be made in a form-based interface. ‘List’ function will give a table as an output that will list the desired item that the user has chosen. ‘Edit’ and ‘delete’ function will be built in a way that the user selects the desired item to be edited or deleted from a list and it can be modified through a form-based interface or deleted with a simple button.

The reason that a form-based interface is proposed, it is because of its many advantages that are:

* Easy to program
* Easy for user to see the options available
* Data validation can be used on data entry forms
* Fast to enter data or to make choices
* Little or no training required
* They don’t huge amounts of processing power or memory

This type of design is been presented to my client and he has agreed with this design of the program.

**ii) Designing the data**

**Entity relationship diagram**

An **entity–relationship model** (**ER model**) is a data model for describing the data or information aspects of a business domain or its process requirements, in an abstract way that lends itself to ultimately being implemented in a database such as a relational database. The main components of ER models are entities (things) and the relationships that can exist among them.

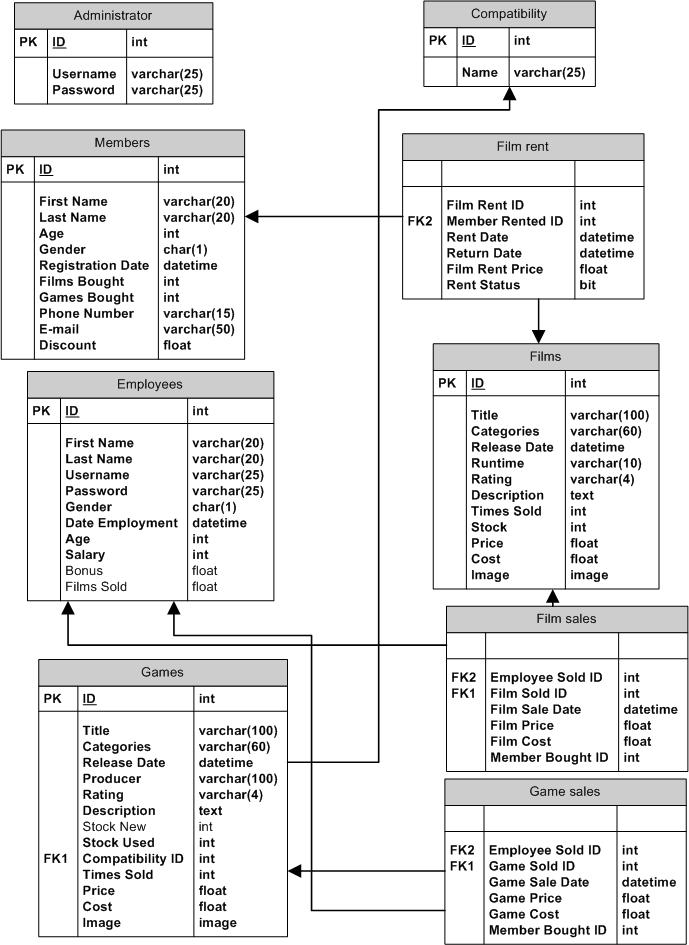
****

Fig. 3.1 Entity relationship diagram that represents the relationship between tables in database

**iii) Code design**

a) Good practices

In order to make the code clearer, more structured or easy to understand, some good programming techniques come in handy. These techniques and their advantages is presented below:

* Code annotation – it is written before the code so that the programmer (me) is clear about certain aspects of the design.
* Code indentation – it is used so that it’s easier to see how the code flows
* Modularizing the work – this technique is used not to rewrite the code many times
* Using meaningful names for variables, functions and classes – it’s easier to understand the code when you view it for a second time
* No use of ‘wizards’ – this helps to have full control on the software behavior

**iv) Test plan**

The mission of this test plan is to ensure that this program will achieve the final objective and to surpass all the obstacles along the way. The solution of this program is modularized and the test plan will consist of continuous testing after each completion of a module

To do this test, the required software and hardware are all of the hardware and software specifications mentioned above.

The first type of test is white box testing. After each part of code is written, it’ll be review in order of any mistakes. After the white box testing it’s complete that the black box testing starts. These two types of tasting are going to be made before the alpha testing.

During this test plan the areas that are going to be tested are the four basic functions that this program provides (add, edit, delete, list) of a single product. Other options that are going to be tested as well are the options to sell or rent a product and to make a hard copy of a list that’s provided by the program.

At the ‘add, edit, delete’ option we will test if the program communicates successfully with the database and if the changes are made correct. At the listing option we will see if the program reads and selects successfully from the database tables. We will also test the sell and rent option and see their effect on database table. The option to printout a list we will test it through a printer.

Another thing to be tested in this software is the authentication security. Because this program has more than one user with different levels of access, it’s essential to test this part of the program very carefully.

The following are to be tested:

* Adding a product
* Listing a product
* Modifying a product
* Deleting a product
* Authentication security
* Sell a product
* Rent a movie

**4. Intended benefits**

**i) Feasibility study**

After analyzing the list of objectives, there’s need for the feasibility status to see if the application software is beneficial for a long-term period or a short-term period. During the analysis of the feasibility status the following questions came up:

* *Is the solution technically possible?* The solution is technically possible because all of the tools needed for the realization of this new software exist.
* *Is the solution economic to produce?*

On the software part there’s no cost because it’s developed as a school project. The only cost is to buy a new computer

* *Is the solution economic to run?*

There’s no cost for this application software to run. Unless the business owner decides to buy the latest software needed for this application to run, which require extra payment.

* *What are the social effects?* No negative social effects, but just positive. It makes the employees’ job easier.

* *Is the workforce skilled enough?*

The application will be with a simple, user-friendly interface. Only skill needed is basic

computer knowledge.

* *What effect will there be on the costumer?*

The impact will be positive because the environment will look more technological. There’ll be a terminal for the client.

* *Will the introduction of the new system be economically beneficial in the long term?* Yes because there’s no extra cost. System will be robust and it will be able to store the data for a long time until the business does not need it anymore.

*After the conclusions we can say that the system is economic to produce and economic to run. The workforce is skilled enough to use it and the system will not have any negative impacts on neither the society nor the client. Therefore, we can say that this system is feasible to produce.*

**5. Limits of the scope of the solution**

The owner of the store already has a personal computer sufficient for executing a program that will be able to store data in database tables and modify or delete those data. The owner has also agreed to buy another computer that is going to be used as a terminal for the employee that handles with clients and for the clients too. As far as it concerns to the hardware there’s no problem that relates to the software application that it’s going to be created.

**6. Development**

**i) Data structures**

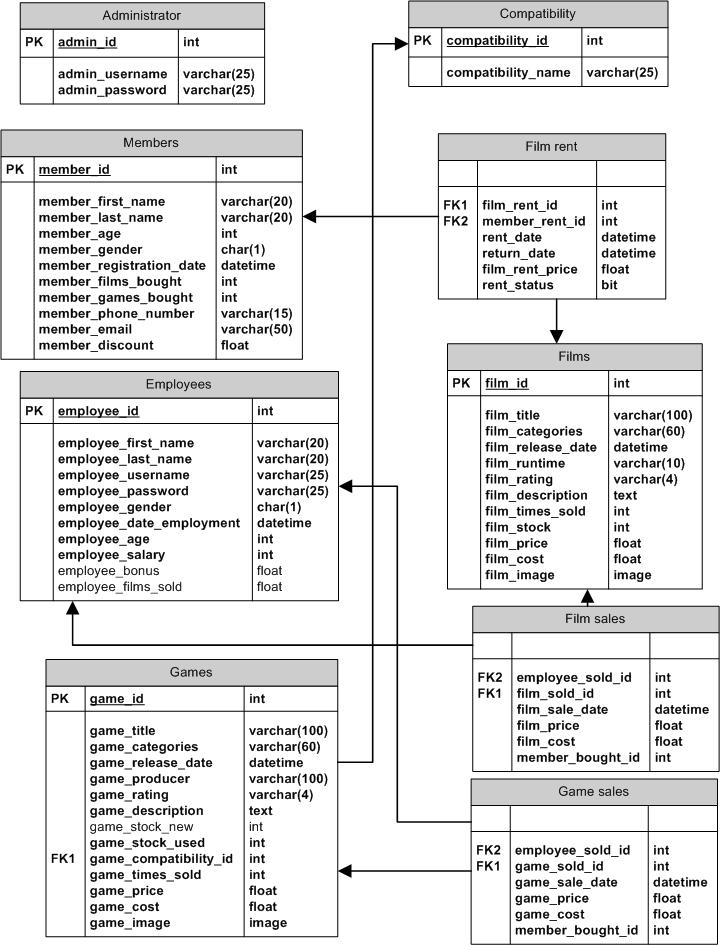
****

Fig. 6.1 Entity relationship that represents the relationship between the exact tables in database

**ii) Input and output effectiveness and changes in the database**

**a) Logging in**

Before the program is created in the database table is created an administrator with username = ‘admin’ and password = ‘admin’. The login is successful as an administrator (fig. 6.1) only with these credentials. If logged in is successful a new window is opened else if the username or the password is incorrect or you don’t complete at least one of the fields, then the program won’t let you proceed with the program (fig 6.2 and 6.3).

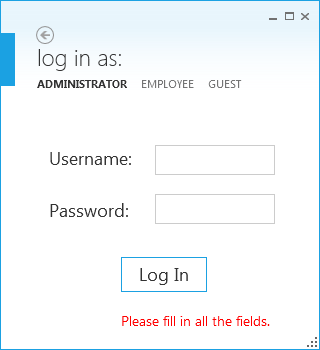
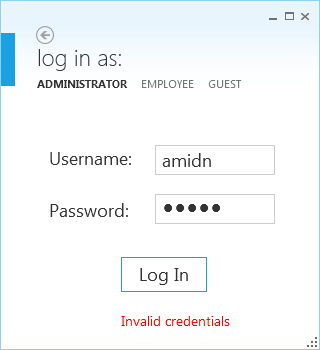
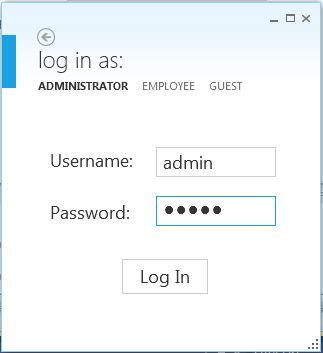


Fig. 6.1 Login window Fig. 6.2 Invalid credentials Fig 6.3 Not all the fields completed

Other ways of logging in are as an employee with the correct username and password which are provided to him from the administrator (this will be explained later on in further detail) or as a guest with no obstacle at all (fig. 6.4).

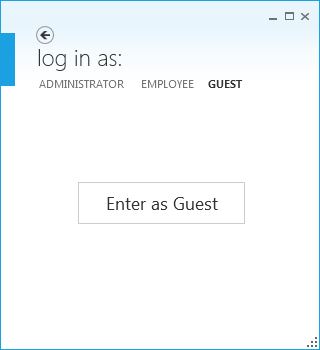


Fig 6.4 Logging in as a guest

**b) Adding a new item to the program**

Let’s take the example of adding a new game to the store. In the fig 6.5 is shown the table of the database with all the games in it. In the figure 6.6 is shown the process of filling all the necessary fields for adding a game (a more detailed explanation on how to add a game is given in the Section 11/ Adding a game/ page ).

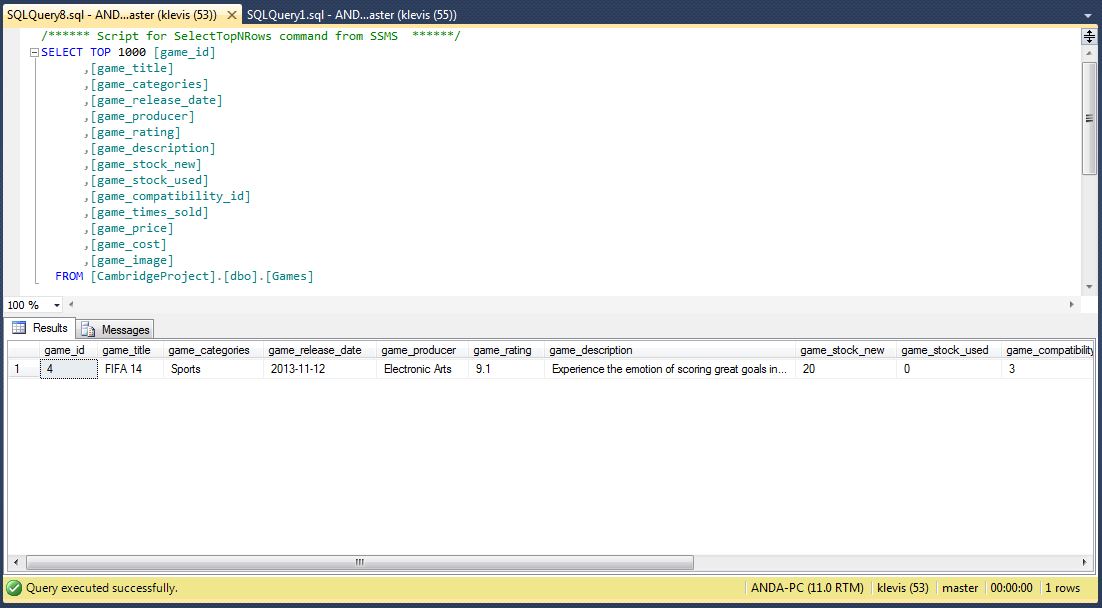


Fig 6.5 The database table before adding the game.

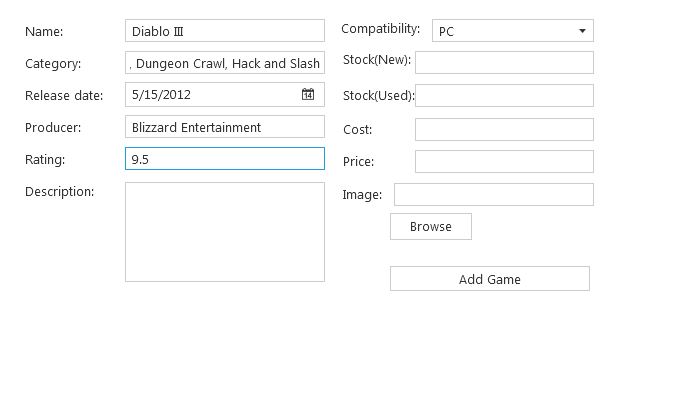


Fig 6.6 The process of adding a game

In the fig. 6.7 and 6.8 below is shown respectively the page with the added game and the change in the database table for the games.

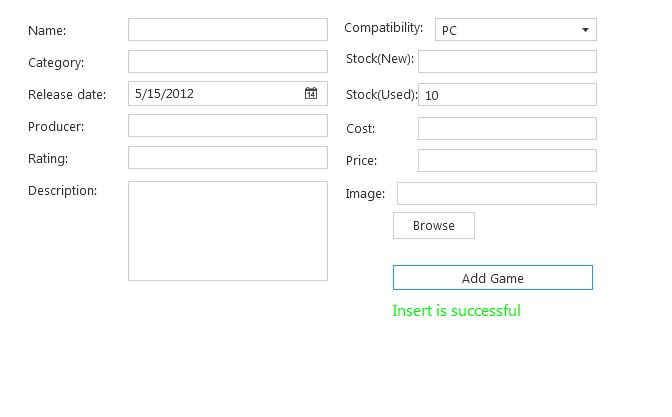


Fig. 6.7 The page after the game is added.

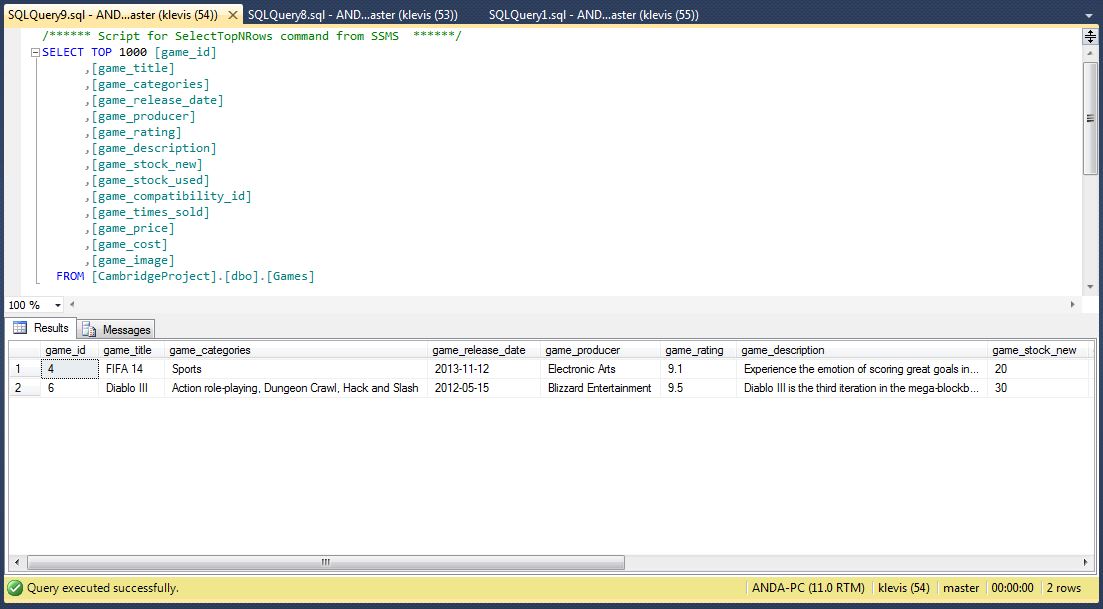


Fig 6.8 The new entry in the database table after the game is added in the program.

In the exact similar with the movies, members, compatibilities and employees are added to the program and the database tables.

Let’s have a look at the process of adding an employee. Fig. 6.9 shows the interface for adding an employee. When the administrator adds an employee he provides a username and a password for the employee added. Figure 6.10 and 6.11 shows that the employee added in the fig. 6.9, logged in successfully with those username and password. The username and the password can be changed but that will be explained later.

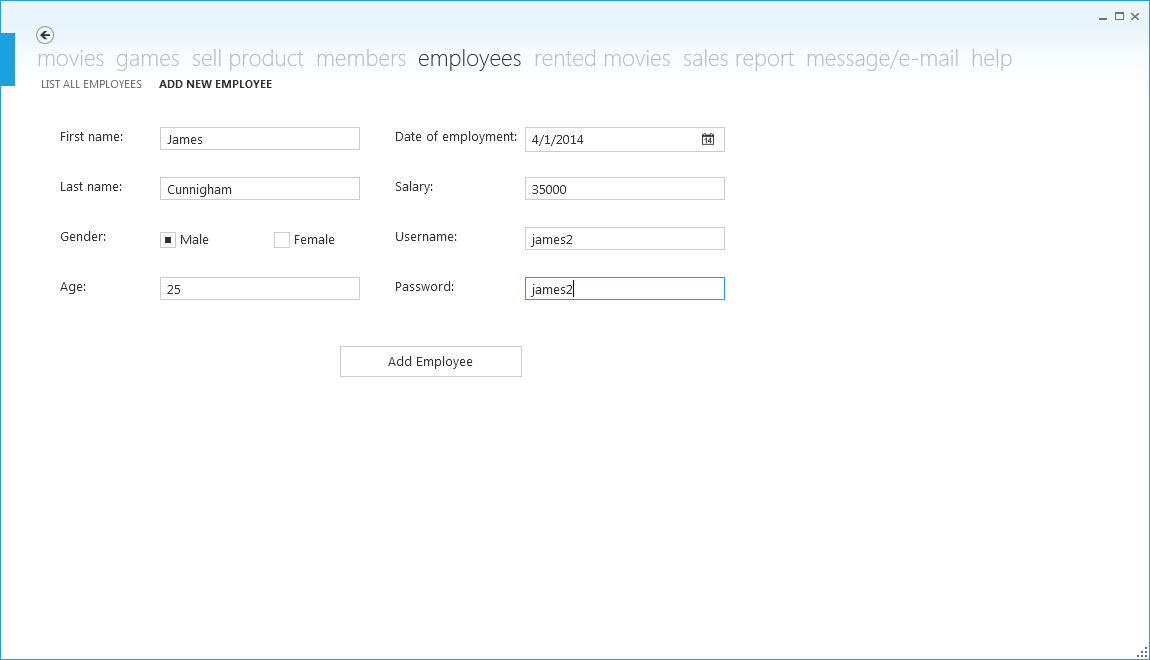


Fig. 6.9 The interface for adding a new employee.

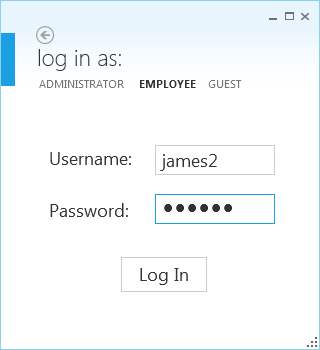


Fig. 6.10 Logging in at the employee panel with the new account.

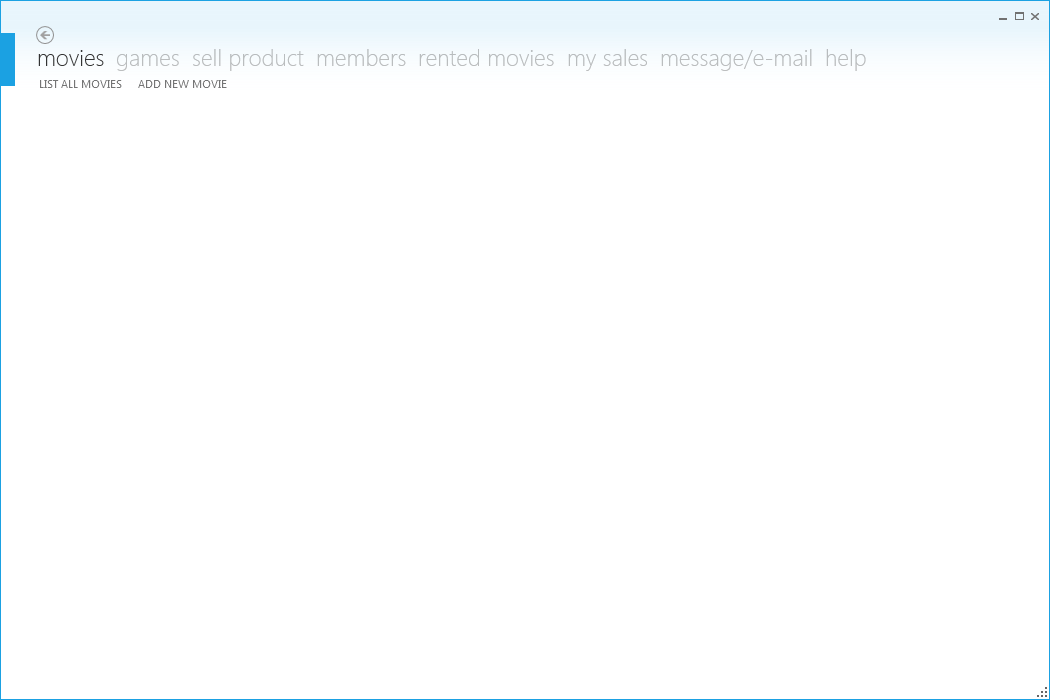


Fig. 6.11 Employee panel after logging in.

**c) Listing, modifying, or deleting an item**

One of the functions of the program is to give a list of the desired item (movies, games, members, employees)

The fig. 6.12 shows a screenshot of the page ‘list all movies’ with a table with all the movies of the store. Fig 6.13 shows all movies in the database.

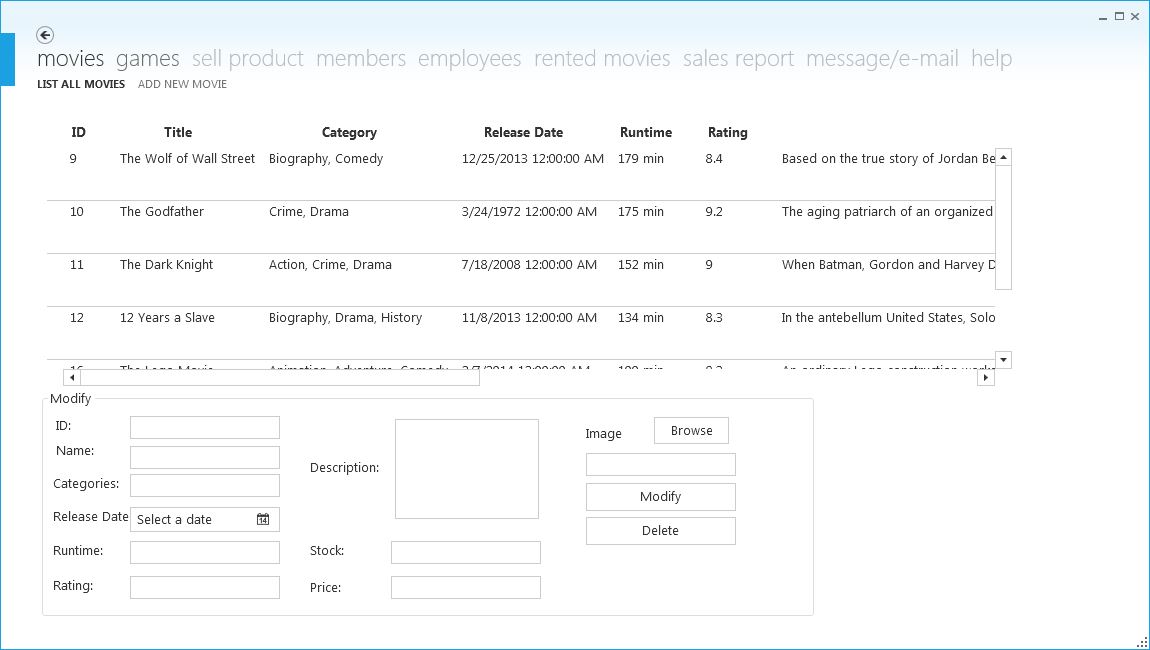


Fig 6.12

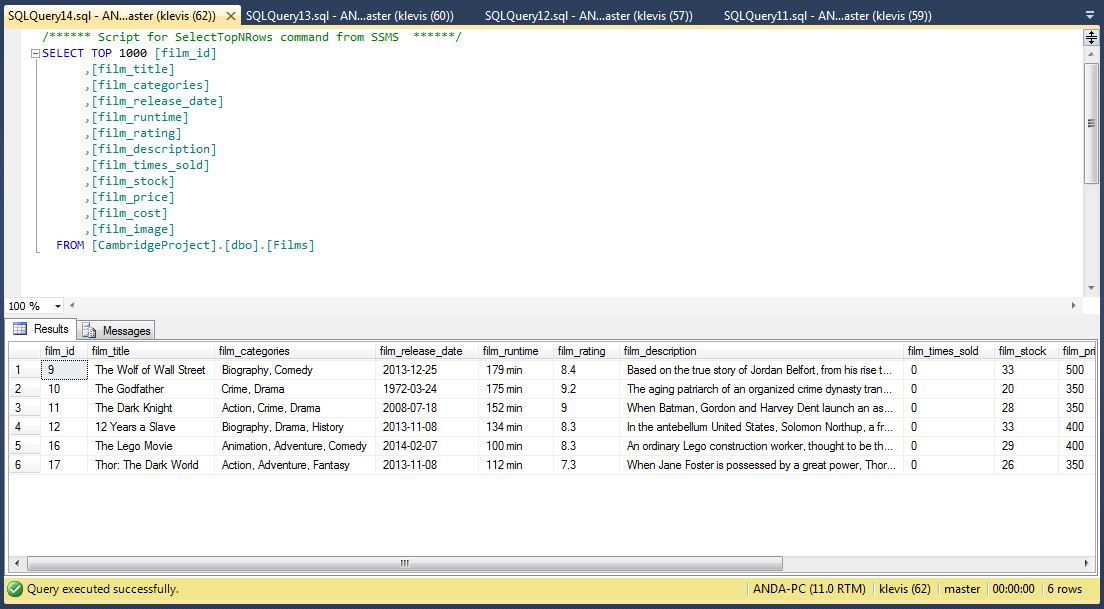


Fig 6.13

Another function that this program provides is to modify one of the items. Fig. 6.14 shows the process of modifying a movie and fig. 6.16 shows the database before the movie is modified. The process of how to modify a movie is explained step by step in section 11. Fig. 6.15 shows the change in the list in the program immediately after the button ‘modify’ is clicked. The change in the database table is shown in the fig. 6.13



Fig 6.14 The process of modifying a movie (in this case the release date is changed)

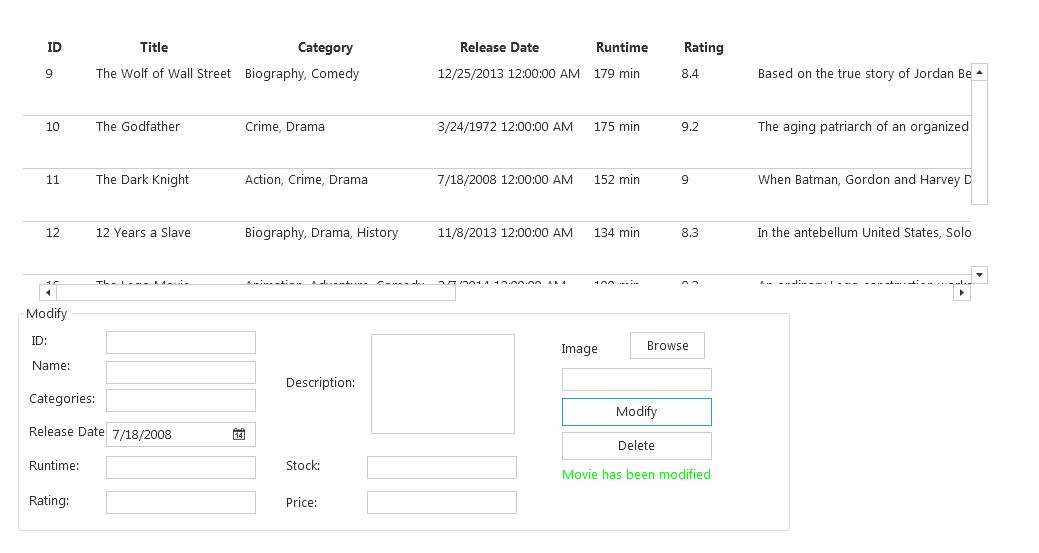


Fig. 6.15 Movie has been modified successfully and the list is refreshed with the new data.

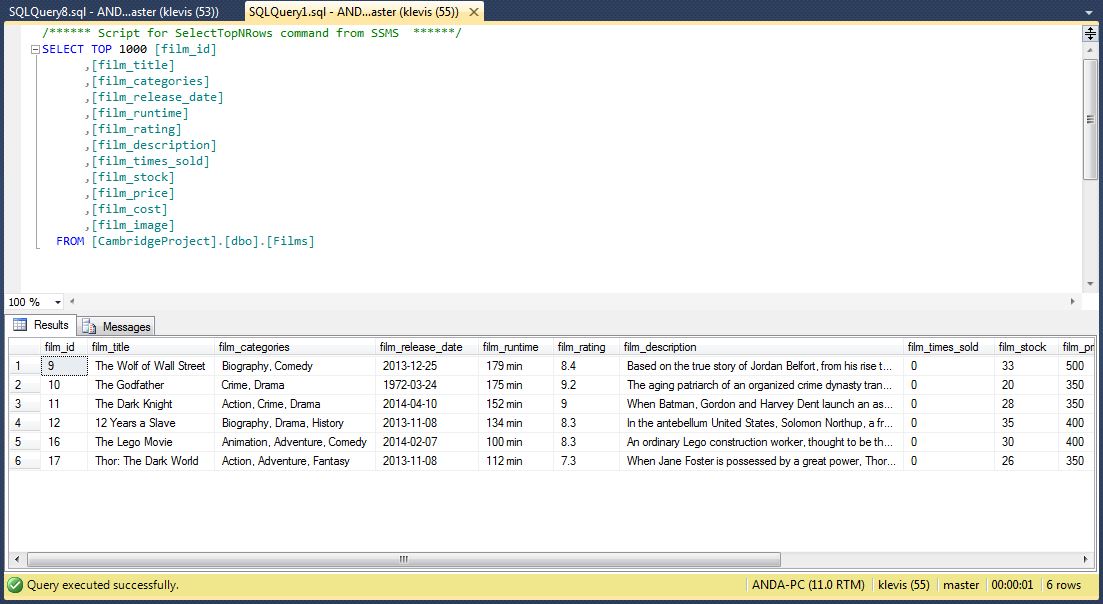


Fig 6.16 The database table before the movie has been modified.

And the last but not least function of the part c) of this section is deleting an item from the program.

The fig. 6.17 shows the process of deleting a movie from the list. The fig. 6.18 shows the table of the database before the deletion of the movie.

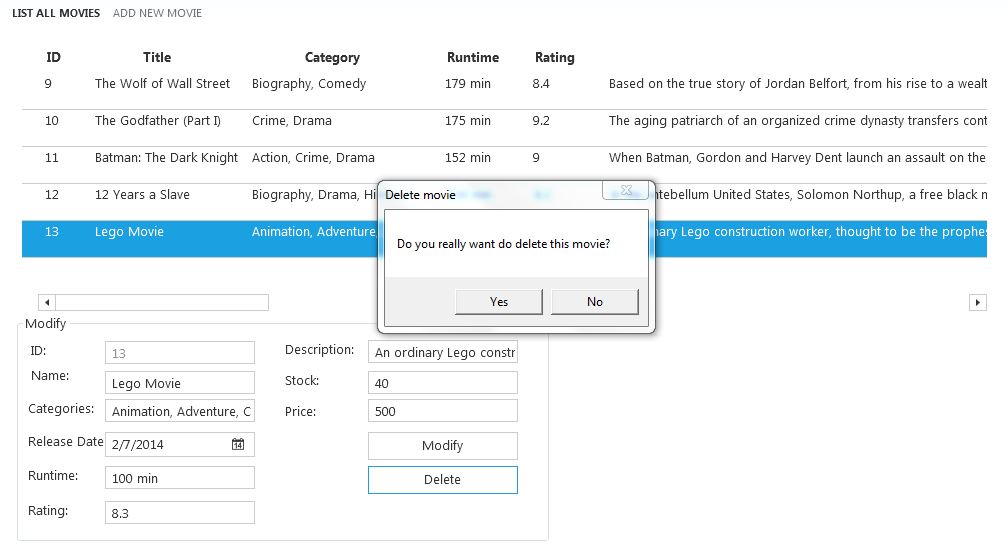


Fig 6.17

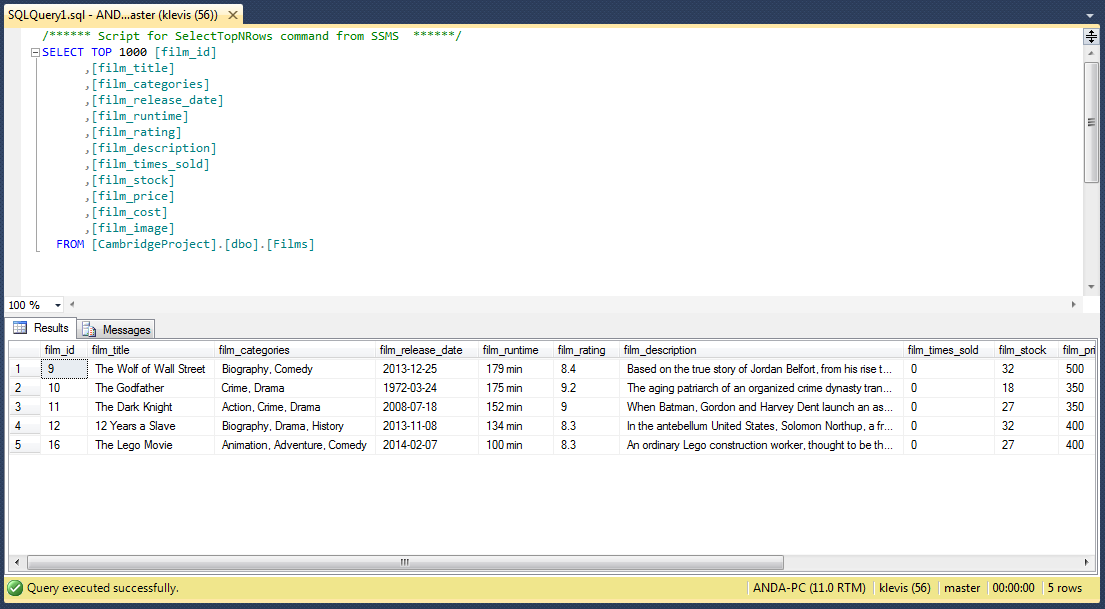


Fig 6.18

After the movie is deleted successfully like in the fig 6.19, the changes also happen in the database (fig. 6.20).

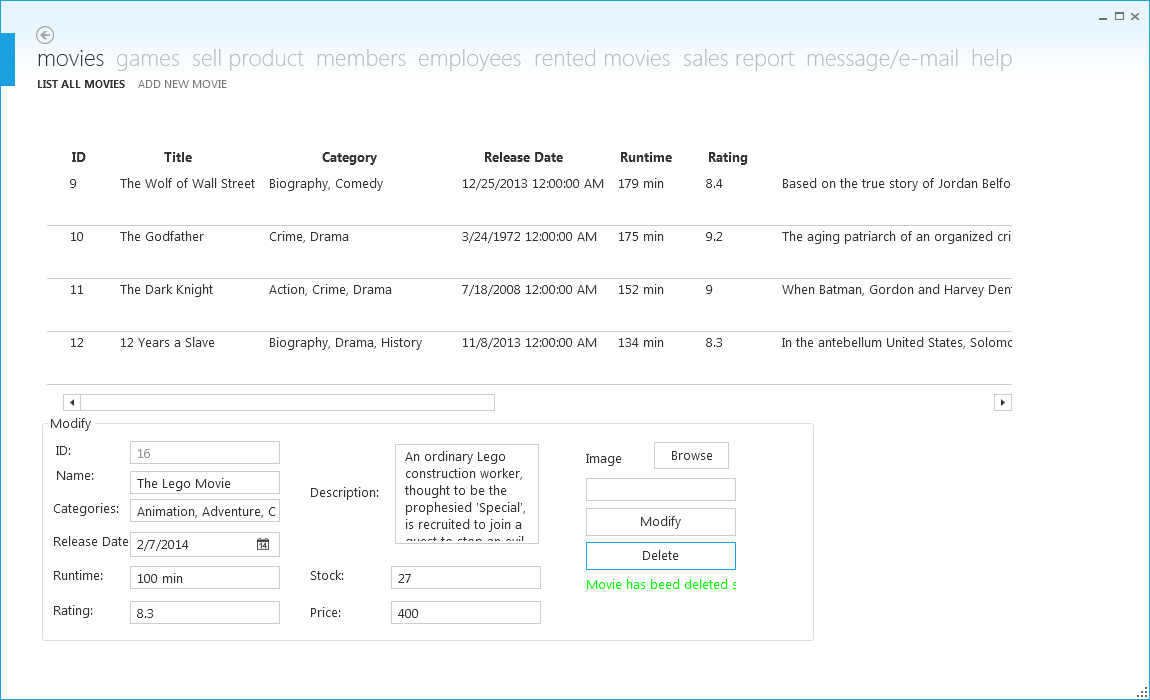


Fig 6.19

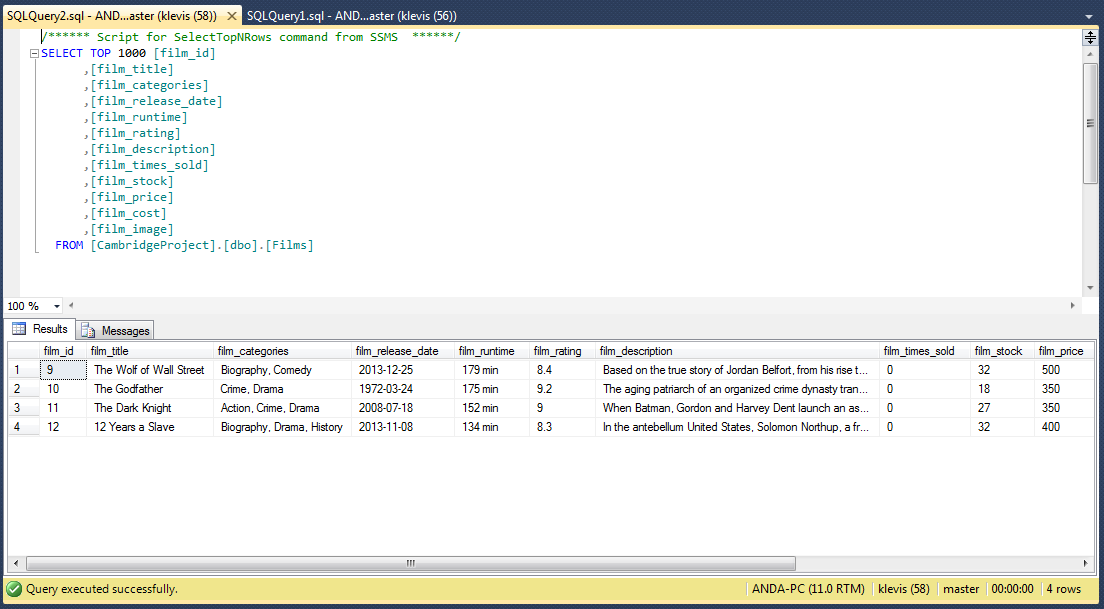


Fig 6.20

**d) Selling a product, renting a movie and pushing its deadline**

Even for these options the changes in the program also occur in the database tables.

The first of these options is selling a movie. Fig 6.21 shows the interface of doing so.

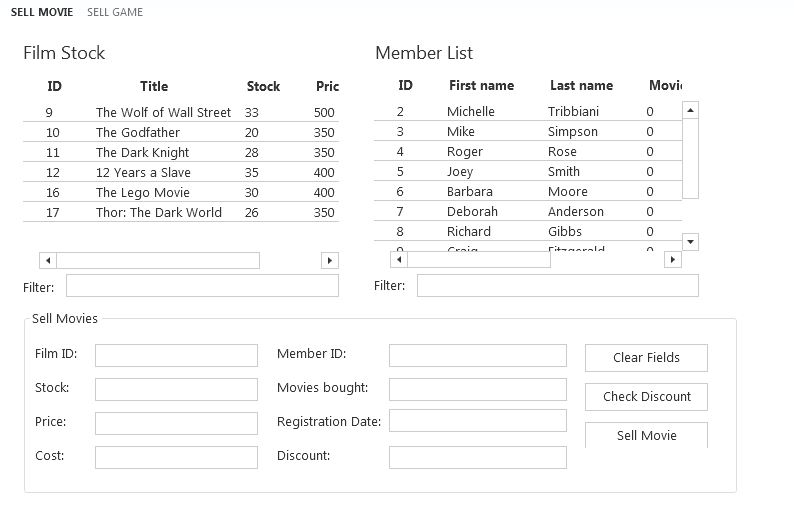


Fig 6.21

A movie can be sold simply by selecting one of the movies from the list and the clicking ‘sell movie’. This is the procedure of selling a movie to a not-member.

To sell the movie to a member you have to select the movie and the member from the list (fig. 6.22). You can also check for discount (fig. 6.22) (this is based on how the owner applies discount to a member). Than by selecting ‘sell movie’ the list changes, so does the database table.

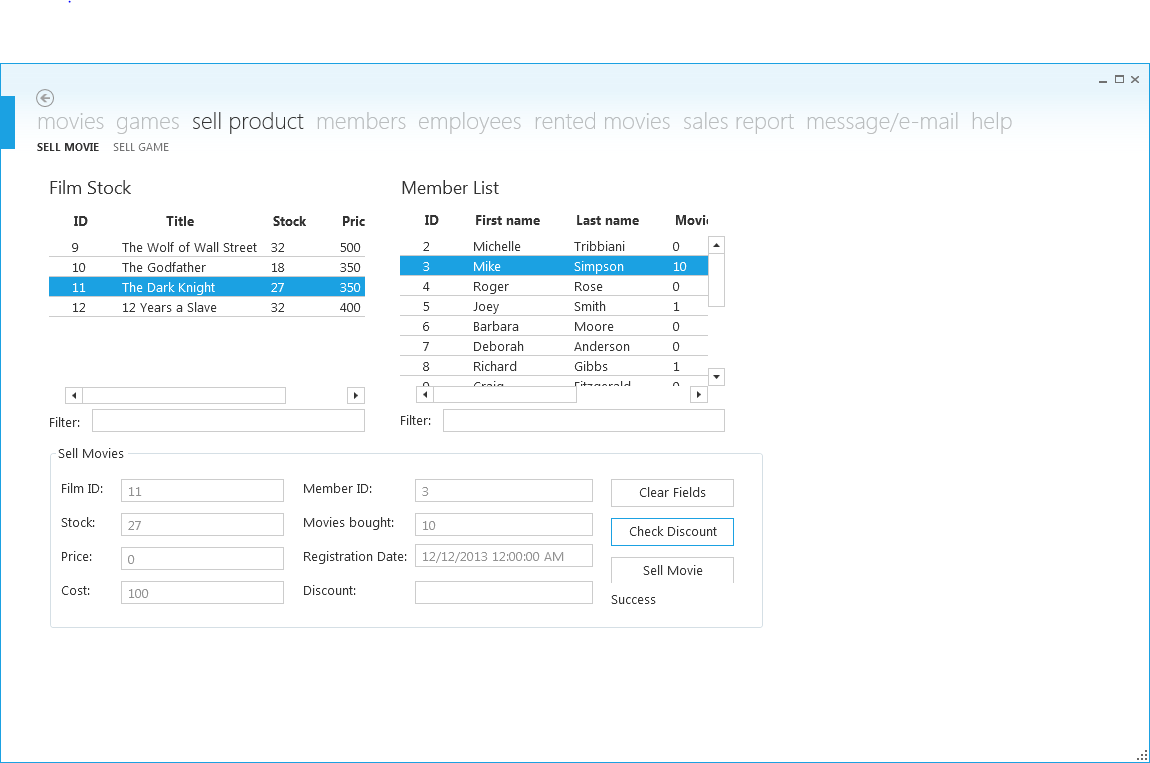


Fig 6.22 The process of selling a movie to a member and checking for discount.

After selling a movie the lists are updated immediately (fig. 6.23), so does the changes in the database fig (6.24). Fig 6.20 shows the database before the sale.

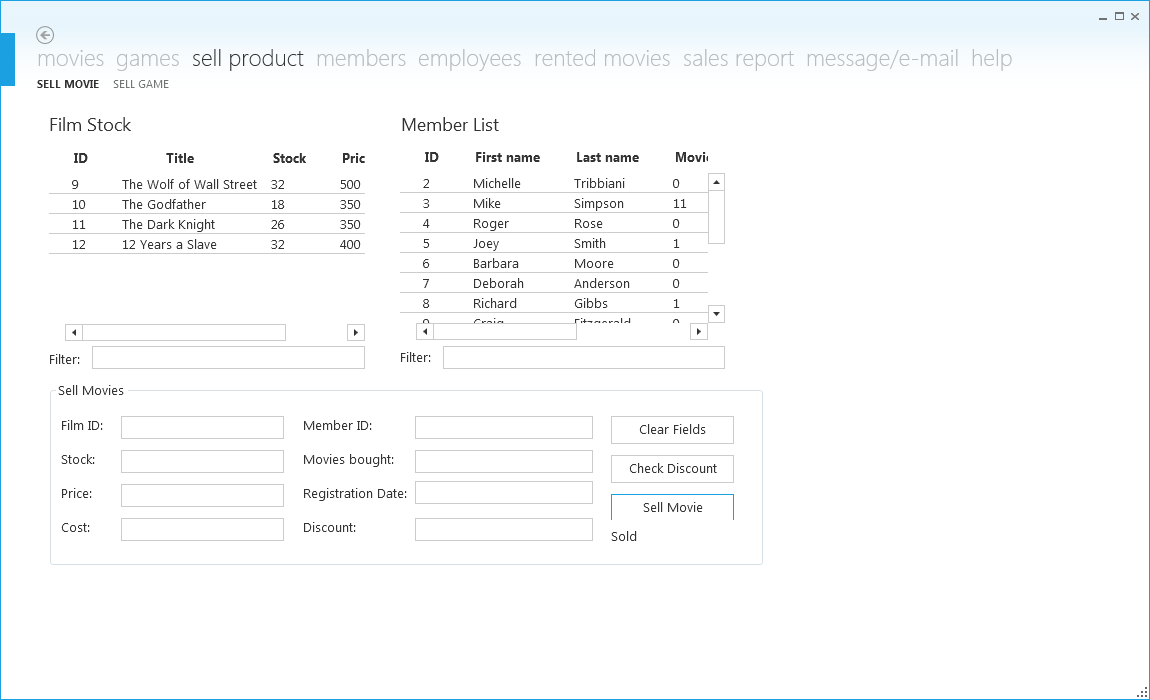


Fig. 6.23

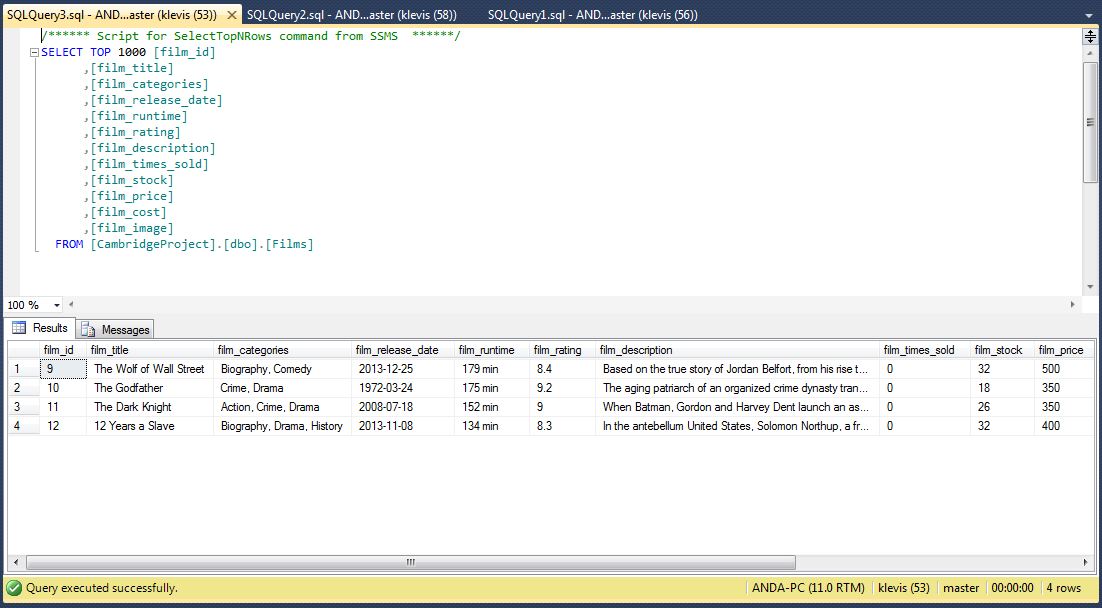


Fig. 6.24 Change in the database table after the sale.

Renting a movie is quite similar to selling a product to a member (fig. 6.25).

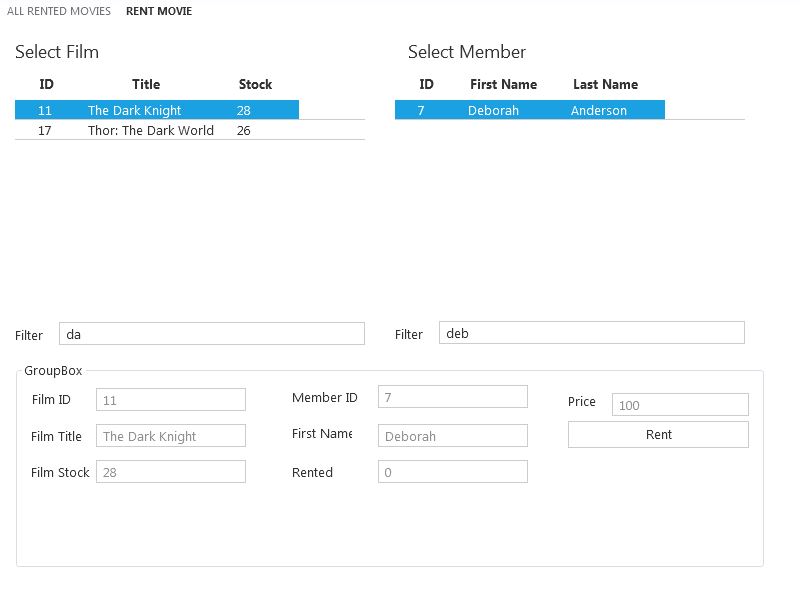


Fig. 6.25 The process of renting a movie

After the movie has been rented the list is updated (fig. 6.26) and so is the database (fig.6.27).

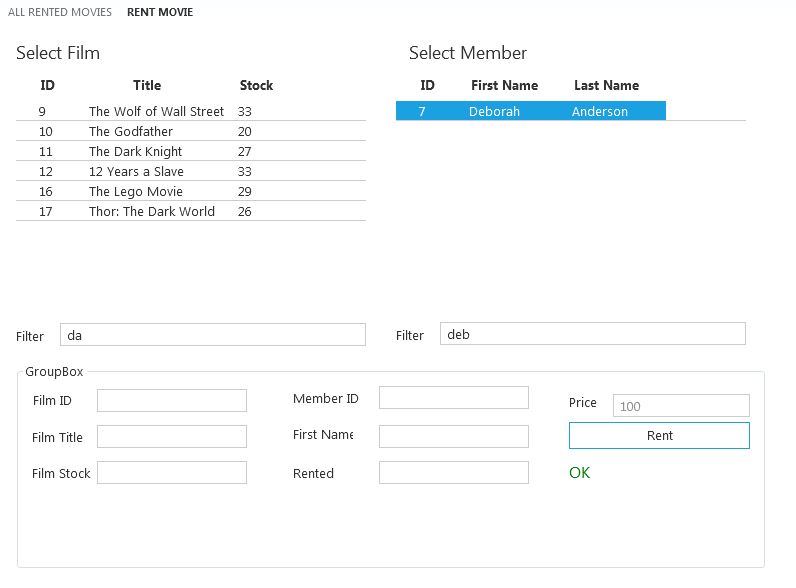


Fig. 6.26 Movie rented successfully.

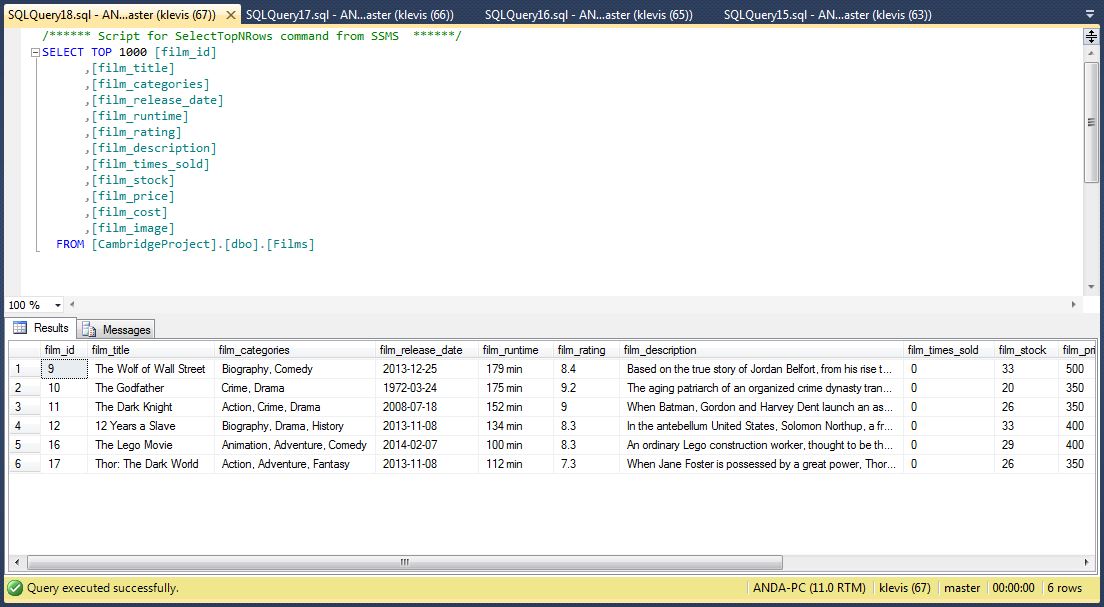


Fig. 6.27 The changes in the database occurred.

When a movie is rented all the rents are stored on a table especially for this process (fig. 6.28).

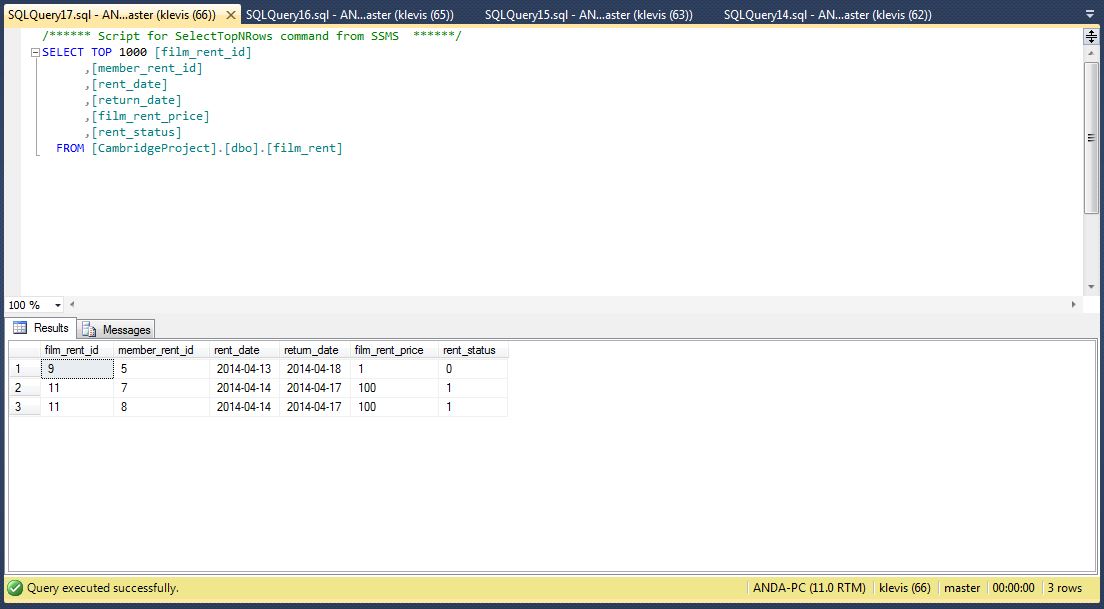


Fig. 6.28

This table helps to make a list of the rents (fig. 6.29). From there you can return the movie and then the stock is increased again by one or you can postpone the date to add one more day to the return date.

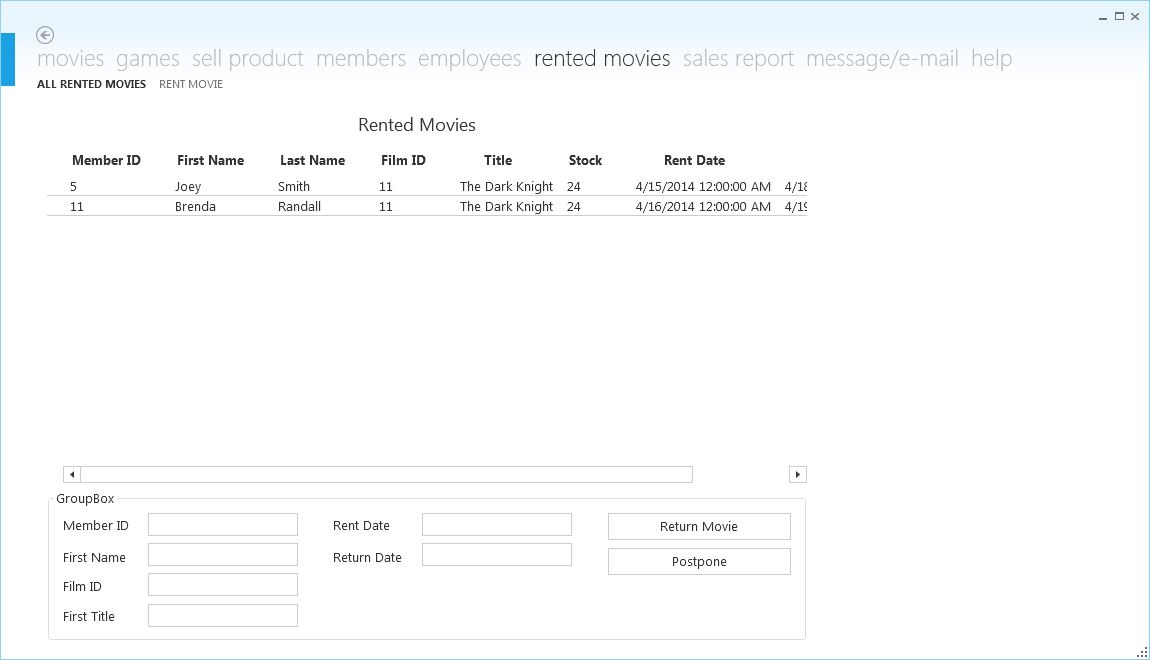


Fig. 6.29 Rent list of movies

**e) Sales report**

Both the administrator and employee can have as an output the list of sales of games or movies (fig. 6.30) of the selected time interval for the administrator, while the employee can view only the total sales (fig. 6.31). These data on the sales list are taken by a table in the database that keeps track of every sale (fig. 6.32). The accounting panel gives the turnover and the brute income of the business for the end of the month. You can also generate a PDF file of games sales, movie sales and at the accounting panel so you can print it.

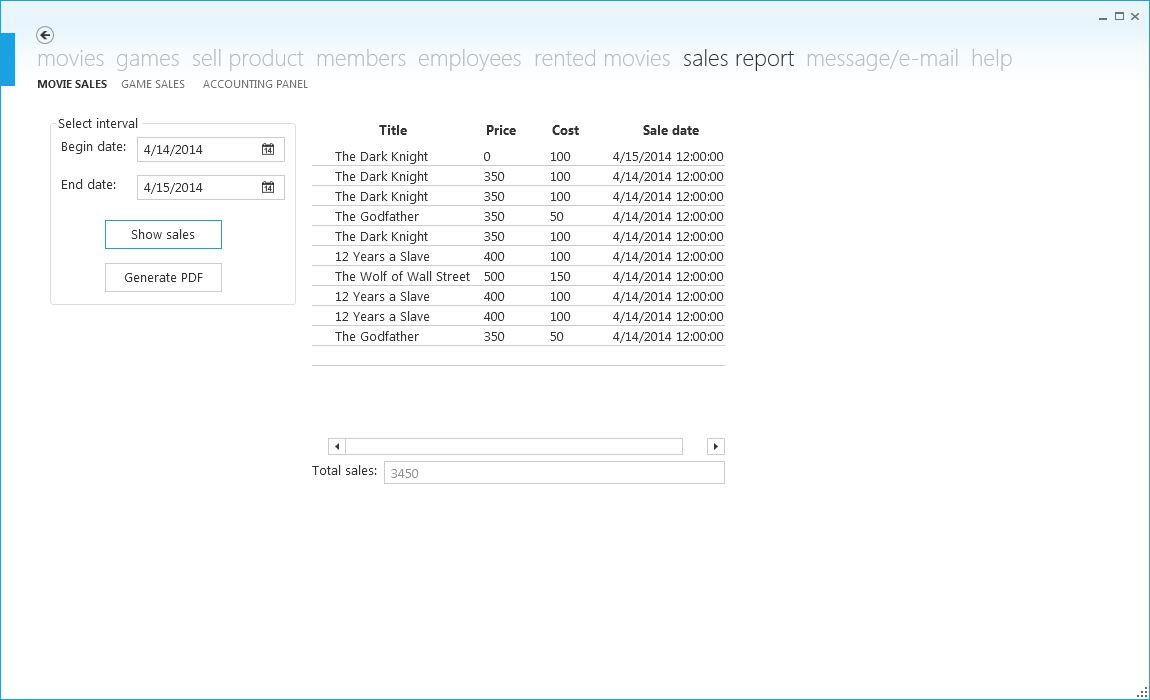


Fig 6.30

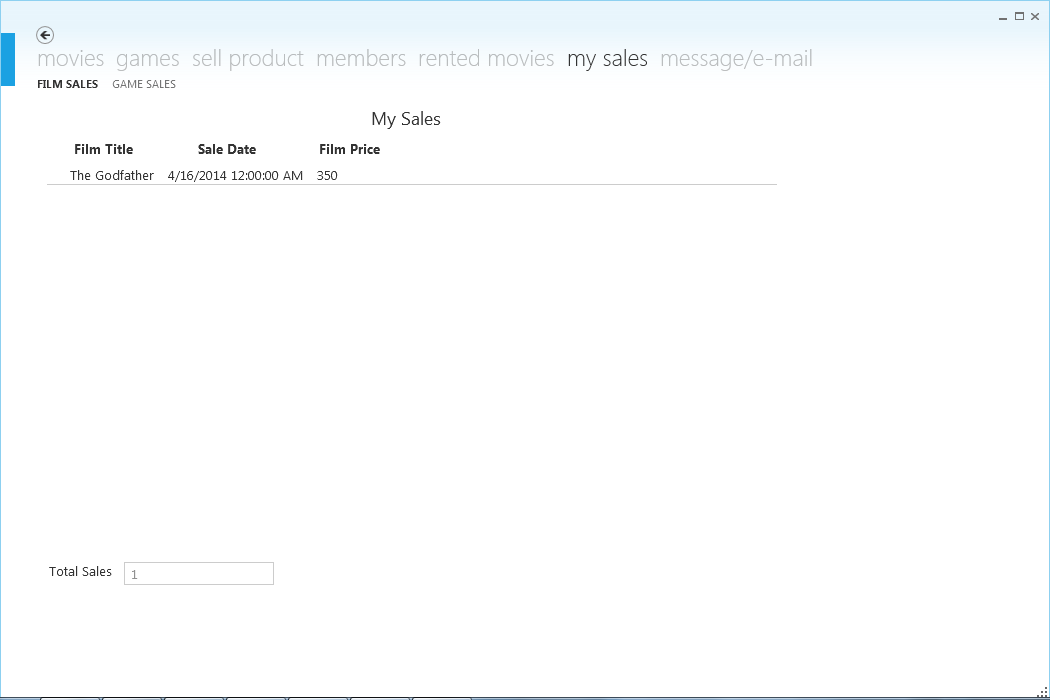


Fig 6.31 A list of sales of an employee.

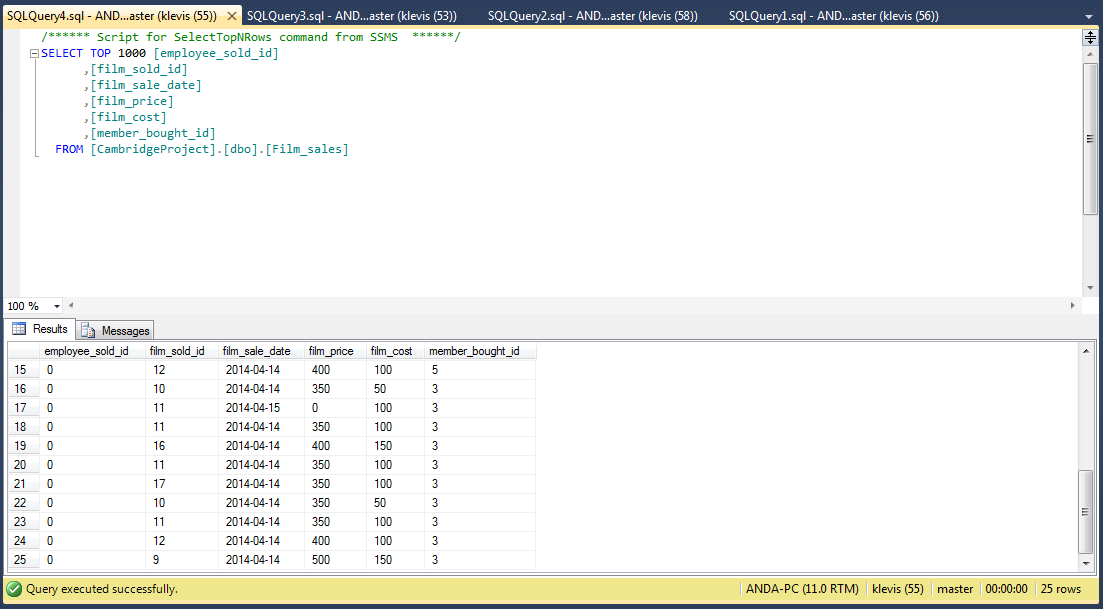


Fig. 6.32 Film sales table in the database

**f) Message/ email to a member**

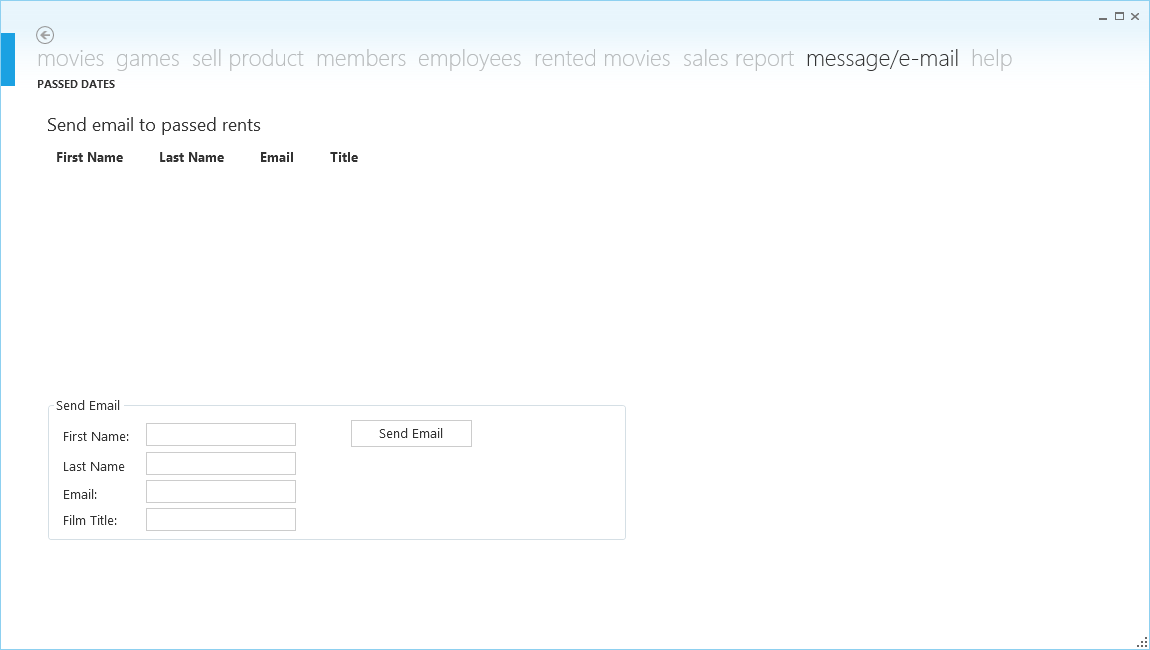
One of many options that this program offers is sending an email to a member that has rented a movie. At the page where you can do this there’s also a list of all the past dates rents (fig 6.33). This list takes the data from the database table ‘film\_rent’ (fig. 6.34). 

Fig. 6.33

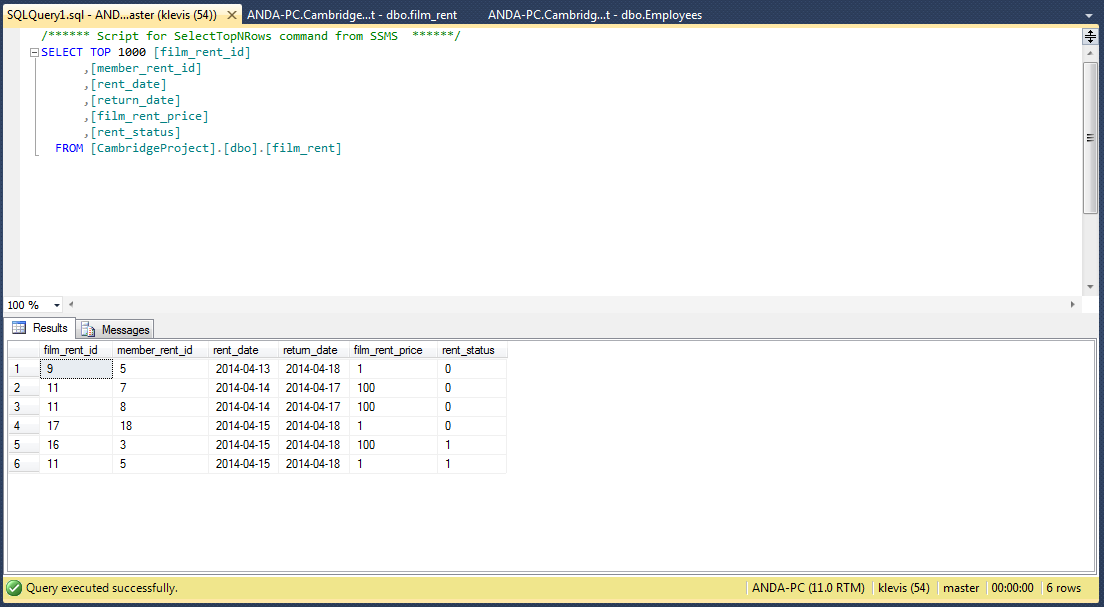


Fig. 6.34 All the rows of the rented movies.

When a movie is not returned after the deadline it’s also shown on the list in the fig. 6.33. From that page an email can be sent by double clicking the member on the list and then clicking ‘send email’ button.

**g) Help**

At the last tab of the program there are three last pages. The first page is a troubleshooting guide (fig. 6.35), the other one is ‘my account’ on which the password is changed (fig. 6.36) and the final one is the log out page from which you can log out with a simple button (fig. 6.37). After logged out, the log in panel appears again (f­ig. 6.38).

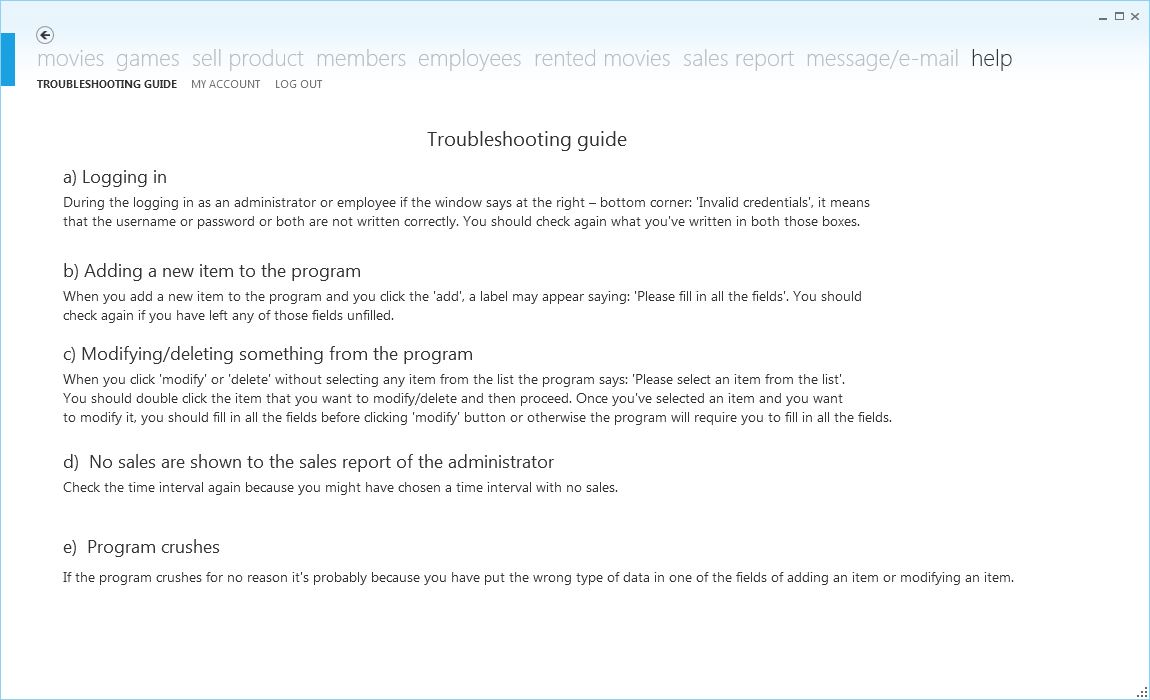


Fig. 6.35 The troubleshooting guide at the help tab.

When the password is changed in the fig. 6.36, the password is changed on the database too (fig. 6.39 and 6.40).



Fig. 6.36 In this case the password is changed to admin2 which is represented in the changed database table in the fig. 6.40.

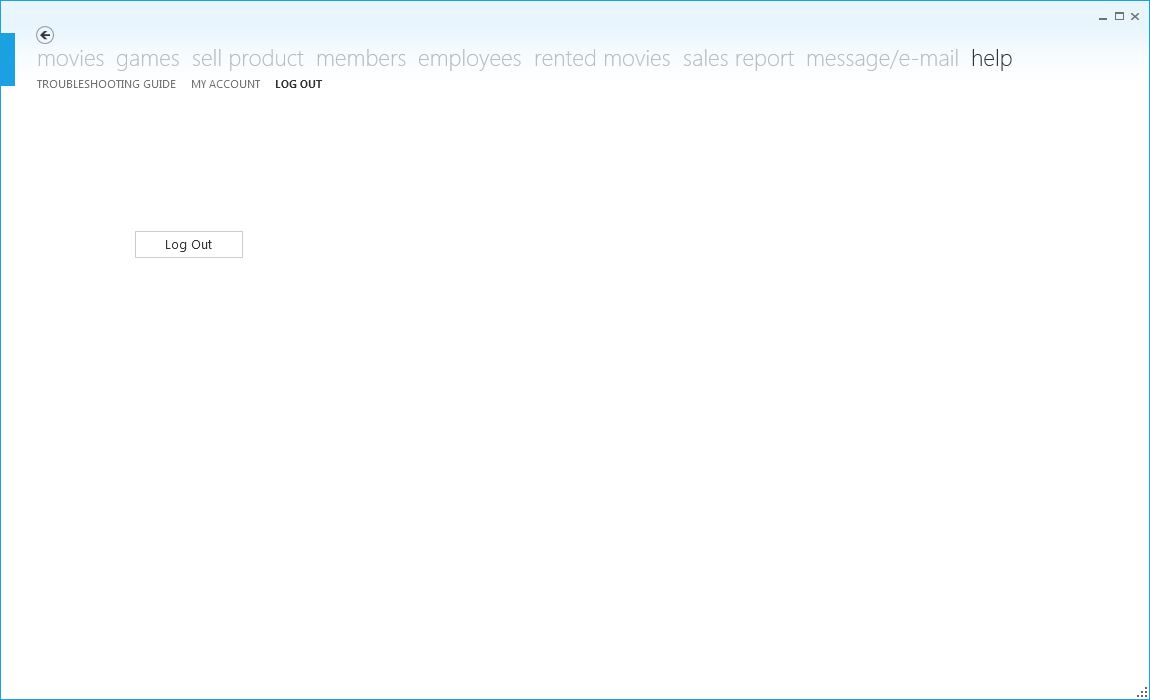
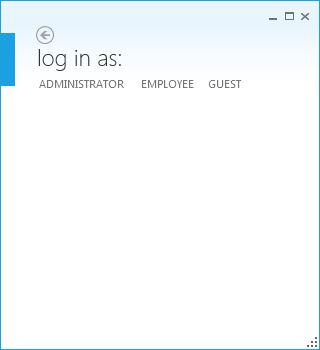


Fig. 6.37 The log out button to log out from the account

 Fig. 6.38 The log in panel.

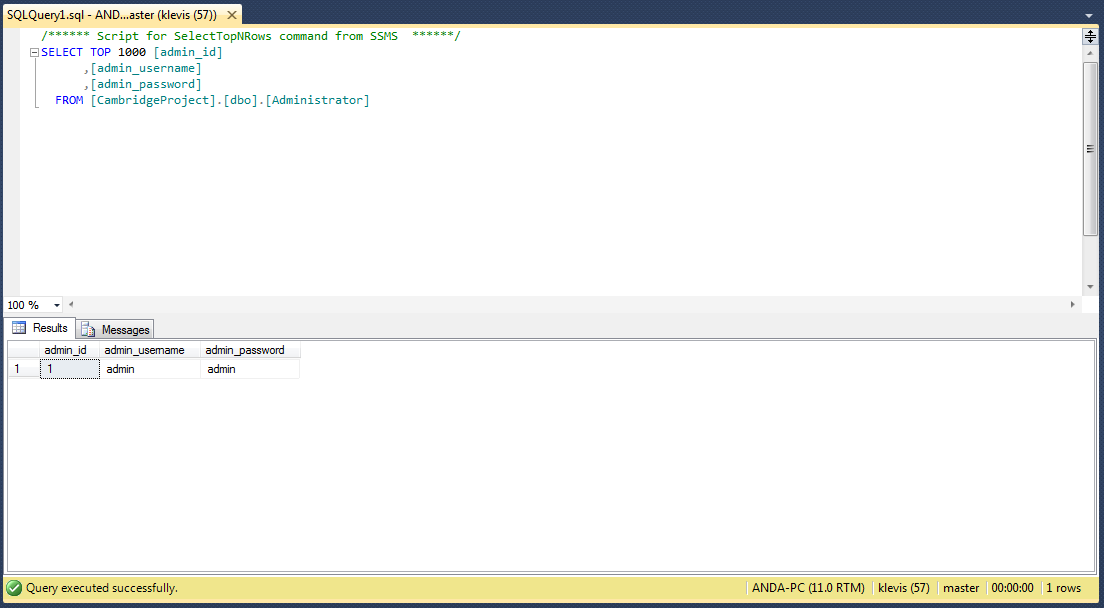


Fig. 6.39 The table ‘Administrator’ in the database before the modification.

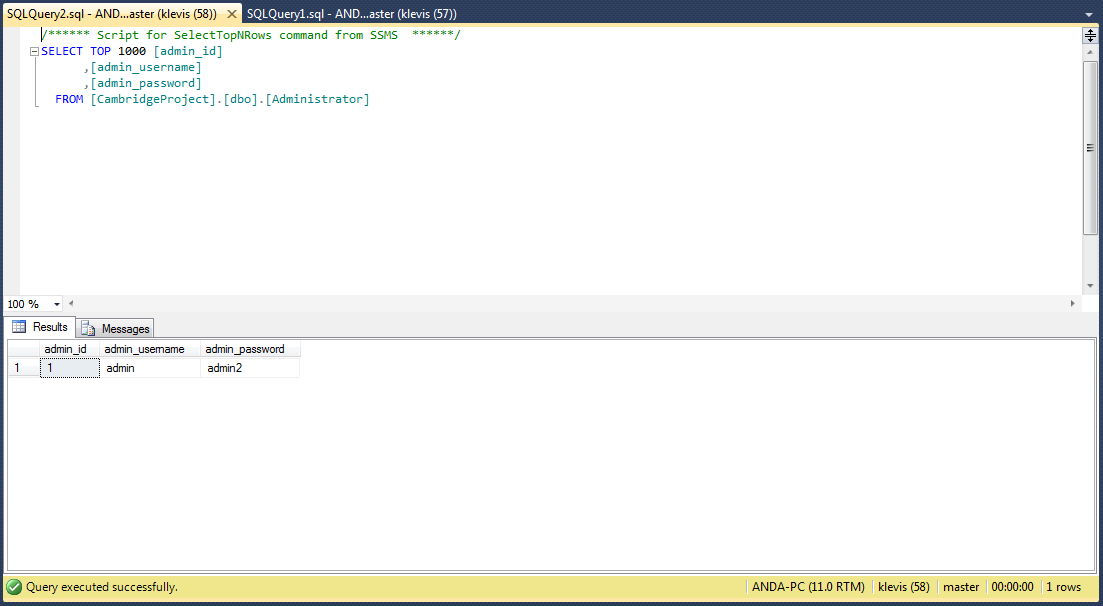


Fig. 6.40 The ‘Administrator’ table after the change.

**h) Guest panel**

The guest panel can make a list of the movies or games. There are three types of lists that this panel can create: top rated, newest entering and top sold (fig. 6.41, 6.42, and 6.43). Another thing that can be done at guest panel is to search a type of movie or game (fig. 6.44).

All these lists get the data from the database table of the movies or games (fig. 6.45). How do they get it is given in the code at section 7.

And the final tab of the guest panel is that one that logs out the same way as in the employee panel or administrator panel.

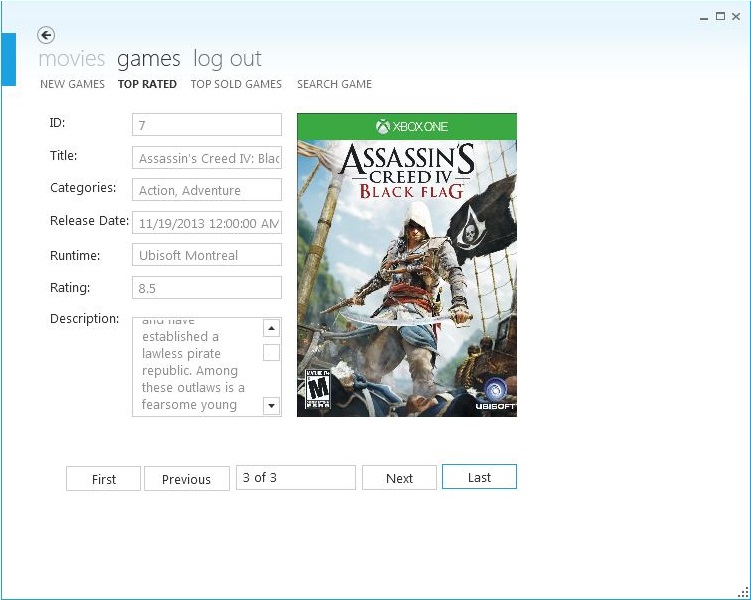


Fig. 6.41 Top Rated Category

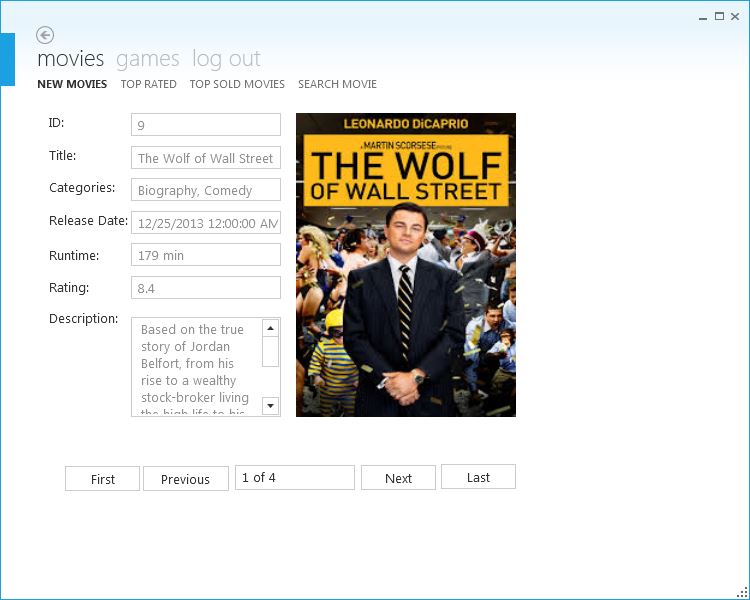


Fig. 6.42 Newest Entering Movies

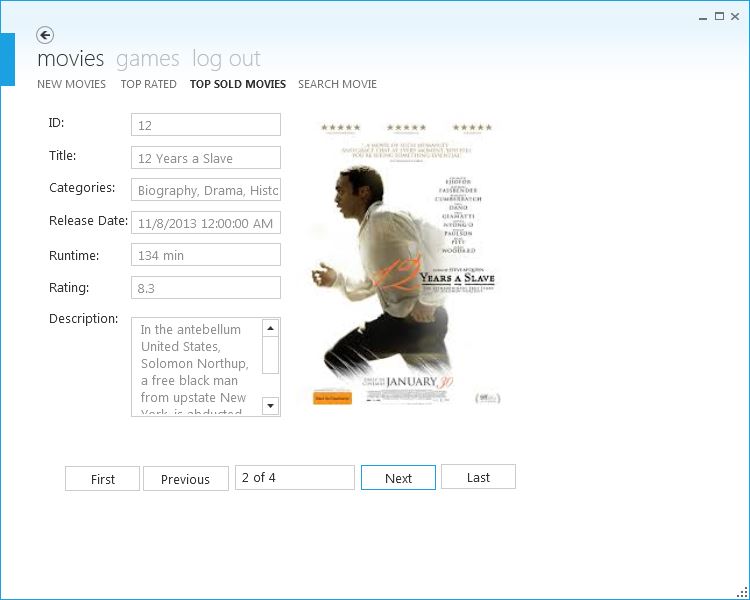


Fig. 6.43 Top Sold Movies

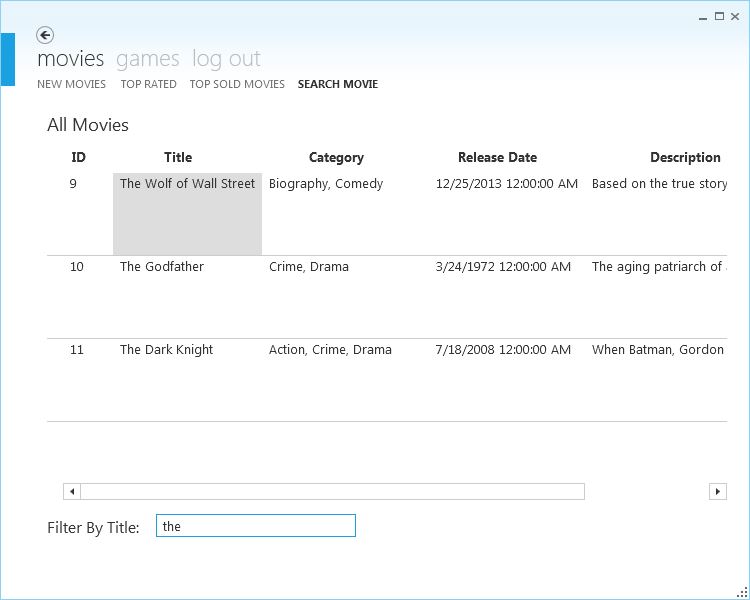


Fig. 6.44 Searching a Movie

**7. Programming**

**i) How to use the Modern UI for the WPF Application**

**Creating a WPF Project with a Modern UI theme:**

1) Get the latest ModernUI release and unzip its contents to disk or install the ModernUI.WPF package using NuGet.

2) Open Visual Studio and create a new WPF Application project  
  
3) Add an assembly reference to FirstFloor.ModernUI.dll (for WPF4 projects you need to add an additional reference to Microsoft.Windows.Shell.dll).  
  
4) Derive MainWindow from ModernWindow

**MainWindow.xaml**

<mui:ModernWindow x:Class="MuiTest.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mui="http://firstfloorsoftware.com/ModernUI"

Title="MainWindow" Height="350" Width="800" IsTitleVisible="True" WindowStartupLocation="CenterScreen">

</mui:ModernWindow>

**MainWindow.xaml.vb**

Imports FirstFloor.ModernUI.Windows.Controls

Partial Public Class MainWindow

Inherits ModernWindow

End Sub

**Application.xaml**

<Application x:Class="Application"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

StartupUri="LoginWindow.xaml">

<Application.Resources>

<ResourceDictionary>

<ResourceDictionary.MergedDictionaries>

<ResourceDictionary Source="/FirstFloor.ModernUI;component/Assets/ModernUI.xaml" />

<ResourceDictionary Source="/FirstFloor.ModernUI;component/Assets/ModernUI.Light.xaml"/>

</ResourceDictionary.MergedDictionaries>

</ResourceDictionary>

</Application.Resources>

</Application>

**ii) Original code**

**a) Logging-in (administrator, employee or guest)**

**MainWindow.xaml.vb**

Imports FirstFloor.ModernUI.Windows.Controls

Imports System.Data

Imports System.Data.SqlClient

Partial Public Class MainWindow

Inherits ModernWindow

'Private variable of this class

Private dbT As DBTransaction

Private dataTable As DataTable

'This ID is used to store who is logged in and identify them in any time

Public Shared id As Integer

'Class constructor

Public Sub New(ByVal eId As Integer)

id = eId

dbT = New DBTransaction()

dataTable = New DataTable()

dbT.checkMembers()

End Sub

'Function to get the logged in user id

Public Shared Function getId() As Integer

Return id

End Function

End Class

**LognWindow.xaml**

<mui:ModernWindow.MenuLinkGroups>

<mui:LinkGroup DisplayName="Log In As:" >

<mui:LinkGroup.Links>

<mui:Link DisplayName="Administrator" Source="/AdministratorLogin.xaml" />

<mui:Link DisplayName="Employee" Source="EmployeeLogin.xaml" />

<mui:Link DisplayName="Guest" Source="GuestLogin.xaml"/>

</mui:LinkGroup.Links>

</mui:LinkGroup>

</mui:ModernWindow.MenuLinkGroups>

**LoginWndow.xaml.vb**

Imports FirstFloor.ModernUI.Windows.Controls

Partial Public Class LoginWindow

Inherits ModernWindow

End Class

**AdministratorLogin.xaml**

<Page x:Class="AdministratorLogin"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="300" d:DesignWidth="250"

Title="AdministratorLogin">

<Grid Loaded="Grid\_Loaded\_1">

<Label Content="Username:" Height="30" HorizontalAlignment="Left" Margin="12,44,0,0" Name="Label1" VerticalAlignment="Top" FontSize="18" />

<Label Content="Password:" FontSize="18" Height="37" HorizontalAlignment="Left" Margin="12,96,0,0" Name="Label2" VerticalAlignment="Top" />

<TextBox Height="30" HorizontalAlignment="Left" Margin="118,44,0,0" Name="txtUsername" VerticalAlignment="Top" Width="120" FontSize="18" />

<PasswordBox Height="30" HorizontalAlignment="Left" Margin="118,93,0,0" Name="txtPassword" VerticalAlignment="Top" Width="120" FontSize="18" />

<Button Content="Log In" Height="35" HorizontalAlignment="Left" Margin="84,156,0,0" Name="btnLogin" VerticalAlignment="Top" Width="86" FontSize="18" />

<Label Height="28" HorizontalAlignment="Left" Margin="84,210,0,0" Name="lblError" VerticalAlignment="Top" FontSize="14" Foreground="Red" />

</Grid>

</Page>

**AdministratorLogin.xaml.vb**

Imports FirstFloor.ModernUI.Windows.Controls

Imports System.Data

Imports System.Data.SqlClient

Imports System.Security

Imports System.IO

Class AdministratorLogin

Dim objConnection As SqlConnection

Dim objCommand As SqlCommand

Dim objDataAdapter As SqlDataAdapter

Dim objDataSet As DataSet

'Button used for logging in as an administrator

Private Sub btnLogin\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnLogin.Click

Dim username As String = txtUsername.Text

Dim password As String = txtPassword.Password

If username.Length = 0 Or password.Length = 0 Then

lblError.Content = "Please fill in all the fields."

Else

Dim connString As String = Utility.GetConnectionString()

objConnection = New SqlConnection(connString)

objCommand.Connection = objConnection

objCommand.Parameters.Clear()

objCommand.CommandText = "SELECT \* FROM administrator WHERE admin\_username=@username AND admin\_password=@password"

objCommand.Parameters.AddWithValue("@username", username)

objCommand.Parameters.AddWithValue("@password", password)

objConnection.Open()

objDataAdapter = New SqlDataAdapter()

objDataAdapter.SelectCommand = objCommand

objDataSet = New DataSet()

objDataAdapter.Fill(objDataSet, "admin")

objConnection.Close()

Dim result As Integer = objDataSet.Tables(0).Rows.Count

'Checking if the inputs of the user match a row in the database table of administrator

If result = 0 Then

lblError.Content = "Invalid credentials"

Else

'Opening a window with all the menus and functions of the administrator

Dim newWindow As New MainWindow(0)

newWindow.Width = 1150

newWindow.Height = 700

Dim mainMenu(9) As FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(1) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(1).DisplayName = "Movies"

mainMenu(2) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(2).DisplayName = "Games"

mainMenu(3) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(3).DisplayName = "Sell Product"

mainMenu(4) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(4).DisplayName = "Members"

mainMenu(5) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(5).DisplayName = "Employees"

mainMenu(6) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(6).DisplayName = "Rented Movies"

mainMenu(7) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(7).DisplayName = "Sales Report"

mainMenu(8) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(8).DisplayName = "Message/E-mail"

mainMenu(9) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(9).DisplayName = "Help"

newWindow.MenuLinkGroups.Add(mainMenu(1))

newWindow.MenuLinkGroups.Add(mainMenu(2))

newWindow.MenuLinkGroups.Add(mainMenu(3))

newWindow.MenuLinkGroups.Add(mainMenu(4))

newWindow.MenuLinkGroups.Add(mainMenu(5))

newWindow.MenuLinkGroups.Add(mainMenu(6))

newWindow.MenuLinkGroups.Add(mainMenu(7))

newWindow.MenuLinkGroups.Add(mainMenu(8))

newWindow.MenuLinkGroups.Add(mainMenu(9))

newWindow.Show()

Dim listMovies As New FirstFloor.ModernUI.Presentation.Link

listMovies.DisplayName = "List all movies"

listMovies.Source = New Uri("/listMovies.xaml", UriKind.Relative)

Dim addMovies As New FirstFloor.ModernUI.Presentation.Link

addMovies.DisplayName = "Add new movie"

addMovies.Source = New Uri("/addMovies.xaml", UriKind.Relative)

mainMenu(1).Links.Add(listMovies)

mainMenu(1).Links.Add(addMovies)

Dim listGames As New FirstFloor.ModernUI.Presentation.Link

listGames.DisplayName = "List all games"

listGames.Source = New Uri("/listGames.xaml", UriKind.Relative)

Dim addGame As New FirstFloor.ModernUI.Presentation.Link

addGame.DisplayName = "Add new game"

addGame.Source = New Uri("/addGame.xaml", UriKind.Relative)

Dim comp As New FirstFloor.ModernUI.Presentation.Link

comp.DisplayName = "Compatibilities"

comp.Source = New Uri("/addCompatibility.xaml", UriKind.Relative)

mainMenu(2).Links.Add(listGames)

mainMenu(2).Links.Add(addGame)

mainMenu(2).Links.Add(comp)

Dim sellMovie As New FirstFloor.ModernUI.Presentation.Link

sellMovie.DisplayName = "Sell movie"

sellMovie.Source = New Uri("/sellMovie.xaml", UriKind.Relative)

Dim sellGame As New FirstFloor.ModernUI.Presentation.Link

sellGame.DisplayName = "Sell game"

sellGame.Source = New Uri("/sellGame.xaml", UriKind.Relative)

mainMenu(3).Links.Add(sellMovie)

mainMenu(3).Links.Add(sellGame)

Dim listMember As New FirstFloor.ModernUI.Presentation.Link

listMember.DisplayName = "List all members"

listMember.Source = New Uri("/listMember.xaml", UriKind.Relative)

Dim addMember As New FirstFloor.ModernUI.Presentation.Link

addMember.DisplayName = "Add new member"

addMember.Source = New Uri("/addMember.xaml", UriKind.Relative)

mainMenu(4).Links.Add(listMember)

mainMenu(4).Links.Add(addMember)

Dim listEmployee As New FirstFloor.ModernUI.Presentation.Link

listEmployee.DisplayName = "List all employees"

listEmployee.Source = New Uri("/listEmployee.xaml", UriKind.Relative)

Dim addEmployee As New FirstFloor.ModernUI.Presentation.Link

addEmployee.DisplayName = "Add new employee"

addEmployee.Source = New Uri("/addEmployee.xaml", UriKind.Relative)

mainMenu(5).Links.Add(listEmployee)

mainMenu(5).Links.Add(addEmployee)

Dim rentedMovies As New FirstFloor.ModernUI.Presentation.Link

rentedMovies.DisplayName = "All rented movies"

rentedMovies.Source = New Uri("/rentedMovies.xaml", UriKind.Relative)

Dim returnMovie As New FirstFloor.ModernUI.Presentation.Link

returnMovie.DisplayName = "Rent Movie"

returnMovie.Source = New Uri("/rentMovie.xaml", UriKind.Relative)

Dim sendEmail As New FirstFloor.ModernUI.Presentation.Link

sendEmail.DisplayName = "Passed Dates"

sendEmail.Source = New Uri("/emailForRents.xaml", UriKind.Relative)

mainMenu(6).Links.Add(rentedMovies)

mainMenu(6).Links.Add(returnMovie)

mainMenu(8).Links.Add(sendEmail)

Dim movieSales As New FirstFloor.ModernUI.Presentation.Link

movieSales.DisplayName = "Movie sales"

movieSales.Source = New Uri("/salesMovies.xaml", UriKind.Relative)

Dim gameSales As New FirstFloor.ModernUI.Presentation.Link

gameSales.DisplayName = "Game Sales"

gameSales.Source = New Uri("/salesGames.xaml", UriKind.Relative)

Dim accounting As New FirstFloor.ModernUI.Presentation.Link

accounting.DisplayName = "Accounting Panel"

accounting.Source = New Uri("/AccountingPanel.xaml", UriKind.Relative)

mainMenu(7).Links.Add(movieSales)

mainMenu(7).Links.Add(gameSales)

mainMenu(7).Links.Add(accounting)

Dim manual As New FirstFloor.ModernUI.Presentation.Link

manual.DisplayName = "Troubleshooting guide"

manual.Source = New Uri("/manual.xaml", UriKind.Relative)

Dim account As New FirstFloor.ModernUI.Presentation.Link

account.DisplayName = "My Account"

account.Source = New Uri("/adminAccount.xaml", UriKind.Relative)

Dim logOut As New FirstFloor.ModernUI.Presentation.Link

logOut.DisplayName = "Log Out"

logOut.Source = New Uri("/logOut.xaml", UriKind.Relative)

mainMenu(9).Links.Add(manual)

mainMenu(9).Links.Add(account)

mainMenu(9).Links.Add(logOut)

Dim currentWindow As ModernWindow = ModernWindow.GetWindow(Me)

currentWindow.Close()

End If

End If

End Sub

Private Sub txtUsername\_GotFocus(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles txtUsername.GotFocus

lblError.Content = ""

End Sub

Private Sub txtPassword\_GotFocus(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles txtPassword.GotFocus

lblError.Content = ""

End Sub

'When loaded this window checks if the database and a folder called reports exists

'If not it creates them

Private Sub Grid\_Loaded\_1(sender As Object, e As RoutedEventArgs)

Dim ut As Utility = New Utility()

objConnection = New SqlConnection("server=localhost;Integrated Security=SSPI")

objCommand = New SqlCommand()

objCommand.Connection = objConnection

objCommand.CommandType = CommandType.Text

Dim databaseSt As Boolean = Utility.CheckDatabaseExists("localhost", "CambridgeProject")

Dim filePath As String = Directory.GetCurrentDirectory() & "\databaseScripts.sql"

Dim filePathTables As String = Directory.GetCurrentDirectory() & "\databaseTableScripts.sql"

If databaseSt = False Then

Dim file As FileInfo = New FileInfo(filePath)

Dim file1 As FileInfo = New FileInfo(filePathTables)

Dim script As String = file.OpenText().ReadToEnd()

Dim rscript As String = script.Replace("GO", "")

Dim script1 As String = file1.OpenText().ReadToEnd()

Dim rscript1 As String = script1.Replace("GO", "")

objCommand.CommandText = rscript

objCommand.Connection.Open()

objCommand.ExecuteNonQuery()

objCommand.CommandText = rscript1

objCommand.ExecuteNonQuery()

objConnection.Close()

Dim path As String = Directory.GetCurrentDirectory()

path = path.Substring(0, 1)

path = path & ":\Reports"

If Not Directory.Exists(path) Then

Directory.CreateDirectory(path)

End If

End If

End Sub

End Class

**EmployeeLogin.xaml**

<Page x:Class="EmployeeLogin"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="300" d:DesignWidth="250"

Title="EmployeeLogin">

<Grid Name="mainGrid" Loaded="Grid\_Loaded\_1">

<Label Content="Username:" Height="30" HorizontalAlignment="Left" Margin="12,44,0,0" Name="Label1" VerticalAlignment="Top" FontSize="18" />

<Label Content="Password:" FontSize="18" Height="37" HorizontalAlignment="Left" Margin="12,96,0,0" Name="Label2" VerticalAlignment="Top" />

<TextBox Height="30" HorizontalAlignment="Left" Margin="118,44,0,0" Name="txtUsername" VerticalAlignment="Top" Width="120" FontSize="18" />

<PasswordBox Height="30" HorizontalAlignment="Left" Margin="118,93,0,0" Name="txtPassword" VerticalAlignment="Top" Width="120" FontSize="18" />

<Button Content="Log In" Height="35" HorizontalAlignment="Left" Margin="84,156,0,0" Name="btnLogin" VerticalAlignment="Top" Width="86" FontSize="18" />

<Label Height="28" HorizontalAlignment="Left" Margin="84,210,0,0" Name="lblError" VerticalAlignment="Top" FontSize="14" Foreground="Red" />

</Grid>

</Page>

**Employee.Login.xaml.vb**

Imports FirstFloor.ModernUI.Windows.Controls

Imports System.Data

Imports System.Data.SqlClient

Imports System.Security

Class EmployeeLogin

Dim objConnection As SqlConnection

Dim objCommand As SqlCommand

Dim objDataAdapter As SqlDataAdapter

Dim objDataSet As DataSet

Dim db As DBTransaction

'Button used for logging in as an employee

Private Sub btnLogin\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnLogin.Click

Dim username As String = txtUsername.Text

Dim password As String = txtPassword.Password

If username.Length = 0 Or password.Length = 0 Then

lblError.Content = "Please fill in all the fields."

Else

Dim connString As String = Utility.GetConnectionString()

objConnection = New SqlConnection(connString)

objCommand = New SqlCommand()

objCommand.Connection = objConnection

objCommand.CommandType = CommandType.Text

objCommand.CommandText = "SELECT \* FROM Employees WHERE employee\_username=@username AND employee\_password=@password"

objCommand.Parameters.AddWithValue("@username", username)

objCommand.Parameters.AddWithValue("@password", password)

objConnection.Open()

objDataAdapter = New SqlDataAdapter()

objDataAdapter.SelectCommand = objCommand

objDataSet = New DataSet()

objDataAdapter.Fill(objDataSet, "employee")

objConnection.Close()

Dim result As Integer = objDataSet.Tables(0).Rows.Count

'Checking if the inputs of the user match a row in the database table of administrator

If result = 0 Then

lblError.Content = "Invalid credentials"

Else

'Getting the id of the employee logged in

Dim dr As DataRow = objDataSet.Tables(0).Rows(0)

Dim id As Integer = dr.Item(0)

'Opening a window with all the menus and functions of the administrator

Dim newWindow As New MainWindow(id)

newWindow.Width = 1050

newWindow.Height = 700

Dim mainMenu(8) As FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(1) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(1).DisplayName = "Movies"

mainMenu(2) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(2).DisplayName = "Games"

mainMenu(3) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(3).DisplayName = "Sell Product"

mainMenu(4) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(4).DisplayName = "Members"

mainMenu(5) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(5).DisplayName = "Rented Movies"

mainMenu(6) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(6).DisplayName = "My Sales"

mainMenu(7) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(7).DisplayName = "Message/E-Mail"

mainMenu(8) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(8).DisplayName = "Help"

newWindow.MenuLinkGroups.Add(mainMenu(1))

newWindow.MenuLinkGroups.Add(mainMenu(2))

newWindow.MenuLinkGroups.Add(mainMenu(3))

newWindow.MenuLinkGroups.Add(mainMenu(4))

newWindow.MenuLinkGroups.Add(mainMenu(5))

newWindow.MenuLinkGroups.Add(mainMenu(6))

newWindow.MenuLinkGroups.Add(mainMenu(7))

newWindow.MenuLinkGroups.Add(mainMenu(8))

newWindow.Show()

Dim listMovies As New FirstFloor.ModernUI.Presentation.Link

listMovies.DisplayName = "List all movies"

listMovies.Source = New Uri("/listMovies.xaml", UriKind.Relative)

Dim addMovies As New FirstFloor.ModernUI.Presentation.Link

addMovies.DisplayName = "Add new movie"

addMovies.Source = New Uri("/addMovies.xaml", UriKind.Relative)

mainMenu(1).Links.Add(listMovies)

mainMenu(1).Links.Add(addMovies)

Dim listGames As New FirstFloor.ModernUI.Presentation.Link

listGames.DisplayName = "List all games"

listGames.Source = New Uri("/listGames.xaml", UriKind.Relative)

Dim addGame As New FirstFloor.ModernUI.Presentation.Link

addGame.DisplayName = "Add new game"

addGame.Source = New Uri("/addGame.xaml", UriKind.Relative)

Dim comp As New FirstFloor.ModernUI.Presentation.Link

comp.DisplayName = "Compatibilities"

comp.Source = New Uri("/addCompatibility.xaml", UriKind.Relative)

mainMenu(2).Links.Add(listGames)

mainMenu(2).Links.Add(addGame)

mainMenu(2).Links.Add(comp)

Dim sellMovie As New FirstFloor.ModernUI.Presentation.Link

sellMovie.DisplayName = "Sell Movie"

sellMovie.Source = New Uri("/sellMovie.xaml", UriKind.Relative)

Dim sellGame As New FirstFloor.ModernUI.Presentation.Link

sellGame.DisplayName = "Sell Game"

sellGame.Source = New Uri("/sellGame.xaml", UriKind.Relative)

mainMenu(3).Links.Add(sellMovie)

mainMenu(3).Links.Add(sellGame)

Dim listMember As New FirstFloor.ModernUI.Presentation.Link

listMember.DisplayName = "List all members"

listMember.Source = New Uri("/listMember.xaml", UriKind.Relative)

Dim addMember As New FirstFloor.ModernUI.Presentation.Link

addMember.DisplayName = "Add new member"

addMember.Source = New Uri("/addMember.xaml", UriKind.Relative)

mainMenu(4).Links.Add(listMember)

mainMenu(4).Links.Add(addMember)

Dim rentedMovies As New FirstFloor.ModernUI.Presentation.Link

rentedMovies.DisplayName = "All rented movies"

rentedMovies.Source = New Uri("/rentedMovies.xaml", UriKind.Relative)

Dim returnMovie As New FirstFloor.ModernUI.Presentation.Link

returnMovie.DisplayName = "Rent Movie"

returnMovie.Source = New Uri("/rentMovie.xaml", UriKind.Relative)

mainMenu(5).Links.Add(rentedMovies)

mainMenu(5).Links.Add(returnMovie)

Dim filmSales As New FirstFloor.ModernUI.Presentation.Link

filmSales.DisplayName = "Film Sales"

filmSales.Source = New Uri("/filmSales.xaml", UriKind.Relative)

Dim gameSales As New FirstFloor.ModernUI.Presentation.Link

gameSales.DisplayName = "Game Sales"

gameSales.Source = New Uri("/gameSales.xaml", UriKind.Relative)

mainMenu(6).Links.Add(filmSales)

mainMenu(6).Links.Add(gameSales)

Dim passedDates As New FirstFloor.ModernUI.Presentation.Link

passedDates.DisplayName = "Passed Dates"

passedDates.Source = New Uri("/emailForRents.xaml", UriKind.Relative)

mainMenu(7).Links.Add(passedDates)

Dim manual As New FirstFloor.ModernUI.Presentation.Link

manual.DisplayName = "Manual"

manual.Source = New Uri("/manual.xaml", UriKind.Relative)

Dim account As New FirstFloor.ModernUI.Presentation.Link

account.DisplayName = "My Account"

account.Source = New Uri("/myAccount.xaml", UriKind.Relative)

Dim logout As New FirstFloor.ModernUI.Presentation.Link

logout.DisplayName = "Log Out"

logout.Source = New Uri("/logOut.xaml", UriKind.Relative)

mainMenu(8).Links.Add(manual)

mainMenu(8).Links.Add(account)

mainMenu(8).Links.Add(logout)

Dim currentWindow As ModernWindow = ModernWindow.GetWindow(Me)

currentWindow.Close()

End If

End If

End Sub

Private Sub txtUsername\_GotFocus(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles txtUsername.GotFocus

lblError.Content = ""

End Sub

Private Sub txtPassword\_GotFocus(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles txtPassword.GotFocus

lblError.Content = ""

End Sub

'Check if any employees in database

Private Sub Grid\_Loaded\_1(sender As Object, e As RoutedEventArgs)

db = New DBTransaction()

Dim dataT As DataTable = db.getAllEmployees()

If dataT.Rows.Count = 0 Then

mainGrid.Children.Clear()

End If

End Sub

End Class

**GuestLogin.xaml**

<Page x:Class="GuestLogin"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="300" d:DesignWidth="250"

Title="GuestLogin">

<Grid>

<Button Content="Enter as Guest" Height="42" HorizontalAlignment="Left" Margin="41,81,0,0" Name="btnGuest" VerticalAlignment="Top" Width="167" FontSize="18" />

</Grid>

</Page>

**GuestLogin.xaml.vb**

Imports FirstFloor.ModernUI.Windows.Controls

Imports System.Data

Imports System.Data.SqlClient

Imports System.Security

Class GuestLogin

'Button that's used to log in as a gues

Private Sub btnGuest\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnGuest.Click

'Opening a new window with the guest's options and functions

Dim newWindow As New MainWindow(-1)

Dim mainMenu(4) As FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(1) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(1).DisplayName = "Movies"

mainMenu(2) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(2).DisplayName = "Games"

mainMenu(3) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(3).DisplayName = "Log out"

newWindow.MenuLinkGroups.Add(mainMenu(1))

newWindow.MenuLinkGroups.Add(mainMenu(2))

newWindow.MenuLinkGroups.Add(mainMenu(3))

newWindow.Show()

Dim newMovies As New FirstFloor.ModernUI.Presentation.Link

newMovies.DisplayName = "New Movies"

newMovies.Source = New Uri("/NewMovies.xaml", UriKind.Relative)

Dim trMovies As New FirstFloor.ModernUI.Presentation.Link

trMovies.DisplayName = "Top rated"

trMovies.Source = New Uri("/TopRatedMovies.xaml", UriKind.Relative)

Dim searchMovie As New FirstFloor.ModernUI.Presentation.Link

searchMovie.DisplayName = "Search Movie"

searchMovie.Source = New Uri("/SearchMovie.xaml", UriKind.Relative)

Dim topSoldMovies As New FirstFloor.ModernUI.Presentation.Link

topSoldMovies.DisplayName = "Top Sold Movies"

topSoldMovies.Source = New Uri("/TopSoldMovies.xaml", UriKind.Relative)

mainMenu(1).Links.Add(newMovies)

mainMenu(1).Links.Add(trMovies)

mainMenu(1).Links.Add(topSoldMovies)

mainMenu(1).Links.Add(searchMovie)

Dim newGames As New FirstFloor.ModernUI.Presentation.Link

newGames.DisplayName = "New games"

newGames.Source = New Uri("/NewGames.xaml", UriKind.Relative)

Dim trGames As New FirstFloor.ModernUI.Presentation.Link

trGames.DisplayName = "Top rated"

trGames.Source = New Uri("/TopRatedGames.xaml", UriKind.Relative)

Dim searchGame As New FirstFloor.ModernUI.Presentation.Link

searchGame.DisplayName = "Search game"

searchGame.Source = New Uri("/SearchGame.xaml", UriKind.Relative)

Dim topSoldGames As New FirstFloor.ModernUI.Presentation.Link

topSoldGames.DisplayName = "Top Sold Games"

topSoldGames.Source = New Uri("/TopSoldGames.xaml", UriKind.Relative)

mainMenu(2).Links.Add(newGames)

mainMenu(2).Links.Add(trGames)

mainMenu(2).Links.Add(topSoldGames)

mainMenu(2).Links.Add(searchGame)

Dim logOut As New FirstFloor.ModernUI.Presentation.Link

logOut.DisplayName = "Log out"

logOut.Source = New Uri("/logOut.xaml", UriKind.Relative)

Dim currentWindow As ModernWindow = ModernWindow.GetWindow(Me)

currentWindow.Close()

End Sub

End Class

**Utility.vb**

Imports System.Data.SqlClient

Imports System.IO

'Class that provides utility functions

Public Class Utility

'Function that is fired when the application launches and controls if needed database exists

'If not it creates it

Public Shared Function CheckDatabaseExists(ByVal Server As String, ByVal Database As String) As Boolean

Dim connString As String = ("Data Source=" + (Server + ";Initial Catalog=Master;Integrated Security=True;"))

Dim cmdText As String = ("SELECT \* FROM master.dbo.sysdatabases WHERE NAME='" \_

+ (Database + "'"))

Dim bRet As Boolean = False

Using sqlConnection As SqlConnection = New SqlConnection(connString)

sqlConnection.Open()

Using sqlCmd As SqlCommand = New SqlCommand(cmdText, sqlConnection)

Using reader As SqlDataReader = sqlCmd.ExecuteReader

bRet = reader.HasRows

End Using

End Using

End Using

Return bRet

End Function

'Function to retrieve the database connection string

Public Shared Function GetConnectionString() As String

Dim filePath As String = getDrive() & ":\Program Files\Video Store\VideoStoreManagement\connS.txt"

Dim file As FileInfo = New FileInfo(filePath)

Dim cString As String = file.OpenText().ReadToEnd()

Return cString

End Function

'Function to get the drive where the application is installed

Public Shared Function getDrive() As String

Dim path As String = Directory.GetCurrentDirectory()

path = path.Substring(0, 1)

Return path

End Function

End Class

**DBTransractions.vb**

Imports System.Data

Imports System.Data.SqlClient

'Main class for handling database transactions

Public Class DBTransaction

'Private variable which are used in this class

Private objConnection As SqlConnection

Private objCommand As SqlCommand

Private objCommand1 As SqlCommand

Private objAdapter As SqlDataAdapter

Private objDataTable As DataTable

Private objDataReader As SqlDataReader

'Constructor for the class

Public Sub New()

Dim connString As String = Utility.GetConnectionString()

objConnection = New SqlConnection(connString)

objCommand = New SqlCommand()

objCommand1 = New SqlCommand()

objCommand.Connection = objConnection

objCommand1.Connection = objConnection

objAdapter = New SqlDataAdapter()

objAdapter.SelectCommand = New SqlCommand()

objAdapter.SelectCommand.Connection = objConnection

objAdapter.SelectCommand.CommandType = CommandType.Text

objDataTable = New DataTable()

End Sub

'Function to add a new movie in database

Public Function addMovie(ByVal newMovie As Movies) As Integer

objCommand.CommandText = "INSERT INTO Films(film\_title, film\_categories, film\_release\_date " &

",film\_runtime, film\_rating, film\_description, film\_times\_sold, film\_stock, film\_price, film\_cost, film\_image)" &

"VALUES(@title, @category, @releasedate, @runtime, @rating, @description, @times, @stock, @price, @cost, @image)"

objCommand.Parameters.AddWithValue("@title", newMovie.getName())

objCommand.Parameters.AddWithValue("@category", newMovie.getCategory())

objCommand.Parameters.AddWithValue("@releasedate", newMovie.getReleaseDate())

objCommand.Parameters.AddWithValue("@runtime", newMovie.getRuntime())

objCommand.Parameters.AddWithValue("@rating", newMovie.getRating())

objCommand.Parameters.AddWithValue("@description", newMovie.getDescription())

objCommand.Parameters.AddWithValue("@times", 0)

objCommand.Parameters.AddWithValue("@stock", newMovie.getStock())

objCommand.Parameters.AddWithValue("@price", newMovie.getPrice())

objCommand.Parameters.AddWithValue("@cost", newMovie.getCost())

objCommand.Parameters.AddWithValue("@image", newMovie.getImage())

objConnection.Open()

Dim result As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

Return result

End Function

'Function to add a new game in database

Public Function addGame(ByVal newGame As Games) As Integer

objCommand.CommandText = "INSERT INTO Games(game\_title, game\_categories, game\_release\_date " &

",game\_producer, game\_rating, game\_description, game\_compatibility\_id, game\_times\_sold, game\_stock\_new, game\_stock\_used, game\_price, game\_cost, game\_image)" &

"VALUES(@title, @category, @releasedate, @producer, @rating, @description, @com\_id, @times, @stock, @stock\_u, @price, @cost, @image)"

objCommand.Parameters.AddWithValue("@title", newGame.getName())

objCommand.Parameters.AddWithValue("@category", newGame.getCategory())

objCommand.Parameters.AddWithValue("@releasedate", newGame.getReleaseDate())

objCommand.Parameters.AddWithValue("@producer", newGame.getProducer())

objCommand.Parameters.AddWithValue("@rating", newGame.getRating())

objCommand.Parameters.AddWithValue("@cost", newGame.getCost())

objCommand.Parameters.AddWithValue("@price", newGame.getPrice())

objCommand.Parameters.AddWithValue("@description", newGame.getDescription())

objCommand.Parameters.AddWithValue("@times", 0)

objCommand.Parameters.AddWithValue("@stock", newGame.getStock())

objCommand.Parameters.AddWithValue("@stock\_u", newGame.getStockUsed())

objCommand.Parameters.AddWithValue("@image", newGame.getImage())

objCommand.Parameters.AddWithValue("@com\_id", newGame.getCompatibility())

objConnection.Open()

Dim result As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Return result

End Function

'Function to get all movies from database

Public Function getAllMovies() As DataTable

Dim moviesTable As DataTable = New DataTable()

objAdapter.SelectCommand.CommandText = "SELECT film\_id, film\_title, film\_categories, film\_release\_date, film\_runtime," &

"film\_rating, film\_description, film\_stock, film\_price, film\_cost, film\_image FROM Films"

objConnection.Open()

objAdapter.Fill(moviesTable)

objConnection.Close()

Return moviesTable

End Function

'Function to edit a movie in database

Public Function editMovie(ByVal id As Integer, ByVal name As String, ByVal category As String, ByVal releaseDate As Date,

ByVal runtime As String, ByVal rating As Double, ByVal description As String,

ByVal stock As Integer, ByVal price As Double, ByVal img() As Byte)

objCommand.CommandText = "UPDATE Films SET film\_title = @title, film\_categories = @category, film\_release\_date = @releasedate," &

"film\_runtime = @runtime, film\_rating = @rating, film\_description = @description, film\_stock = @stock," &

"film\_price = @price, film\_image=@img WHERE film\_id = @id"

objCommand.Parameters.AddWithValue("@title", name)

objCommand.Parameters.AddWithValue("@category", category)

objCommand.Parameters.AddWithValue("@releasedate", releaseDate)

objCommand.Parameters.AddWithValue("@runtime", runtime)

objCommand.Parameters.AddWithValue("@rating", rating)

objCommand.Parameters.AddWithValue("@description", description)

objCommand.Parameters.AddWithValue("@stock", stock)

objCommand.Parameters.AddWithValue("@price", price)

objCommand.Parameters.AddWithValue("@id", id)

objCommand.Parameters.AddWithValue("@img", img)

objConnection.Open()

Dim result As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Return result

End Function

'Function to add a new employee in database

Public Function addEmployee(ByVal newEmployee As Employees) As Integer

objCommand.CommandText = "INSERT INTO Employees (employee\_first\_name, employee\_last\_name, employee\_gender, employee\_age," &

"employee\_date\_employment, employee\_salary, employee\_username, employee\_password, employee\_films\_sold, employee\_games\_sold, employee\_bonus) VALUES (@fname, @lname, @gender, @age" &

", @dateEmployment, @salary, @username, @password, @fs, @gs, @bonus)"

objCommand.Parameters.AddWithValue("@fname", newEmployee.getFname)

objCommand.Parameters.AddWithValue("@lname", newEmployee.getLname)

objCommand.Parameters.AddWithValue("@gender", newEmployee.getGender)

objCommand.Parameters.AddWithValue("@age", newEmployee.getAge)

objCommand.Parameters.AddWithValue("@dateEmployment", newEmployee.getDateofEmployment)

objCommand.Parameters.AddWithValue("@salary", newEmployee.getSalary)

objCommand.Parameters.AddWithValue("@username", newEmployee.getUsername)

objCommand.Parameters.AddWithValue("@password", newEmployee.getPass)

objCommand.Parameters.AddWithValue("@fs", 0)

objCommand.Parameters.AddWithValue("@gs", 0)

objCommand.Parameters.AddWithValue("@bonus", 0)

objConnection.Open()

Dim result As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

Return result

End Function

'Function to add a new member in database

Public Function addMember(ByVal newMember As Member) As Integer

objCommand.CommandText = "INSERT INTO Members (member\_first\_name, member\_last\_name, member\_gender, member\_age," &

"member\_registration\_date, member\_email, member\_phone\_number, member\_films\_bought, member\_games\_bought, member\_discount) VALUES" &

"(@fname, @lname, @gender, @age, @registrationDate, @email, @phone, @films\_bought, @games\_bought, @dis)"

objCommand.Parameters.AddWithValue("@fname", newMember.getFname)

objCommand.Parameters.AddWithValue("@lname", newMember.getLname)

objCommand.Parameters.AddWithValue("@gender", newMember.getGender)

objCommand.Parameters.AddWithValue("@age", newMember.getAge)

objCommand.Parameters.AddWithValue("@registrationDate", newMember.getRegistrationDate)

objCommand.Parameters.AddWithValue("@email", newMember.getEmail)

objCommand.Parameters.AddWithValue("@phone", newMember.getPhone)

objCommand.Parameters.AddWithValue("@films\_bought", 0)

objCommand.Parameters.AddWithValue("@games\_bought", 0)

objCommand.Parameters.AddWithValue("@dis", newMember.getDiscount)

objConnection.Open()

Dim result As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

Return result

End Function

'Function to get all games from database

Public Function getAllGames() As DataTable

Dim gamesTable As DataTable = New DataTable()

objAdapter.SelectCommand.CommandText = "SELECT game\_id, game\_title, game\_cost, game\_categories, game\_release\_date, game\_producer," &

"game\_rating, game\_description, game\_stock\_new, game\_stock\_used, game\_price, game\_image, compatibility\_id, compatibility\_name FROM Games " &

"JOIN compatibilities ON game\_compatibility\_id=compatibility\_id"

objConnection.Open()

objAdapter.Fill(gamesTable)

objConnection.Close()

Return gamesTable

End Function

'Function to edit a game in database

Public Function editGame(ByVal id As Integer, ByVal name As String, ByVal category As String, ByVal releaseDate As Date,

ByVal producer As String, ByVal rating As Double, ByVal description As String,

ByVal stock As Integer, ByVal stockUsed As Integer, ByVal price As Double, ByVal comp As Integer, ByVal img() As Byte)

objCommand.CommandText = "UPDATE Games SET game\_title = @title, game\_categories = @category, game\_release\_date = @releasedate," &

"game\_producer = @producer, game\_rating = @rating, game\_description = @description, game\_stock\_new = @stock," &

"game\_stock\_used=@stockU, game\_price = @price, game\_compatibility\_id=@com, game\_image=@img WHERE game\_id = @id"

objCommand.Parameters.AddWithValue("@title", name)

objCommand.Parameters.AddWithValue("@category", category)

objCommand.Parameters.AddWithValue("@releasedate", releaseDate)

objCommand.Parameters.AddWithValue("@producer", producer)

objCommand.Parameters.AddWithValue("@rating", rating)

objCommand.Parameters.AddWithValue("@description", description)

objCommand.Parameters.AddWithValue("@stock", stock)

objCommand.Parameters.AddWithValue("@stockU", stockUsed)

objCommand.Parameters.AddWithValue("@price", price)

objCommand.Parameters.AddWithValue("@id", id)

objCommand.Parameters.AddWithValue("@com", comp)

objCommand.Parameters.AddWithValue("@img", img)

objConnection.Open()

Dim result As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Return result

End Function

'Function to get all employee from database

Public Function getAllEmployees() As DataTable

Dim employeesTable As DataTable = New DataTable()

objAdapter.SelectCommand.CommandText = "SELECT employee\_id, employee\_first\_name, employee\_last\_name," &

"employee\_gender, employee\_age, employee\_date\_employment, employee\_salary, employee\_username, employee\_password, employee\_bonus FROM Employees"

objConnection.Open()

objAdapter.Fill(employeesTable)

objConnection.Close()

Return employeesTable

End Function

'Function to edit an employee in database

Public Function editEmployee(ByVal id As Integer, ByVal fname As String, ByVal lname As String, ByVal gender As Char,

ByVal age As String, ByVal employmentDate As Date, ByVal salary As Double, ByVal username As String,

ByVal pass As String)

objCommand.CommandText = "UPDATE Employees SET employee\_first\_name = @fname, employee\_last\_name = @lname," &

"employee\_gender = @gender, employee\_age = @age, employee\_date\_employment = @employmentDate," &

"employee\_salary = @salary, employee\_username = @username, employee\_password = @pass WHERE employee\_id = @id"

objCommand.Parameters.AddWithValue("@id", id)

objCommand.Parameters.AddWithValue("@fname", fname)

objCommand.Parameters.AddWithValue("@lname", lname)

objCommand.Parameters.AddWithValue("@gender", gender)

objCommand.Parameters.AddWithValue("@age", age)

objCommand.Parameters.AddWithValue("@employmentDate", employmentDate)

objCommand.Parameters.AddWithValue("@salary", salary)

objCommand.Parameters.AddWithValue("@username", username)

objCommand.Parameters.AddWithValue("@pass", pass)

objConnection.Open()

Dim result As Integer = objCommand.ExecuteNonQuery

objConnection.Close()

objCommand.Parameters.Clear()

Return result

End Function

'Function to get 10 new movies from database

Public Function getNewMoviesTable() As DataTable

objAdapter.SelectCommand.CommandText = "SELECT TOP 10 film\_id, film\_title, film\_categories, film\_release\_date," &

"film\_runtime, film\_rating, film\_description, film\_image FROM Films ORDER BY film\_release\_date DESC"

objConnection.Open()

objAdapter.Fill(objDataTable)

objConnection.Close()

Return objDataTable

End Function

'Function to get 10 most rated movies from database

Public Function getMostRateMoview() As DataTable

objAdapter.SelectCommand.CommandText = "SELECT TOP 10 film\_id, film\_title, film\_categories, film\_release\_date," &

"film\_runtime, film\_rating, film\_description, film\_image FROM films ORDER BY film\_rating DESC"

objConnection.Open()

objAdapter.Fill(objDataTable)

objConnection.Close()

Return objDataTable

End Function

'Function to get all members from database

Public Function getAllMembers() As DataTable

Dim membersTable As DataTable = New DataTable()

objAdapter.SelectCommand.CommandText = "SELECT \* FROM Members"

objConnection.Open()

objAdapter.Fill(membersTable)

objConnection.Close()

Return membersTable

End Function

'Function to edit a member in database

Public Function editMember(ByVal id As Integer, ByVal fname As String, ByVal lname As String, ByVal gender As Char,

ByVal age As String, ByVal registrationDate As Date, ByVal email As String, ByVal phone As String)

objCommand.CommandText = "UPDATE Members SET member\_first\_name = @fname, member\_last\_name = @lname," &

"member\_gender = @gender, member\_age = @age, member\_registration\_date = @registrationDate," &

"member\_email = @email, member\_phone\_number = @phone WHERE member\_id = @id"

objCommand.Parameters.AddWithValue("@id", id)

objCommand.Parameters.AddWithValue("@fname", fname)

objCommand.Parameters.AddWithValue("@lname", lname)

objCommand.Parameters.AddWithValue("@gender", gender)

objCommand.Parameters.AddWithValue("@age", age)

objCommand.Parameters.AddWithValue("@registrationDate", registrationDate)

objCommand.Parameters.AddWithValue("@phone", phone)

objCommand.Parameters.AddWithValue("@email", email)

objConnection.Open()

Dim result As Integer = objCommand.ExecuteNonQuery

objConnection.Close()

objCommand.Parameters.Clear()

Return result

End Function

'Function to get 10 new games from database

Public Function getNewGamesTable() As DataTable

objAdapter.SelectCommand.CommandText = "SELECT TOP 10 game\_id, game\_title, game\_categories, game\_release\_date," &

"game\_producer, game\_rating, game\_description, game\_image FROM Games ORDER BY game\_release\_date DESC"

objConnection.Open()

objAdapter.Fill(objDataTable)

objConnection.Close()

Return objDataTable

End Function

'Function to get 10 most rated games from database

Public Function getMostRateGame() As DataTable

objAdapter.SelectCommand.CommandText = "SELECT TOP 10 game\_id, game\_title, game\_categories, game\_release\_date," &

"game\_producer, game\_rating, game\_description, game\_image FROM Games ORDER BY game\_rating DESC"

objConnection.Open()

objAdapter.Fill(objDataTable)

objConnection.Close()

Return objDataTable

End Function

'Function to delete a movie

Public Function deleteMovie(ByVal id As Integer) As Integer

objCommand.CommandText = "DELETE FROM Films WHERE film\_id = @id"

objCommand.Parameters.AddWithValue("@id", id)

objConnection.Open()

Dim rs As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Return rs

End Function

'Function to sell a movie

Public Function sellMovie(ByVal employeeId As Integer, ByVal movieId As Integer, ByVal movieSellDate As Date, ByVal moviePrice As Double,

ByVal movieCost As Double, ByVal memberId As Integer) As Integer

objCommand.CommandText = "INSERT INTO Film\_sales(employee\_sold\_id, film\_sold\_id, film\_sale\_date, film\_price, film\_cost, member\_bought\_id)" &

"VALUES (@employeeId, @movieId, @movieSellDate, @moviePrice, @movieCost, @memberId)"

objCommand.Parameters.AddWithValue("@employeeId", employeeId)

objCommand.Parameters.AddWithValue("@movieId", movieId)

objCommand.Parameters.AddWithValue("@movieSellDate", movieSellDate)

objCommand.Parameters.AddWithValue("@moviePrice", moviePrice)

objCommand.Parameters.AddWithValue("@movieCost", movieCost)

objCommand.Parameters.AddWithValue("@memberId", memberId)

objConnection.Open()

Dim rs As Integer = objCommand.ExecuteNonQuery()

objCommand.Parameters.Clear()

Dim rs1 As Integer = 1

Dim rs2 As Integer = 1

If employeeId <> 0 Then

objCommand.CommandText = "UPDATE Employees SET employee\_films\_sold = employee\_films\_sold + 1 WHERE employee\_id = @employeeId"

objCommand.Parameters.AddWithValue("@employeeId", employeeId)

rs2 = objCommand.ExecuteNonQuery()

objCommand.Parameters.Clear()

Dim bonus As Double = checkBonus(employeeId)

objCommand.CommandText = "UPDATE employees SET employee\_bonus = @bonus WHERE employee\_id=@eid"

objCommand.Parameters.AddWithValue("@bonus", bonus)

objCommand.Parameters.AddWithValue("@eid", employeeId)

objCommand.ExecuteNonQuery()

objCommand.Parameters.Clear()

End If

If memberId <> 0 Then

objCommand.CommandText = "UPDATE members SET member\_films\_bought = member\_films\_bought + 1 WHERE member\_id = @memberId"

objCommand.Parameters.AddWithValue("@memberId", memberId)

rs1 = objCommand.ExecuteNonQuery()

End If

objConnection.Close()

objCommand.Parameters.Clear()

objCommand.CommandText = "UPDATE films SET film\_times\_sold = film\_times\_sold + 1 WHERE film\_id=@id"

objCommand.Parameters.AddWithValue("@id", movieId)

objConnection.Open()

objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Return rs + rs1 + rs2

End Function

'Function to decrement film stock

Public Function decrementFilmStock(ByVal movieId As Integer) As Integer

objCommand.CommandText = "UPDATE Films SET film\_stock = film\_stock - 1 WHERE film\_id = @movieId"

objCommand.Parameters.AddWithValue("@movieId", movieId)

objConnection.Open()

Dim rs As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Return rs

End Function

'Function to get all film sales between two dates

Public Function getAllmovieSalesTable(ByVal date1 As Date, ByVal date2 As Date) As DataTable

objAdapter.SelectCommand.CommandText = "SELECT film\_title, film\_id, Film\_sales.film\_price, Film\_sales.film\_cost, film\_sale\_date FROM Films JOIN " &

"Film\_sales ON film\_id = film\_sold\_id WHERE film\_sale\_date <= @edate AND film\_sale\_date >= @bdate ORDER BY film\_sale\_date DESC"

objAdapter.SelectCommand.Parameters.AddWithValue("@bdate", date1)

objAdapter.SelectCommand.Parameters.AddWithValue("@edate", date2)

objConnection.Open()

objAdapter.Fill(objDataTable)

objConnection.Close()

objAdapter.SelectCommand.Parameters.Clear()

Return objDataTable

End Function

'Function to get total income from film sales between two dates

Public Function getTotalMovieSalesbyDate(ByVal date1 As Date, ByVal date2 As Date) As Double

objCommand.CommandText = "SELECT SUM(film\_price) FROM Film\_sales WHERE film\_sale\_date <= @edate AND film\_sale\_date >= @bdate"

objCommand.Parameters.AddWithValue("@bdate", date1)

objCommand.Parameters.AddWithValue("@edate", date2)

objConnection.Open()

objDataReader = objCommand.ExecuteReader()

objDataReader.Read()

Dim total As Double = 0.0

If objDataReader.HasRows Then

total = objDataReader.Item(0)

End If

objConnection.Close()

objCommand.Parameters.Clear()

Return total

End Function

'Get number of rented movies from a member of the store

Public Function getRentedMoviesFromMemberId(ByVal id As Integer) As Integer

objCommand.CommandText = "SELECT COUNT(member\_rent\_id) FROM film\_rent WHERE member\_rent\_id=@id AND rent\_status=1"

objCommand.Parameters.AddWithValue("@id", id)

Dim t As Integer = 0

objConnection.Open()

objDataReader = objCommand.ExecuteReader()

objDataReader.Read()

If objDataReader.HasRows Then

t = objDataReader.Item(0)

End If

objConnection.Close()

objCommand.Parameters.Clear()

Return t

End Function

'Function to add a new rent entry in database

Public Function rentMovie(ByVal fID As Integer, ByVal mID As Integer, ByVal price As Double) As Integer

objCommand.CommandText = "INSERT INTO film\_rent(film\_rent\_id, member\_rent\_id, rent\_date, return\_date, film\_rent\_price, rent\_status) " &

"VALUES(@fid, @mid, @rd, @red, @pr, @st)"

objCommand.Parameters.AddWithValue("@fid", fID)

objCommand.Parameters.AddWithValue("@mid", mID)

objCommand.Parameters.AddWithValue("@rd", Date.Now())

objCommand.Parameters.AddWithValue("@red", Date.Now.AddDays(3))

objCommand.Parameters.AddWithValue("@pr", price)

objCommand.Parameters.AddWithValue("@st", True)

objConnection.Open()

Dim rs As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Return rs

End Function

'Function to get all rented movies from database

Public Function getAllRentedMovies() As DataTable

objDataTable.Dispose()

objAdapter.SelectCommand.CommandText = "SELECT film\_id, film\_title, film\_stock, member\_id, member\_first\_name, member\_last\_name, rent\_date, return\_date FROM " &

"films JOIN film\_rent ON film\_id=film\_rent\_id JOIN members ON member\_id=member\_rent\_id WHERE rent\_status=1"

objConnection.Open()

objAdapter.Fill(objDataTable)

objConnection.Close()

Return objDataTable

End Function

'Function to return a rented movie in database

Public Function returnMovie(ByVal fID As Integer, ByVal mID As Integer, ByVal rDate As Date) As Integer

objCommand.CommandText = "UPDATE film\_rent SET rent\_status=0 WHERE film\_rent\_id=@fid AND member\_rent\_id=@mid AND rent\_date=@rd"

objCommand.Parameters.AddWithValue("@fid", fID)

objCommand.Parameters.AddWithValue("@mid", mID)

objCommand.Parameters.AddWithValue("@rd", rDate)

objConnection.Open()

Dim rs As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Return rs

End Function

'Function to postpone a rent

Public Function postponeRent(ByVal fID As Integer, ByVal mID As Integer, ByVal rDate As Date) As Integer

objCommand.CommandText = "UPDATE film\_rent SET return\_date=DATEADD(day, 1, return\_date), film\_rent\_price=film\_rent\_price+50 WHERE film\_rent\_id=@fid AND member\_rent\_id=@mid AND rent\_date=@rd"

objCommand.Parameters.AddWithValue("@fid", fID)

objCommand.Parameters.AddWithValue("@mid", mID)

objCommand.Parameters.AddWithValue("@rd", rDate)

objConnection.Open()

Dim rs As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Return rs

End Function

'Function to increment movie stock

Public Function incrementMovieStock(ByVal fID As Integer) As Integer

objCommand.CommandText = "UPDATE Films SET film\_stock = film\_stock + 1 WHERE film\_id = @movieId"

objCommand.Parameters.AddWithValue("@movieId", fID)

objConnection.Open()

Dim rs As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Return rs

End Function

'Function to get all game compatibilities from database

Public Function getCompatibilities() As DataTable

objAdapter.SelectCommand.CommandText = "SELECT \* FROM compatibilities"

objConnection.Open()

objAdapter.Fill(objDataTable)

objConnection.Close()

Return objDataTable

End Function

'Function to add a new compatibility in database

Public Function insertCompatibility(ByVal name As String) As Integer

objCommand.CommandText = "INSERT INTO compatibilities(compatibility\_name) VALUES(@name)"

objCommand.Parameters.AddWithValue("@name", name)

objConnection.Open()

Dim rs As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Return rs

End Function

'Function to update a compatibility in database

Public Function updateCompatibility(ByVal id As Integer, ByVal name As String) As Integer

objCommand.CommandText = "UPDATE compatibilities SET compatibility\_name=@name WHERE compatibility\_id=@id"

objCommand.Parameters.AddWithValue("@name", name)

objCommand.Parameters.AddWithValue("@id", id)

objConnection.Open()

Dim rs As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Return rs

End Function

'Function to delete a member from database

Public Function deleteMember(ByVal id As Integer) As Integer

objCommand.CommandText = "DELETE FROM Members WHERE member\_id = @id"

objCommand.Parameters.AddWithValue("@id", id)

objConnection.Open()

Dim rs As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Return rs

End Function

'Function to delete a game from database

Public Function deleteGame(ByVal id As Integer) As Integer

objCommand.CommandText = "DELETE FROM Games WHERE game\_id = @id"

objCommand.Parameters.AddWithValue("@id", id)

objConnection.Open()

Dim rs As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Return rs

End Function

'Function to delete an employee from database

Public Function deleteEmployee(ByVal id As Integer) As Integer

objCommand.CommandText = "DELETE FROM Employees WHERE employee\_id = @id"

objCommand.Parameters.AddWithValue("@id", id)

objConnection.Open()

Dim rs As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Return rs

End Function

'Subroutine to check if members have more than one year

'registered in our video store and sets their discount

Public Sub checkMembers()

objAdapter.SelectCommand.CommandText = "SELECT member\_id, member\_registration\_date FROM members"

objAdapter.Fill(objDataTable)

Dim index As Integer = 0

While index < objDataTable.Rows.Count

Dim rDate As Date = objDataTable.Rows(index).Item("member\_registration\_date")

Dim diff As Integer = DateDiff(DateInterval.Day, rDate, Date.Now)

If diff >= 365 Then

updateDiscount(objDataTable.Rows(index).Item("member\_id"))

End If

index += 1

End While

End Sub

'Function to update member's discount

Private Sub updateDiscount(ByVal member\_id As Integer)

objCommand1.CommandText = "UPDATE members SET member\_discount=10 WHERE member\_id=@id"

objCommand1.Parameters.AddWithValue("@id", member\_id)

objConnection.Open()

objCommand1.ExecuteNonQuery()

objConnection.Close()

objCommand1.Parameters.Clear()

End Sub

'Function to get film sales of an employee

Public Function getMyFilmSales(ByVal id As Integer) As DataTable

objAdapter.SelectCommand.CommandText = "SELECT film\_sale\_date, film\_sales.film\_price, film\_title FROM films JOIN film\_sales ON " &

"film\_id=film\_sold\_id WHERE employee\_sold\_id=@id AND YEAR([film\_sale\_date])=YEAR(GETDATE()) AND MONTH([film\_sale\_date])=MONTH(GETDATE())" &

"AND DAY([film\_sale\_date])<=DAY(GETDATE())"

objAdapter.SelectCommand.Parameters.AddWithValue("@id", id)

objConnection.Open()

objAdapter.Fill(objDataTable)

objConnection.Close()

Return objDataTable

End Function

'Function to get game sales of an employee

Public Function getMyGameSales(ByVal id As Integer) As DataTable

objAdapter.SelectCommand.CommandText = "SELECT game\_sale\_date, games\_sales.game\_price, game\_title FROM games JOIN games\_sales ON " &

"game\_id=game\_sold\_id WHERE employee\_sold\_id=@id AND YEAR([game\_sale\_date])=YEAR(GETDATE()) AND MONTH([game\_sale\_date])=MONTH(GETDATE())" &

"AND DAY([game\_sale\_date])<=DAY(GETDATE())"

objAdapter.SelectCommand.Parameters.AddWithValue("@id", id)

objConnection.Open()

objAdapter.Fill(objDataTable)

objConnection.Close()

Return objDataTable

End Function

'Function to get all game sales between two dates

Public Function getAllGameSalesTable(ByVal d1 As Date, ByVal d2 As Date) As DataTable

objAdapter.SelectCommand.CommandText = "SELECT game\_title, game\_id, games\_sales.game\_price, games\_sales.game\_cost, game\_sale\_date FROM games JOIN " &

"games\_sales ON game\_id = game\_sold\_id WHERE game\_sale\_date <= @edate AND game\_sale\_date >= @bdate ORDER BY game\_sale\_date DESC"

objAdapter.SelectCommand.Parameters.AddWithValue("@bdate", d1)

objAdapter.SelectCommand.Parameters.AddWithValue("@edate", d2)

objConnection.Open()

objAdapter.Fill(objDataTable)

objConnection.Close()

objAdapter.SelectCommand.Parameters.Clear()

Return objDataTable

End Function

'Function to sell a game

Public Function sellGame(ByVal gId As Integer, ByVal eId As Integer, ByVal mID As Integer, ByVal price As Double, ByVal cost As Double, ByVal gNew As Boolean) As Integer

objCommand.CommandText = "INSERT INTO games\_sales(employee\_sold\_id, game\_sold\_id, game\_sale\_date, game\_price, game\_cost, member\_bought\_id) " &

"VALUES(@eid, @gid, @dt, @pr, @co, @mid)"

objCommand.Parameters.AddWithValue("@eid", eId)

objCommand.Parameters.AddWithValue("@gid", gId)

objCommand.Parameters.AddWithValue("@dt", Date.Now())

objCommand.Parameters.AddWithValue("@pr", price)

objCommand.Parameters.AddWithValue("@co", cost)

objCommand.Parameters.AddWithValue("@mid", mID)

objConnection.Open()

Dim rs As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Dim status As String = ""

If gNew = True Then

status = "game\_stock\_new"

Else

status = "game\_stock\_used"

End If

objCommand.CommandText = "UPDATE games SET " & status & "=" & status & "-1 WHERE game\_id=@gid"

objCommand.Parameters.AddWithValue("@gid", gId)

objConnection.Open()

objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

objCommand.CommandText = "UPDATE games SET game\_times\_sold = game\_times\_sold + 1 WHERE game\_id=@id"

objCommand.Parameters.AddWithValue("@id", gId)

objConnection.Open()

objCommand.ExecuteNonQuery()

objConnection.Close()

If mID <> 0 Then

objCommand.CommandText = "UPDATE members SET member\_games\_bought = member\_games\_bought + 1 WHERE member\_id = @memberId"

objCommand.Parameters.AddWithValue("@memberId", mID)

objConnection.Open()

objCommand.ExecuteNonQuery()

objConnection.Close()

End If

If eId <> 0 Then

objConnection.Open()

objCommand.CommandText = "UPDATE Employees SET employee\_games\_sold = employee\_games\_sold + 1 WHERE employee\_id = @employeeId"

objCommand.Parameters.AddWithValue("@employeeId", eId)

objCommand.ExecuteNonQuery()

objCommand.Parameters.Clear()

Dim bonus As Double = checkBonus(eId)

objCommand.CommandText = "UPDATE employees SET employee\_bonus = @bonus WHERE employee\_id=@eid"

objCommand.Parameters.AddWithValue("@bonus", bonus)

objCommand.Parameters.AddWithValue("@eid", eId)

objCommand.ExecuteNonQuery()

objConnection.Close()

End If

objCommand.Parameters.Clear()

Return rs

End Function

'Function to get all rents which have passed their return dates

Public Function getPassedRents() As DataTable

objAdapter.SelectCommand.CommandText = "SELECT member\_first\_name, member\_last\_name, member\_email, film\_title FROM members JOIN film\_rent " &

"ON member\_id=member\_rent\_id JOIN films ON film\_id=film\_rent\_id WHERE return\_date < GETDATE() AND rent\_status=1"

objConnection.Open()

objAdapter.Fill(objDataTable)

objConnection.Close()

Return objDataTable

End Function

'Function to modify administrator password

Public Function modifyAdminPassword(ByVal password As String) As Integer

objCommand.CommandText = "UPDATE administrator SET admin\_password=@pas WHERE admin\_id=1"

objCommand.Parameters.AddWithValue("@pas", password)

Dim rs As Integer

objConnection.Open()

rs = objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Return rs

End Function

'Function to modify employee password

Public Function modifyEmployeePassword(ByVal password As String, ByVal id As Integer) As Integer

objCommand.CommandText = "UPDATE employees SET employee\_password=@pas WHERE employee\_id=@id"

objCommand.Parameters.AddWithValue("@pas", password)

objCommand.Parameters.AddWithValue("@id", id)

Dim rs As Integer

objConnection.Open()

rs = objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Return rs

End Function

'Function to get most sold games

Public Function getMostSoldGames() As DataTable

objAdapter.SelectCommand.CommandText = "SELECT TOP 10 \* FROM games ORDER BY game\_times\_sold DESC"

objConnection.Open()

objAdapter.Fill(objDataTable)

objConnection.Close()

Return objDataTable

End Function

'Function to get most sold movies

Public Function getMostSoldMoview() As DataTable

objAdapter.SelectCommand.CommandText = "SELECT TOP 10 \* FROM films ORDER BY film\_times\_sold DESC"

objConnection.Open()

objAdapter.Fill(objDataTable)

objConnection.Close()

Return objDataTable

End Function

'Function to check employee bonuses

Public Function checkBonus(ByVal eId As Integer) As Double

objAdapter.SelectCommand.CommandText = "SELECT employee\_films\_sold, employee\_games\_sold, employee\_salary FROM employees " &

"WHERE employee\_id=@eid"

objAdapter.SelectCommand.Parameters.AddWithValue("@eid", eId)

Dim bonus As Double = 0.0

Dim sum As Integer = 0

objAdapter.Fill(objDataTable)

sum = objDataTable.Rows(0).Item("employee\_films\_sold") + objDataTable.Rows(0).Item("employee\_games\_sold")

If sum Mod 200 = 0 Then

bonus = objDataTable.Rows(0).Item(2) \* 0.1

End If

objAdapter.SelectCommand.Parameters.Clear()

Return bonus

End Function

'Function to get total movie sales income for current month

Public Function getTotalMovieSales(ByVal currentDate As Date) As Double

objCommand.CommandText = "SELECT SUM(film\_price) as total FROM film\_sales WHERE " &

"YEAR([film\_sale\_date])=YEAR(GETDATE()) AND MONTH([film\_sale\_date])=MONTH(GETDATE())" &

"AND DAY([film\_sale\_date])<=DAY(GETDATE())"

objConnection.Open()

objDataReader = objCommand.ExecuteReader()

objDataReader.Read()

Dim total As Double = objDataReader.Item(0)

objConnection.Close()

Return total

End Function

'Function to get total game sales income for current month

Public Function getTotalGamesSales(ByVal currentDate As Date) As Double

objCommand.CommandText = "SELECT SUM(game\_price) as total FROM games\_sales WHERE " &

"YEAR([game\_sale\_date])=YEAR(GETDATE()) AND MONTH([game\_sale\_date])=MONTH(GETDATE())" &

"AND DAY([game\_sale\_date])<=DAY(GETDATE())"

objConnection.Open()

objDataReader = objCommand.ExecuteReader()

objDataReader.Read()

Dim total As Double = objDataReader.Item(0)

objConnection.Close()

Return total

End Function

'Function to get total game costs for current month

Public Function getTotalGamesCost(ByVal currentDate As Date) As Double

objCommand.CommandText = "SELECT SUM(game\_cost) as total FROM games\_sales WHERE " &

"YEAR([game\_sale\_date])=YEAR(GETDATE()) AND MONTH([game\_sale\_date])=MONTH(GETDATE())" &

"AND DAY([game\_sale\_date])<=DAY(GETDATE())"

objConnection.Open()

objDataReader = objCommand.ExecuteReader()

objDataReader.Read()

Dim total As Double = objDataReader.Item(0)

objConnection.Close()

Return total

End Function

'Function to get total movie costs for current month

Public Function getTotalMoviesCost(ByVal currentDate As Date) As Double

objCommand.CommandText = "SELECT SUM(film\_cost) as total FROM film\_sales WHERE " &

"YEAR([film\_sale\_date])=YEAR(GETDATE()) AND MONTH([film\_sale\_date])=MONTH(GETDATE())" &

"AND DAY([film\_sale\_date])<=DAY(GETDATE())"

objConnection.Open()

objDataReader = objCommand.ExecuteReader()

objDataReader.Read()

Dim total As Double = objDataReader.Item(0)

objConnection.Close()

Return total

End Function

'Function to get total movie rent income for current month

Public Function getTotalMovieRents(ByVal currentDate As Date) As Double

objCommand.CommandText = "SELECT SUM(film\_rent\_price) as total FROM film\_rent WHERE " &

"YEAR([rent\_date])=YEAR(GETDATE()) AND MONTH([rent\_date])=MONTH(GETDATE())" &

"AND DAY([rent\_date])<=DAY(GETDATE())"

objConnection.Open()

objDataReader = objCommand.ExecuteReader()

objDataReader.Read()

Dim total As Double = objDataReader.Item(0)

objConnection.Close()

Return total

End Function

'Subroutine to reset employee bonus

Public Sub zeroBonuses()

objCommand.CommandText = "UPDATE employees SET employee\_bonus=0"

objConnection.Open()

objCommand.ExecuteNonQuery()

objConnection.Close()

End Sub

End Class

**b) Adding a new item to the program**

1. Adding a movie

**addMovies.xaml**

<Page x:Class="addMovies"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="400" d:DesignWidth="700"

Title="addMovies">

<Grid Name="Grid1">

<Label Content="Name\*:" Height="24" HorizontalAlignment="Left" Margin="23,26,0,0" Name="Label1" VerticalAlignment="Top" Width="74" />

<Label Content="Category:" Height="24" HorizontalAlignment="Left" Margin="23,58,0,0" Name="Label2" VerticalAlignment="Top" />

<Label Content="Release date:" Height="24" HorizontalAlignment="Left" Margin="23,90,0,0" Name="Label3" VerticalAlignment="Top" />

<Label Content="Runtime:" Height="24" HorizontalAlignment="Left" Margin="23,122,0,0" Name="Label4" VerticalAlignment="Top" />

<Label Content="Rating(Numeric):" Height="24" HorizontalAlignment="Left" Margin="23,154,0,0" Name="Label5" VerticalAlignment="Top" />

<Label Content="Description:" Height="24" HorizontalAlignment="Left" Margin="23,186,0,0" Name="Label6" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="123,23,0,0" Name="txtName" VerticalAlignment="Top" Width="200" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="123,55,0,0" Name="txtCategory" VerticalAlignment="Top" Width="200" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="123,151,0,0" Name="txtRating" VerticalAlignment="Top" Width="200" />

<DatePicker Height="25" HorizontalAlignment="Left" Margin="123,86,0,0" Name="dtpReleaseDate" VerticalAlignment="Top" Width="200" />

<Button Content="Add Movie" Height="25" HorizontalAlignment="Left" Margin="448,220,0,0" Name="btnAddMovie" VerticalAlignment="Top" Width="200" />

<RichTextBox Height="100" HorizontalAlignment="Left" Margin="123,186,0,0" Name="txtDescription" VerticalAlignment="Top" Width="200" />

<Label Content="Stock(Numeric):" Height="28" HorizontalAlignment="Left" Margin="345,23,0,0" Name="Label7" VerticalAlignment="Top" />

<Label Content="Cost(Numeric):" Height="28" HorizontalAlignment="Left" Margin="345,58,0,0" Name="Label8" VerticalAlignment="Top" />

<Label Content="Price(Numeric):" Height="28" HorizontalAlignment="Left" Margin="343,90,0,0" Name="Label9" VerticalAlignment="Top" />

<Label Content="Image:" Height="28" HorizontalAlignment="Left" Margin="341,122,0,0" Name="Label10" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="448,55,0,0" Name="txtCost" VerticalAlignment="Top" Width="200" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="448,23,0,0" Name="txtStock" VerticalAlignment="Top" Width="200" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="448,86,0,0" Name="txtPrice" VerticalAlignment="Top" Width="200" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="448,119,0,0" Name="txtImage" VerticalAlignment="Top" Width="200" IsEnabled="False" />

<Button Content="Browse" Height="27" HorizontalAlignment="Left" Margin="448,151,0,0" Name="btnBrowse" VerticalAlignment="Top" Width="82" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="123,119,0,0" Name="txtRuntime" VerticalAlignment="Top" Width="200" />

<Label Height="28" HorizontalAlignment="Left" Margin="448,258,0,0" Name="lblError" VerticalAlignment="Top" Width="200" Foreground="Red" />

<Label Height="28" HorizontalAlignment="Left" Margin="448,258,0,0" Name="lblSuccess" VerticalAlignment="Top" Width="200" FontSize="16" Foreground="#FF00FA00" />

</Grid>

</Page>

**Movies.vb**

'Class used for to get all the data from the inputs of the administrator or employee when adding a movie and put it to database

Public Class Movies

Private cName As String

Private cCategory As String

Private cReleaseDate As Date

Private cRuntime As String

Private cRating As Double

Private cDescription As String

Private cStock As Integer

Private cPrice As Double

Private cCost As Double

Private cImage() As Byte

Public Sub New(ByVal name As String, ByVal category As String, ByVal releaseDate As Date, ByVal runtime As String, ByVal rating As Double,

ByVal description As String, ByVal stock As Integer, ByVal price As Double, ByVal cost As Double, ByVal img() As Byte)

cName = name

cCategory = category

cReleaseDate = releaseDate

cRuntime = runtime

cRating = rating

cDescription = description

cStock = stock

cPrice = price

cCost = cost

cImage = img

End Sub

Public Function getName() As String

Return cName

End Function

Public Function getCategory() As String

Return cCategory

End Function

Public Function getReleaseDate() As Date

Return cReleaseDate

End Function

Public Function getRuntime() As String

Return cRuntime

End Function

Public Function getRating() As Double

Return cRating

End Function

Public Function getDescription() As String

Return cDescription

End Function

Public Function getStock() As String

Return cStock

End Function

Public Function getPrice() As String

Return cPrice

End Function

Public Function getCost() As String

Return cCost

End Function

Public Function getImage() As Byte()

Return cImage

End Function

End Class

**addMovies.xaml.vb**

Imports System.IO

Class addMovies

'Button used to add the data in the fields in the movies table in the database

Private Sub btnAddMovie\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnAddMovie.Click

Dim name As String = txtName.Text

Dim category As String = txtCategory.Text

Dim releaseDate As Date = dtpReleaseDate.Text

Dim runtime As String = txtRuntime.Text

Dim rating As String = txtRating.Text

Dim txtRange As New TextRange(txtDescription.Document.ContentStart, txtDescription.Document.ContentEnd)

Dim description As String = txtRange.Text

Dim stock As String = txtStock.Text

Dim price As String = txtPrice.Text

Dim cost As String = txtCost.Text

Dim img() As Byte = Nothing

If txtImage.Text.Length <> 0 Then

Dim fs As FileStream = New FileStream(txtImage.Text, FileMode.Open, FileAccess.Read)

Dim br As BinaryReader = New BinaryReader(fs)

img = br.ReadBytes(fs.Length())

End If

If name.Length = 0 Or category.Length = 0 Or runtime.Length = 0 Or rating.Length = 0 Or

description.Length = 0 Or stock.Length = 0 Or price.Length = 0 Or cost.Length = 0 Then

lblError.Content = "Please fill in all the fields!"

Else

Dim newMovie As Movies = New Movies(name, category, releaseDate, runtime, CType(rating, Double),

description, CType(stock, Integer), CType(price, Double), CType(cost, Double), img)

Dim db As DBTransaction = New DBTransaction

If db.addMovie(newMovie) = 1 Then

lblError.Content = ""

lblSuccess.Content = "Insert is successful"

resetFields()

Else

lblError.Content = "Something happened. Please try again later."

End If

End If

End Sub

'Button used to browse an image from the computer

Private Sub btnBrowse\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnBrowse.Click

Dim dlg As Microsoft.Win32.OpenFileDialog = New Microsoft.Win32.OpenFileDialog

dlg.DefaultExt = ".jpg"

dlg.Filter = "Image File (\*.png, \*.jpg)|\*.png; \*.jpg"

Dim result As Boolean = dlg.ShowDialog

If result = True Then

txtImage.Text = dlg.FileName

End If

End Sub

'Sub used to set the present date in the date picker

Private Sub Grid1\_Loaded(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles Grid1.Loaded

dtpReleaseDate.SelectedDate = DateTime.Today

End Sub

'Sub used for clearing all the fields

Private Sub resetFields()

txtName.Text = ""

txtCategory.Text = ""

txtCost.Text = ""

txtPrice.Text = ""

txtStock.Text = ""

txtRuntime.Text = ""

txtImage.Text = ""

txtDescription.Document.Blocks.Clear()

txtRating.Text = ""

End Sub

End Class

2. Adding/modifying a compatibility

**addCompatibility.xaml**

<Page x:Class="AddCompatibility"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="300" d:DesignWidth="500"

Title="AddCompatibility">

<Grid Loaded="Grid\_Loaded\_1">

<Label Content="Add New Compatibility" HorizontalAlignment="Left" Margin="10,238,0,0" VerticalAlignment="Top" FontSize="18"/>

<TextBox x:Name="txtCompatibility" HorizontalAlignment="Left" Height="23" Margin="10,267,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="184"/>

<DataGrid IsReadOnly="True" AutoGenerateColumns="False" x:Name="dgCompatibility" HorizontalAlignment="Left" Margin="10,10,0,0" VerticalAlignment="Top" Width="280" Height="223">

<DataGrid.Columns>

<DataGridTextColumn Width="70" Header="ID" Binding="{Binding Path=compatibility\_id}"></DataGridTextColumn>

<DataGridTextColumn Width="210" Header="Compatibility" Binding="{Binding Path=compatibility\_name}"></DataGridTextColumn>

</DataGrid.Columns>

</DataGrid>

<Button x:Name="btnAdd" Content="Add" HorizontalAlignment="Left" Margin="199,264,0,0" VerticalAlignment="Top" Width="91"/>

<Label x:Name="lblError" Foreground="Red" Content="" HorizontalAlignment="Left" Margin="311,264,0,0" VerticalAlignment="Top" Width="179" Height="26" FontSize="16"/>

<Label x:Name="lblSuccess" Foreground="green" Content="" HorizontalAlignment="Left" Margin="311,264,0,0" VerticalAlignment="Top" Width="179" Height="26" FontSize="16"/>

<Label Content="ID" HorizontalAlignment="Left" Margin="311,10,0,0" VerticalAlignment="Top"/>

<Label Content="Name" HorizontalAlignment="Left" Margin="311,41,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtID" IsReadOnly="True" HorizontalAlignment="Left" Height="23" Margin="370,10,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="120"/>

<TextBox x:Name="txtName" HorizontalAlignment="Left" Height="23" Margin="370,38,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="120"/>

<Button x:Name="btnModify" Content="Modify" HorizontalAlignment="Left" Margin="389,79,0,0" VerticalAlignment="Top" Width="75"/>

</Grid>

</Page>

**addCompatibility.xaml.vb**

Imports System.Data

Class AddCompatibility

Private db As DBTransaction

Private dTableComp As DataTable

Private Sub Grid\_Loaded\_1(sender As Object, e As RoutedEventArgs)

db = New DBTransaction()

dTableComp = New DataTable()

dTableComp = db.getCompatibilities()

dgCompatibility.ItemsSource = dTableComp.DefaultView

lblError.Content = ""

lblSuccess.Content = ""

txtCompatibility.Text = ""

End Sub

'Subroutine to collect information about the new compatibility and call a funtion to insert it into the database

Private Sub btnAdd\_Click(sender As Object, e As RoutedEventArgs) Handles btnAdd.Click

If txtCompatibility.Text.Length <> 0 Then

Dim name As String = txtCompatibility.Text.ToString

Dim rs As Integer = db.insertCompatibility(name)

If rs = 1 Then

lblSuccess.Content = "Inserted with success."

txtCompatibility.Text = ""

txtID.Text = ""

txtName.Text = ""

dTableComp.Clear()

dgCompatibility.ItemsSource = dTableComp.DefaultView

dTableComp = db.getCompatibilities()

dgCompatibility.ItemsSource = dTableComp.DefaultView

Else

lblError.Content = "Error"

End If

Else

lblError.Content = "Please fill the name."

End If

End Sub

'Sub to clear fields when compatibility text field has focus

Private Sub txtCompatibility\_GotFocus(sender As Object, e As RoutedEventArgs) Handles txtCompatibility.GotFocus

lblError.Content = ""

lblSuccess.Content = ""

End Sub

Private Sub dgCompatibility\_MouseDoubleClick(sender As Object, e As MouseButtonEventArgs) Handles dgCompatibility.MouseDoubleClick

Dim dr As DataRowView = dgCompatibility.SelectedItem

txtID.Text = dr.Item(0)

txtName.Text = dr.Item(1)

lblError.Content = ""

lblSuccess.Content = ""

End Sub

'Subroutine to collect information about the compatibility and call a funtion to modify it

Private Sub btnModify\_Click(sender As Object, e As RoutedEventArgs) Handles btnModify.Click

If txtID.Text.Length <> 0 Then

Dim id As Integer = CType(txtID.Text, Integer)

Dim name As String = txtName.Text.ToString

Dim rs As Integer = db.updateCompatibility(id, name)

If rs = 1 Then

lblSuccess.Content = "Updated."

txtID.Text = ""

txtName.Text = ""

dTableComp.Clear()

dgCompatibility.ItemsSource = dTableComp.DefaultView

dTableComp = db.getCompatibilities()

dgCompatibility.ItemsSource = dTableComp.DefaultView

Else

lblError.Content = "Error, try again."

End If

Else

lblError.Content = "Select a Compatibility"

End If

End Sub

End Class

3. Adding a game

**addGame.xaml**

<Page x:Class="addGame"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="400" d:DesignWidth="700"

Title="addGame">

<Grid Name="Grid1">

<Label Content="Name:" Height="24" HorizontalAlignment="Left" Margin="23,26,0,0" Name="Label1" VerticalAlignment="Top" Width="74" />

<Label Content="Category:" Height="24" HorizontalAlignment="Left" Margin="23,58,0,0" Name="Label2" VerticalAlignment="Top" />

<Label Content="Release date:" Height="24" HorizontalAlignment="Left" Margin="23,90,0,0" Name="Label3" VerticalAlignment="Top" />

<Label Content="Producer:" Height="24" HorizontalAlignment="Left" Margin="23,122,0,0" Name="Label4" VerticalAlignment="Top" />

<Label Content="Rating(Numeric):" Height="24" HorizontalAlignment="Left" Margin="23,154,0,0" Name="Label5" VerticalAlignment="Top" />

<Label Content="Description:" Height="24" HorizontalAlignment="Left" Margin="23,186,0,0" Name="Label6" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="123,23,0,0" Name="txtName" VerticalAlignment="Top" Width="200" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="123,55,0,0" Name="txtCategory" VerticalAlignment="Top" Width="200" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="123,151,0,0" Name="txtRating" VerticalAlignment="Top" Width="200" />

<DatePicker Height="25" HorizontalAlignment="Left" Margin="123,86,0,0" Name="dtpReleaseDate" VerticalAlignment="Top" Width="200" />

<Button Content="Add Game" Height="25" HorizontalAlignment="Left" Margin="471,261,0,0" Name="btnAddGame" VerticalAlignment="Top" Width="200" />

<RichTextBox Height="100" HorizontalAlignment="Left" Margin="123,186,0,0" Name="txtDescription" VerticalAlignment="Top" Width="200" />

<Label Content="Stock(New Numeric):" Height="28" HorizontalAlignment="Left" Margin="341,54,0,0" Name="Label7" VerticalAlignment="Top" />

<Label Content="Cost(Numeric):" Height="28" HorizontalAlignment="Left" Margin="341,121,0,0" Name="Label8" VerticalAlignment="Top" />

<Label Content="Price(Numeric):" Height="28" HorizontalAlignment="Left" Margin="341,153,0,0" Name="Label9" VerticalAlignment="Top" />

<Label Content="Image:" Height="28" HorizontalAlignment="Left" Margin="341,186,0,0" Name="Label10" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="471,122,0,0" Name="txtCost" VerticalAlignment="Top" Width="179" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="471,55,0,0" Name="txtStock" VerticalAlignment="Top" Width="179" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="471,154,0,0" Name="txtPrice" VerticalAlignment="Top" Width="179" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="471,187,0,0" Name="txtImage" VerticalAlignment="Top" Width="200" IsEnabled="False" />

<Button Content="Browse" Height="27" HorizontalAlignment="Left" Margin="471,215,0,0" Name="btnBrowse" VerticalAlignment="Top" Width="82" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="123,119,0,0" Name="txtProducer" VerticalAlignment="Top" Width="200" />

<Label Height="28" HorizontalAlignment="Left" Margin="471,304,0,0" Name="lblError" VerticalAlignment="Top" Width="200" Foreground="Red" />

<Label Height="28" HorizontalAlignment="Left" Margin="471,304,0,0" Name="lblSuccess" VerticalAlignment="Top" Width="200" FontSize="16" Foreground="#FF00FA00" />

<Label Content="Compatibility:" Height="28" HorizontalAlignment="Left" Margin="339,23,0,0" Name="Label11" VerticalAlignment="Top" />

<ComboBox Height="23" HorizontalAlignment="Left" Margin="471,23,0,0" Name="cmbCompatibility" VerticalAlignment="Top" Width="162" />

<Label Content="Stock(UsedNumeric):" HorizontalAlignment="Left" Margin="341,90,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtStockUsed" HorizontalAlignment="Left" Height="23" Margin="471,86,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="179"/>

</Grid>

</Page>

**Games.vb**

'Class used for to get all the data from the inputs of the administrator or employee when adding a game and put it to database

Public Class Games

Private cName As String

Private cCategory As String

Private cReleaseDate As Date

Private cProducer As String

Private cRating As Double

Private cDescription As String

Private cStock As Integer

Private cStockUsed As Integer

Private cPrice As Double

Private cCost As Double

Private cImage() As Byte

Private cCompatibility As Integer

Public Sub New(ByVal name As String, ByVal category As String, ByVal releaseDate As Date, ByVal producer As String, ByVal rating As Double,

ByVal description As String, ByVal compatibility As Integer, ByVal stock As Integer, ByVal stockUsed As Integer, ByVal price As Double, ByVal cost As Double, ByVal img() As Byte)

cCompatibility = compatibility

cName = name

cCategory = category

cReleaseDate = releaseDate

cProducer = producer

cRating = rating

cDescription = description

cStock = stock

cPrice = price

cCost = cost

cImage = img

cStockUsed = stockUsed

End Sub

Public Function getName() As String

Return cName

End Function

Public Function getCategory() As String

Return cCategory

End Function

Public Function getReleaseDate() As Date

Return cReleaseDate

End Function

Public Function getProducer() As String

Return cProducer

End Function

Public Function getRating() As Double

Return cRating

End Function

Public Function getDescription() As String

Return cDescription

End Function

Public Function getStock() As String

Return cStock

End Function

Public Function getPrice() As String

Return cPrice

End Function

Public Function getCost() As String

Return cCost

End Function

Public Function getImage() As Byte()

Return cImage

End Function

Public Function getCompatibility()

Return cCompatibility

End Function

Public Function getStockUsed()

Return cStockUsed

End Function

End Class

**addGame.xaml.vb**

Imports System.IO

Imports System.Data

Class addGame

Private db As DBTransaction

Private dTableCom As DataTable

'Button used to add the data in the fields in the database table of the movies

Private Sub btnAddGame\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnAddGame.Click

Dim name As String = txtName.Text

Dim category As String = txtCategory.Text

Dim releaseDate As Date = dtpReleaseDate.Text

Dim producer As String = txtProducer.Text

Dim rating As String = txtRating.Text

Dim txtRange As New TextRange(txtDescription.Document.ContentStart, txtDescription.Document.ContentEnd)

Dim description As String = txtRange.Text

Dim stock As String = txtStock.Text

Dim stockUsed As String = txtStockUsed.Text

Dim price As String = txtPrice.Text

Dim cost As String = txtCost.Text

Dim compatibility As Integer = cmbCompatibility.SelectedValue

Dim img() As Byte = Nothing

If txtImage.Text.Length <> 0 Then

Dim fs As FileStream = New FileStream(txtImage.Text, FileMode.Open, FileAccess.Read)

Dim br As BinaryReader = New BinaryReader(fs)

img = br.ReadBytes(fs.Length())

End If

If name.Length = 0 Or category.Length = 0 Or producer.Length = 0 Or rating.Length = 0 Or

description.Length = 0 Or stock.Length = 0 Or price.Length = 0 Or cost.Length = 0 Then

lblError.Content = "Please fill in all the fields!"

Else

Dim newGame As Games = New Games(name, category, releaseDate, producer, CType(rating, Double),

description, compatibility, CType(stock, Integer), CType(stockUsed, Integer), CType(price, Double), CType(cost, Double), img)

Dim db As DBTransaction = New DBTransaction

If db.addGame(newGame) = 1 Then

lblError.Content = ""

lblSuccess.Content = "Insert is successful"

resetFields()

Else

lblError.Content = "Something happened. Please try again later."

End If

End If

End Sub

'Button used to browse a image from the computer and filling the field with its path of location

Private Sub btnBrowse\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnBrowse.Click

Dim dlg As Microsoft.Win32.OpenFileDialog = New Microsoft.Win32.OpenFileDialog

dlg.DefaultExt = ".jpg"

dlg.Filter = "Image File (\*.png, \*.jpg)|\*.png; \*.jpg"

Dim result As Boolean = dlg.ShowDialog

If result = True Then

txtImage.Text = dlg.FileName

End If

End Sub

'Sub the clears all the fields

Private Sub resetFields()

txtName.Text = ""

txtCategory.Text = ""

txtCost.Text = ""

txtPrice.Text = ""

txtStock.Text = ""

txtProducer.Text = ""

txtImage.Text = ""

txtDescription.Document.Blocks.Clear()

txtRating.Text = ""

End Sub

'Sub that loads the drop-down list with all the compatibilites that exist in the database

Private Sub Grid1\_Loaded(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles Grid1.Loaded

db = New DBTransaction()

dTableCom = New DataTable()

dTableCom = db.getCompatibilities()

cmbCompatibility.ItemsSource = dTableCom.DefaultView

cmbCompatibility.DisplayMemberPath = "compatibility\_name"

cmbCompatibility.SelectedValuePath = "compatibility\_id"

cmbCompatibility.SelectedIndex = 0

dtpReleaseDate.SelectedDate = DateTime.Today

End Sub

End Class

4. Adding an employee

**addEmployee.xaml**

<Page x:Class="addEmployee"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="400" d:DesignWidth="700"

Title="addEmployee">

<Grid Name="Grid1">

<Label Content="First name:" Height="24" HorizontalAlignment="Left" Margin="23,26,0,0" Name="Label1" VerticalAlignment="Top" Width="74" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="123,26,0,0" Name="txtFname" VerticalAlignment="Top" Width="200" />

<Label Content="Last name:" Height="28" HorizontalAlignment="Left" Margin="23,76,0,0" Name="Label2" VerticalAlignment="Top" />

<Label Content="Gender:" Height="28" HorizontalAlignment="Left" Margin="23,126,0,0" Name="Label3" VerticalAlignment="Top" />

<Label Content="Age(Numeric):" Height="28" HorizontalAlignment="Left" Margin="23,176,0,0" Name="Label4" VerticalAlignment="Top" />

<Label Content="Date of employment:" Height="28" HorizontalAlignment="Left" Margin="358,26,0,0" Name="Label5" VerticalAlignment="Top" />

<Label Content="Salary(Double):" Height="28" HorizontalAlignment="Left" Margin="358,76,0,0" Name="Label6" VerticalAlignment="Top" />

<Label Content="Username:" Height="28" HorizontalAlignment="Left" Margin="358,126,0,0" Name="Label7" VerticalAlignment="Top" />

<Label Content="Password:" Height="28" HorizontalAlignment="Left" Margin="358,176,0,0" Name="Label8" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="123,76,0,0" Name="txtLname" VerticalAlignment="Top" Width="200" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="123,176,0,0" Name="txtAge" VerticalAlignment="Top" Width="200" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="488,76,0,0" Name="txtSalary" VerticalAlignment="Top" Width="200" />

<DatePicker Height="25" HorizontalAlignment="Left" Margin="488,26,0,0" Name="dtpEmployment" VerticalAlignment="Top" Width="200" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="488,126,0,0" Name="txtUsername" VerticalAlignment="Top" Width="200" />

<RadioButton Content="Male" Height="18" HorizontalAlignment="Left" Margin="123,129,0,0" Name="rdbMale" VerticalAlignment="Top" IsChecked="True" />

<RadioButton Content="Female" Height="18" HorizontalAlignment="Left" Margin="237,129,0,0" Name="rdbFemale" VerticalAlignment="Top" />

<Button Content="Add Employee" Height="31" HorizontalAlignment="Left" Margin="303,245,0,0" Name="btnAddEmployee" VerticalAlignment="Top" Width="182" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="488,176,0,0" Name="txtPass" VerticalAlignment="Top" Width="200" />

<Label Height="28" HorizontalAlignment="Left" Margin="488,360,0,0" Name="lblError" VerticalAlignment="Top" Width="200" Foreground="#FFFA0000" />

<Label Height="28" HorizontalAlignment="Left" Margin="488,360,0,0" Name="lblSuccess" VerticalAlignment="Top" Width="200" Foreground="Lime" />

</Grid>

</Page>

**Employees.vb**

'Class used for to get all the data from the inputs of the administrator when adding an employee and put it to database

Public Class Employees

Private cFname As String

Private cLname As String

Private cGender As Char

Private cAge As String

Private cDateofEmployment As Date

Private cSalary As Double

Private cUsername As String

Private cPass As String

Public Sub New(ByVal fname As String, ByVal lname As String, ByVal gender As Char, ByVal age As String,

ByVal dateofEmployment As Date, ByVal salary As Double, ByVal username As String, ByVal pass As String)

cFname = fname

cLname = lname

cGender = gender

cAge = age

cDateofEmployment = dateofEmployment

cSalary = salary

cUsername = username

cPass = pass

End Sub

Public Function getFname() As String

Return cFname

End Function

Public Function getLname() As String

Return cLname

End Function

Public Function getGender() As Char

Return cGender

End Function

Public Function getAge() As String

Return cAge

End Function

Public Function getDateofEmployment() As String

Return cDateofEmployment

End Function

Public Function getSalary() As Double

Return cSalary

End Function

Public Function getUsername() As String

Return cUsername

End Function

Public Function getPass() As String

Return cPass

End Function

End Class

**addEmployee.xaml.vb**

Imports System.IO

Class addEmployee

'Button used to add the text in the fields in the database table

Private Sub btnAddEmployee\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnAddEmployee.Click

Dim fname As String = txtFname.Text

Dim lname As String = txtLname.Text

Dim uname As String = txtUsername.Text

Dim pass As String = txtUsername.Text

Dim salary As String = txtSalary.Text

Dim age As String = txtAge.Text

Dim gender As Char

If rdbMale.IsChecked Then

gender = "M"

Else

gender = "F"

End If

Dim dateOf As Date = dtpEmployment.Text

If fname.Length = 0 Or lname.Length = 0 Or uname.Length = 0 Or pass.Length = 0 Or salary.Length = 0 Or age.Length = 0 Then

lblError.Content = "Please fill in all the fields!"

lblSuccess.Content = ""

Else

Dim newEmployee As Employees = New Employees(fname, lname, gender, age, dateOf, CType(salary, Double), uname, pass)

Dim db As DBTransaction = New DBTransaction

If db.addEmployee(newEmployee) = 1 Then

lblError.Content = ""

lblSuccess.Content = "Insert is successful"

resetFields()

Else

lblSuccess.Content = ""

lblError.Content = "Something happened. Please try again later."

End If

End If

End Sub

'Sub to clear all the fields

Private Sub resetFields()

txtFname.Text = ""

txtLname.Text = ""

txtAge.Text = ""

txtSalary.Text = ""

txtUsername.Text = ""

txtPass.Text = ""

End Sub

'Sub that sets the present date in the date picker

Private Sub Grid1\_Loaded(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles Grid1.Loaded

dtpEmployment.SelectedDate = DateTime.Today

End Sub

End Class

5. Adding a member

**addMember.xaml**

<Page x:Class="addMember"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="400" d:DesignWidth="700"

Title="addMember">

<Grid Name="Grid1">

<Label Content="First name:" Height="24" HorizontalAlignment="Left" Margin="23,26,0,0" Name="Label1" VerticalAlignment="Top" Width="74" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="123,26,0,0" Name="txtFname" VerticalAlignment="Top" Width="200" />

<Label Content="Last name:" Height="28" HorizontalAlignment="Left" Margin="23,76,0,0" Name="Label2" VerticalAlignment="Top" />

<Label Content="Gender:" Height="28" HorizontalAlignment="Left" Margin="23,126,0,0" Name="Label3" VerticalAlignment="Top" />

<Label Content="Age(Numeric):" Height="28" HorizontalAlignment="Left" Margin="23,176,0,0" Name="Label4" VerticalAlignment="Top" />

<Label Content="Registration date:" Height="28" HorizontalAlignment="Left" Margin="358,26,0,0" Name="Label5" VerticalAlignment="Top" />

<Label Content="E-mail:" Height="28" HorizontalAlignment="Left" Margin="358,76,0,0" Name="Label6" VerticalAlignment="Top" />

<Label Content="Phone number:" Height="28" HorizontalAlignment="Left" Margin="358,126,0,0" Name="Label7" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="123,76,0,0" Name="txtLname" VerticalAlignment="Top" Width="200" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="123,176,0,0" Name="txtAge" VerticalAlignment="Top" Width="200" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="488,76,0,0" Name="txtEmail" VerticalAlignment="Top" Width="200" />

<DatePicker Height="25" HorizontalAlignment="Left" Margin="488,26,0,0" Name="dtpRegistration" VerticalAlignment="Top" Width="200" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="488,126,0,0" Name="txtPhone" VerticalAlignment="Top" Width="200" />

<RadioButton Content="Male" Height="16" HorizontalAlignment="Left" Margin="123,129,0,0" Name="rdbMale" VerticalAlignment="Top" IsChecked="True" />

<RadioButton Content="Female" Height="16" HorizontalAlignment="Left" Margin="237,129,0,0" Name="rdbFemale" VerticalAlignment="Top" />

<Button Content="Add Member" Height="31" HorizontalAlignment="Left" Margin="488,173,0,0" Name="btnAddMember" VerticalAlignment="Top" Width="200" />

<Label Height="28" HorizontalAlignment="Left" Margin="488,360,0,0" Name="lblError" VerticalAlignment="Top" Width="200" Foreground="#FFFA0000" />

<Label Height="28" HorizontalAlignment="Left" Margin="488,360,0,0" Name="lblSuccess" VerticalAlignment="Top" Width="200" Foreground="Lime" />

</Grid>

</Page>

**Member.vb**

'Class used for to get all the data from the inputs of the administrator or employee when adding a member and put it to database

Public Class Member

Private cFname As String

Private cLname As String

Private cGender As String

Private cAge As String

Private cRegistrationDate As Date

Private cEmail As String

Private cPhone As String

Private cDiscount

Public Sub New(ByVal fname As String, ByVal lname As String, ByVal gender As String, ByVal age As String,

ByVal registrationDate As Date, ByVal email As String, ByVal phone As String, ByVal dicount As Double)

cFname = fname

cLname = lname

cGender = gender

cAge = age

cRegistrationDate = registrationDate

cEmail = email

cPhone = phone

cDiscount = dicount

End Sub

Public Function getFname() As String

Return cFname

End Function

Public Function getLname() As String

Return cLname

End Function

Public Function getGender() As String

Return cGender

End Function

Public Function getAge() As String

Return cAge

End Function

Public Function getRegistrationDate() As String

Return cRegistrationDate

End Function

Public Function getEmail() As String

Return cEmail

End Function

Public Function getPhone() As String

Return cPhone

End Function

Public Function getDiscount() As Double

Return cDiscount

End Function

End Class

**addMember.xaml.vb**

Imports System.IO

Class addMember

'Button used to add all the data in the fields in the member's table in the dattabase

Private Sub btnAddMember\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnAddMember.Click

Dim fname As String = txtFname.Text

Dim lname As String = txtLname.Text

Dim email As String = txtEmail.Text

Dim phone As String = txtPhone.Text

Dim age As String = txtAge.Text

Dim gender As String

If rdbMale.IsChecked Then

gender = "M"

Else

gender = "F"

End If

Dim dateOf As Date = dtpRegistration.Text

If fname.Length = 0 Or lname.Length = 0 Or email.Length = 0 Or phone.Length = 0 Or age.Length = 0 Then

lblError.Content = "Please fill in all the fields!"

lblSuccess.Content = ""

Else

Dim newMember As Member = New Member(fname, lname, gender, age, dateOf, email, phone, 0)

Dim db As DBTransaction = New DBTransaction

If db.addMember(newMember) = 1 Then

lblError.Content = ""

lblSuccess.Content = "Insert is successful"

resetFields()

Else

lblSuccess.Content = ""

lblError.Content = "Something happened. Please try again later."

End If

End If

End Sub

'Sub the clears all the fields

Private Sub resetFields()

txtFname.Text = ""

txtLname.Text = ""

txtAge.Text = ""

txtEmail.Text = ""

txtPhone.Text = ""

End Sub

'Sub the sets the present date in the date picker

Private Sub Grid1\_Loaded(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles Grid1.Loaded

dtpRegistration.SelectedDate = DateTime.Today

End Sub

End Class

**c) Listing/modifying/deleting an item**

1. Listing/modifying/deleting a movie

**listMovies.xaml**

<Page x:Class="listMovies"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="565" d:DesignWidth="1050"

Title="listMovies">

<Grid Name="Grid1" Height="524">

<DataGrid AutoGenerateColumns="False" x:Name="dgMovies" HorizontalAlignment="Left" Margin="10,10,0,0" VerticalAlignment="Top" Height="271" Width="965" IsReadOnly="True" >

<DataGrid.Columns>

<DataGridTextColumn Header="ID" Binding="{Binding Path=film\_id}"></DataGridTextColumn>

<DataGridTextColumn Header="Title" Binding="{Binding Path=film\_title}"></DataGridTextColumn>

<DataGridTextColumn Header="Category" Binding="{Binding Path=film\_categories}"></DataGridTextColumn>

<DataGridTextColumn Header="Release Date" Binding="{Binding Path=film\_release\_date, StringFormat=d}"></DataGridTextColumn>

<DataGridTextColumn Header="Runtime" Binding="{Binding Path=film\_runtime}"></DataGridTextColumn>

<DataGridTextColumn Header="Rating" Binding="{Binding Path=film\_rating}"></DataGridTextColumn>

<DataGridTextColumn Header="Description" Binding="{Binding Path=film\_description}"></DataGridTextColumn>

<DataGridTextColumn Header="Stock" Binding="{Binding Path=film\_stock}"></DataGridTextColumn>

<DataGridTextColumn Header="Price" Binding="{Binding Path=film\_price}"></DataGridTextColumn>

<DataGridTextColumn Header="Cost" Binding="{Binding Path=film\_cost}"></DataGridTextColumn>

<DataGridTemplateColumn Header="Image">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Image Width="50" Height="50" Source="{Binding Path=film\_image}">

</Image>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

<GroupBox Header="Modify" Height="229" HorizontalAlignment="Left" Margin="4,310,0,-15" Name="GroupBox1" VerticalAlignment="Top" Width="774"></GroupBox>

<Grid Height="212" Margin="10,327,284,-15">

<Label Content="Name:" Height="28" HorizontalAlignment="Left" Margin="9,36,0,0" Name="Label1" VerticalAlignment="Top" />

<Label Content="Categories:" Height="28" HorizontalAlignment="Left" Margin="6,69,0,0" Name="Label2" VerticalAlignment="Top" />

<Label Content="Release Date:" Height="28" HorizontalAlignment="Left" Margin="6,102,0,0" Name="Label3" VerticalAlignment="Top" />

<Label Content="Runtime:" Height="28" HorizontalAlignment="Left" Margin="6,136,0,0" Name="Label4" VerticalAlignment="Top" />

<Label Content="Rating:" Height="28" HorizontalAlignment="Left" Margin="6,171,0,0" Name="Label5" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="92,39,0,0" Name="txtName" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="92,69,0,0" Name="txtCategories" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="92,136,0,0" Name="txtRuntime" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="92,171,0,0" Name="txtRating" VerticalAlignment="Top" Width="150" />

<DatePicker Height="25" HorizontalAlignment="Left" Margin="92,102,0,0" Name="dtpRelease" VerticalAlignment="Top" Width="150" />

<Label Content="Description:" Height="28" HorizontalAlignment="Left" Margin="263,53,0,0" Name="Label6" VerticalAlignment="Top" />

<Label Content="Stock:" Height="28" HorizontalAlignment="Left" Margin="263,136,0,0" Name="Label7" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="344,136,0,0" Name="txtStock" VerticalAlignment="Top" Width="150" />

<Label Content="Price:" Height="25" HorizontalAlignment="Left" Margin="263,171,0,0" Name="Label8" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="344,171,0,0" Name="txtPrice" VerticalAlignment="Top" Width="150" />

<Button Content="Modify" Height="28" HorizontalAlignment="Left" Margin="539,78,0,0" Name="btnModify" VerticalAlignment="Top" Width="150" />

<Label Content="ID:" Height="27" HorizontalAlignment="Left" Margin="9,11,0,0" Name="Label9" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="92,11,0,0" Name="txtID" VerticalAlignment="Top" Width="150" IsReadOnly="True" />

<Label Height="28" HorizontalAlignment="Left" Margin="539,145,0,0" Name="lblError" VerticalAlignment="Top" Width="150" Foreground="Red" />

<Label Content="" Height="28" HorizontalAlignment="Left" Margin="539,145,0,0" Name="lblSuccess" VerticalAlignment="Top" Width="150" Foreground="Lime" />

<Button Content="Delete" Height="28" HorizontalAlignment="Left" Margin="539,112,0,0" Name="btnDelete" VerticalAlignment="Top" Width="150" />

<Label Content="Image:" HorizontalAlignment="Left" Margin="539,12,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtImage" HorizontalAlignment="Left" Height="23" Margin="539,48,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

<Button x:Name="btnBrowse" Content="Browse" HorizontalAlignment="Left" Margin="607,12,0,0" VerticalAlignment="Top" Width="75"/>

</Grid>

<RichTextBox HorizontalAlignment="Left" Height="100" Margin="357,350,0,0" VerticalAlignment="Top" Width="144">

<FlowDocument>

<Paragraph>

<Run Name="txtDescripton" Text=""/>

</Paragraph>

</FlowDocument>

</RichTextBox>

<Label Content="Filter:" FontSize="16" HorizontalAlignment="Left" Margin="10,286,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtFilter" HorizontalAlignment="Left" Height="23" Margin="66,287,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="175" TextChanged="TextBox\_TextChanged\_1"/>

</Grid>

</Page>

**listMovies.xaml.vb**

Imports System.Data

Imports System.IO

Class listMovies

Private db As DBTransaction = New DBTransaction

Private moviesTable As DataTable

'Sub that fills the datagrid with the movie's data from the datatable when the grid is loaded

Private Sub Grid1\_Loaded(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles Grid1.Loaded

moviesTable = db.getAllMovies()

dgMovies.ItemsSource = moviesTable.DefaultView

End Sub

'Sub used do fill the fields with the selected movie from the data grid

Private Sub dgMovies\_MouseDoubleClick(sender As System.Object, e As System.Windows.Input.MouseButtonEventArgs) Handles dgMovies.MouseDoubleClick

Dim dataRow As DataRowView = dgMovies.SelectedItem

txtID.Text = dataRow.Item(0).ToString()

txtName.Text = dataRow.Item(1).ToString()

txtCategories.Text = dataRow.Item(2).ToString()

dtpRelease.Text = dataRow.Item(3).ToString()

txtRuntime.Text = dataRow.Item(4).ToString()

txtRating.Text = dataRow.Item(5).ToString()

txtDescripton.Text = dataRow.Item(6).ToString()

txtStock.Text = dataRow.Item(7).ToString()

txtPrice.Text = dataRow.Item(8).ToString()

lblError.Content = ""

lblSuccess.Content = ""

End Sub

'Button used to modify the selected movie with the data in the fields

Private Sub btnModify\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnModify.Click

If txtID.Text.Length <> 0 Then

Dim id As Integer = CType(txtID.Text, Integer)

Dim name As String = txtName.Text

Dim category As String = txtCategories.Text

Dim releaseDate As Date = dtpRelease.Text

Dim runtime As String = txtRuntime.Text

Dim rating As String = txtRating.Text

Dim description As String = txtDescripton.Text

Dim stock As String = txtStock.Text

Dim price As String = txtPrice.Text

Dim img() As Byte = Nothing

If name.Length = 0 Or category.Length = 0 Or runtime.Length = 0 Or rating.Length = 0 Or

description.Length = 0 Or stock.Length = 0 Or price.Length = 0 Or txtImage.Text.Length = 0 Then

lblError.Content = "Please fill in all the fields!"

lblSuccess.Content = ""

Else

Dim fs As FileStream = New FileStream(txtImage.Text, FileMode.Open, FileAccess.Read)

Dim br As BinaryReader = New BinaryReader(fs)

img = br.ReadBytes(fs.Length())

Dim rs As Integer = db.editMovie(id, name, category, releaseDate, runtime, CType(rating, Double), description,

CType(stock, Double), CType(price, Double), img)

If rs = 1 Then

lblSuccess.Content = "Movie has been modified!"

lblError.Content = ""

dgMovies.ItemsSource = db.getAllMovies().DefaultView

clearFields()

Else

lblError.Content = "Something wrong has happened."

lblSuccess.Content = ""

End If

End If

Else

lblError.Content = "Please select a movie."

lblSuccess.Content = ""

End If

End Sub

'Button used to delete the selected movie from the database

Private Sub btnDelete\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnDelete.Click

If txtID.Text.Length <> 0 Then

Dim id As Integer = CType(txtID.Text, Integer)

Dim result As Integer = MessageBox.Show("Do you really want do delete this movie?", "Delete movie", MessageBoxButton.YesNo)

If result = vbYes Then

Dim rs As Integer = db.deleteMovie(id)

If rs = 1 Then

lblSuccess.Content = "Movie has been deleted."

lblError.Content = ""

dgMovies.ItemsSource = db.getAllMovies().DefaultView

Else

lblError.Content = "Something wrong has happened."

lblSuccess.Content = ""

End If

End If

Else

lblError.Content = "Please select a movie."

lblSuccess.Content = ""

End If

End Sub

'Sub used to clear fields

Private Sub clearFields()

txtID.Text = ""

txtCategories.Text = ""

txtDescripton.Text = ""

txtName.Text = ""

txtPrice.Text = ""

txtRating.Text = ""

txtRuntime.Text = ""

txtStock.Text = ""

txtImage.Text = ""

End Sub

'Button used to browse and image for the movie

Private Sub btnBrowse\_Click(sender As Object, e As RoutedEventArgs) Handles btnBrowse.Click

Dim dlg As Microsoft.Win32.OpenFileDialog = New Microsoft.Win32.OpenFileDialog

dlg.DefaultExt = ".jpg"

dlg.Filter = "Image File (\*.png, \*.jpg)|\*.png; \*.jpg"

Dim result As Boolean = dlg.ShowDialog

If result = True Then

txtImage.Text = dlg.FileName

End If

End Sub

'Sub to handle the event of changing text in a text field and filters the data table rows

Private Sub TextBox\_TextChanged\_1(sender As Object, e As TextChangedEventArgs)

moviesTable.DefaultView.RowFilter = "film\_title LIKE '\*" & txtFilter.Text.ToString() & "\*'"

End Sub

End Class

2. Listing/modifying/deleting a game

**listGames.xaml**

<Page x:Class="listGames"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="565" d:DesignWidth="1050"

Title="listGames">

<Grid Name="Grid1">

<DataGrid x:Name="dgGame" AutoGenerateColumns="False" IsReadOnly="True" HorizontalAlignment="Left" Margin="10,10,0,0" VerticalAlignment="Top" Height="271" Width="965">

<DataGrid.Columns>

<DataGridTextColumn Header="ID" Binding="{Binding Path=game\_id}"></DataGridTextColumn>

<DataGridTextColumn Header="Title" Binding="{Binding Path=game\_title}"></DataGridTextColumn>

<DataGridTextColumn Header="Categories" Binding="{Binding Path=game\_categories}"></DataGridTextColumn>

<DataGridTextColumn Header="Release Date" Binding="{Binding Path=game\_release\_date, StringFormat=d}"></DataGridTextColumn>

<DataGridTextColumn Header="Producer" Binding="{Binding Path=game\_producer}"></DataGridTextColumn>

<DataGridTextColumn Header="Rating" Binding="{Binding Path=game\_rating}"></DataGridTextColumn>

<DataGridTextColumn Header="Decription" Binding="{Binding Path=game\_description}"></DataGridTextColumn>

<DataGridTextColumn Header="Stock New" Binding="{Binding Path=game\_stock\_new}"></DataGridTextColumn>

<DataGridTextColumn Header="Stock Used" Binding="{Binding Path=game\_stock\_used}"></DataGridTextColumn>

<DataGridTextColumn Header="Price" Binding="{Binding Path=game\_price}"></DataGridTextColumn>

<DataGridTextColumn Header="Compatibility" Binding="{Binding Path=compatibility\_name}"></DataGridTextColumn>

<DataGridTemplateColumn Header="Image">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Image Width="50" Height="50" Source="{Binding Path=game\_image}">

</Image>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

<GroupBox Header="Modify" Height="252" HorizontalAlignment="Left" Margin="4,313,0,0" Name="GroupBox1" VerticalAlignment="Top" Width="1036"></GroupBox>

<Grid Margin="10,333,10,-13">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="508\*" />

<ColumnDefinition Width="7\*" />

</Grid.ColumnDefinitions>

<Label Content="Name:" Height="28" HorizontalAlignment="Left" Margin="9,41,0,0" Name="Label1" VerticalAlignment="Top" Width="47" />

<Label Content="Categories:" Height="28" HorizontalAlignment="Left" Margin="6,72,0,0" Name="Label2" VerticalAlignment="Top" Width="71" />

<Label Content="Release Date:" Height="28" HorizontalAlignment="Left" Margin="6,102,0,0" Name="Label3" VerticalAlignment="Top" Width="84" />

<Label Content="Producer:" Height="28" HorizontalAlignment="Left" Margin="6,130,0,0" Name="Label4" VerticalAlignment="Top" Width="71" />

<Label Content="Rating:" Height="28" HorizontalAlignment="Left" Margin="6,158,0,0" Name="Label5" VerticalAlignment="Top" Width="50" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="95,41,0,0" Name="txtName" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="95,69,0,0" Name="txtCategories" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="95,130,0,0" Name="txtProducer" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="95,163,0,0" Name="txtRating" VerticalAlignment="Top" Width="150" />

<DatePicker Height="25" HorizontalAlignment="Left" Margin="95,99,0,0" Name="dtpRelease" VerticalAlignment="Top" Width="150" />

<Label Content="Description:" Height="28" HorizontalAlignment="Left" Margin="263,84,0,0" Name="Label6" VerticalAlignment="Top" Width="82" />

<Label Content="Stock (New):" Height="28" HorizontalAlignment="Left" Margin="534,15,0,0" Name="Label7" VerticalAlignment="Top" Width="82" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="617,15,0,0" Name="txtStock" VerticalAlignment="Top" Width="150" />

<Label Content="Price:" Height="23" HorizontalAlignment="Left" Margin="534,89,0,0" Name="Label8" VerticalAlignment="Top" Width="43" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="617,87,0,0" Name="txtPrice" VerticalAlignment="Top" Width="150" />

<Button Content="Modify" Height="28" HorizontalAlignment="Left" Margin="826,15,0,0" Name="btnModify" VerticalAlignment="Top" Width="152" />

<Label Content="ID:" Height="27" HorizontalAlignment="Left" Margin="9,11,0,0" Name="Label9" VerticalAlignment="Top" Width="34" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="95,11,0,0" Name="txtID" VerticalAlignment="Top" Width="150" IsReadOnly="True" />

<Label Content="Compatibility:" Height="28" HorizontalAlignment="Left" Margin="263,11,0,0" Name="Label10" VerticalAlignment="Top" Width="86" />

<ComboBox Height="23" HorizontalAlignment="Left" Margin="349,11,0,0" Name="cmbCompatibility" VerticalAlignment="Top" Width="150" />

<Label Height="28" HorizontalAlignment="Left" Margin="783,96,0,0" Name="lblError" VerticalAlignment="Top" Width="233" Foreground="Red" />

<Label Content="" Height="28" HorizontalAlignment="Left" Margin="783,96,0,0" Name="lblSuccess" VerticalAlignment="Top" Width="233" Foreground="Lime" />

<RichTextBox HorizontalAlignment="Left" Height="133" Margin="350,53,0,0" VerticalAlignment="Top" Width="150">

<FlowDocument>

<Paragraph>

<Run Name="txtDescription" Text=""/>

</Paragraph>

</FlowDocument>

</RichTextBox>

<Label Content="Stock (Used):" HorizontalAlignment="Left" Margin="534,53,0,0" VerticalAlignment="Top" Height="31" Width="78"/>

<TextBox x:Name="txtStockUsed" HorizontalAlignment="Left" Height="23" Margin="617,53,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

<Label Content="Image:" HorizontalAlignment="Left" Margin="534,126,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtImage" IsReadOnly="True" HorizontalAlignment="Left" Height="23" Margin="617,123,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

<Button x:Name="btnBrowse" Content="Browse" HorizontalAlignment="Left" Margin="617,158,0,0" VerticalAlignment="Top" Height="28" Width="150"/>

<Button Content="Delete" Height="28" HorizontalAlignment="Left" Margin="826,53,0,0" Name="btnDelete" VerticalAlignment="Top" Width="152" />

</Grid>

<Label Content="Filter:" FontSize="16" HorizontalAlignment="Left" Margin="10,288,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtFilter" HorizontalAlignment="Left" Height="23" Margin="63,290,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="181"/>

</Grid>

</Page>

**listGames.xaml.vb**

Imports System.Data

Imports System.IO

Class listGames

Private db As DBTransaction = New DBTransaction

Private gamesTable As DataTable

Private compTable As DataTable

'Sub that fills the datagrid with the game's data from the datatable when the grid is loaded

Private Sub Grid1\_Loaded(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles Grid1.Loaded

db = New DBTransaction()

gamesTable = New DataTable()

compTable = New DataTable()

gamesTable = db.getAllGames()

dgGame.ItemsSource = gamesTable.DefaultView

compTable = db.getCompatibilities()

cmbCompatibility.ItemsSource = compTable.DefaultView

cmbCompatibility.DisplayMemberPath = "compatibility\_name"

cmbCompatibility.SelectedValuePath = "compatibility\_id"

End Sub

'Sub used do fill the fields with the selected game from the data grid

Private Sub dgGame\_MouseDoubleClick(sender As System.Object, e As System.Windows.Input.MouseButtonEventArgs) Handles dgGame.MouseDoubleClick

Dim dataRow As DataRowView = dgGame.SelectedItem

txtID.Text = dataRow.Item("game\_id").ToString()

txtName.Text = dataRow.Item("game\_title").ToString()

txtCategories.Text = dataRow.Item("game\_categories").ToString()

dtpRelease.Text = dataRow.Item("game\_release\_date").ToString()

txtProducer.Text = dataRow.Item("game\_producer").ToString()

txtRating.Text = dataRow.Item("game\_rating").ToString()

txtDescription.Text = dataRow.Item("game\_description").ToString()

txtStock.Text = dataRow.Item("game\_stock\_new").ToString()

txtStockUsed.Text = dataRow.Item("game\_stock\_used").ToString

txtPrice.Text = dataRow.Item("game\_price").ToString()

cmbCompatibility.SelectedIndex = CType(dataRow.Item("compatibility\_id"), Integer) - 1

lblSuccess.Content = ""

lblError.Content = ""

End Sub

'Button used to modify the selected game with the data in the fields

Private Sub btnModify\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnModify.Click

If txtID.Text.Length <> 0 Then

Dim id As Integer = CType(txtID.Text, Integer)

Dim name As String = txtName.Text

Dim category As String = txtCategories.Text

Dim releaseDate As Date = dtpRelease.Text

Dim producer As String = txtProducer.Text

Dim rating As String = txtRating.Text

Dim description As String = txtDescription.Text

Dim stock As String = txtStock.Text

Dim stockUsed As String = txtStockUsed.Text

Dim price As String = txtPrice.Text

Dim compatibility As Integer = cmbCompatibility.SelectedValue

Dim img() As Byte = Nothing

If name.Length = 0 Or category.Length = 0 Or producer.Length = 0 Or rating.Length = 0 Or

description.Length = 0 Or stock.Length = 0 Or price.Length = 0 Or txtImage.Text.Length = 0 Then

lblError.Content = "Please fill in all the fields!"

lblSuccess.Content = ""

Else

Dim fs As FileStream = New FileStream(txtImage.Text, FileMode.Open, FileAccess.Read)

Dim br As BinaryReader = New BinaryReader(fs)

img = br.ReadBytes(fs.Length())

Dim rs As Integer = db.editGame(id, name, category, releaseDate, producer, CType(rating, Double), description,

CType(stock, Double), CType(stockUsed, Integer), CType(price, Double), compatibility, img)

If rs = 1 Then

lblSuccess.Content = "Game has been modified successfuly!"

lblError.Content = ""

dgGame.ItemsSource = db.getAllGames().DefaultView

clearFields()

Else

lblError.Content = "Something wrong has happened. Please try again later!"

lblSuccess.Content = ""

End If

End If

Else

lblSuccess.Content = ""

lblError.Content = "Please select a game from the list."

End If

End Sub

'Sub used to clear the fields

Private Sub clearFields()

txtCategories.Text = ""

txtDescription.Text = ""

txtName.Text = ""

txtID.Text = ""

txtPrice.Text = ""

txtProducer.Text = ""

txtRating.Text = ""

txtStock.Text = ""

txtStockUsed.Text = ""

cmbCompatibility.SelectedIndex = 0

txtImage.Text = ""

End Sub

'Button used to browse an image for the game

Private Sub btnBrowse\_Click(sender As Object, e As RoutedEventArgs) Handles btnBrowse.Click

Dim dlg As Microsoft.Win32.OpenFileDialog = New Microsoft.Win32.OpenFileDialog

dlg.DefaultExt = ".jpg"

dlg.Filter = "Image File (\*.png, \*.jpg)|\*.png; \*.jpg"

Dim result As Boolean = dlg.ShowDialog

If result = True Then

txtImage.Text = dlg.FileName

End If

End Sub

'Button used to delete the selected game from the database

Private Sub btnDelete\_Click(sender As Object, e As RoutedEventArgs) Handles btnDelete.Click

If txtID.Text.Length <> 0 Then

Dim id As Integer = CType(txtID.Text, Integer)

Dim result As Integer = MessageBox.Show("Do you really want do delete this game?", "Delete game", MessageBoxButton.YesNo)

If result = vbYes Then

Dim rs As Integer = db.deleteGame(id)

If rs = 1 Then

lblSuccess.Content = "Game has beed deleted successfully."

lblError.Content = ""

dgGame.ItemsSource = db.getAllGames().DefaultView

clearFields()

Else

lblError.Content = "Something wrong has happened. Please try again later."

lblSuccess.Content = ""

End If

End If

Else

lblError.Content = "Please select a game from the list."

lblSuccess.Content = ""

End If

End Sub

'Sub to handle the event of changing text in a text field and filters the data table rows

Private Sub txtFilter\_TextChanged(sender As Object, e As TextChangedEventArgs) Handles txtFilter.TextChanged

gamesTable.DefaultView.RowFilter = "game\_title LIKE '\*" & txtFilter.Text.ToString & "\*'"

End Sub

End Class

3. Listing/modifying/deleting an employee

**listEmployee.xaml**

<Page x:Class="listEmployee"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="565" d:DesignWidth="1050"

Title="listEmployee">

<Grid Name="Grid1">

<DataGrid AutoGenerateColumns="False" x:Name="dgEmployees" HorizontalAlignment="Left" Margin="10,10,0,0" VerticalAlignment="Top" Height="271" Width="920" IsReadOnly="True">

<DataGrid.Columns>

<DataGridTextColumn Width="70" Header="ID" Binding="{Binding Path=employee\_id}"></DataGridTextColumn>

<DataGridTextColumn Width="110" Header="First Name" Binding="{Binding Path=employee\_first\_name}"></DataGridTextColumn>

<DataGridTextColumn Width="110" Header="Last Name" Binding="{Binding Path=employee\_last\_name}"></DataGridTextColumn>

<DataGridTextColumn Width="110" Header="Gender" Binding="{Binding Path=employee\_gender}"></DataGridTextColumn>

<DataGridTextColumn Width="100" Header="Age" Binding="{Binding Path=employee\_age}"></DataGridTextColumn>

<DataGridTextColumn Width="150" Header="Date" Binding="{Binding Path=employee\_date\_employment, StringFormat=d}"></DataGridTextColumn>

<DataGridTextColumn Width="110" Header="Salary" Binding="{Binding Path=employee\_salary}"></DataGridTextColumn>

<DataGridTextColumn Width="110" Header="Username" Binding="{Binding Path=employee\_username}"></DataGridTextColumn>

<DataGridTextColumn Width="110" Header="Password" Binding="{Binding Path=employee\_password}"></DataGridTextColumn>

</DataGrid.Columns>

</DataGrid>

<GroupBox Header="Modify" Height="246" HorizontalAlignment="Left" Margin="4,283,0,0" Name="GroupBox1" VerticalAlignment="Top" Width="545"></GroupBox>

<Grid Margin="10,305,419,24">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="454\*" />

<ColumnDefinition Width="167\*" />

</Grid.ColumnDefinitions>

<Label Content="First name:" Height="28" HorizontalAlignment="Left" Margin="9,46,0,0" Name="Label1" VerticalAlignment="Top" />

<Label Content="Last name:" Height="28" HorizontalAlignment="Left" Margin="10,80,0,0" Name="Label2" VerticalAlignment="Top" />

<Label Content="Employment date:" Height="28" HorizontalAlignment="Left" Margin="10,115,0,0" Name="Label3" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="124,43,0,0" Name="txtFname" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="124,77,0,0" Name="txtLname" VerticalAlignment="Top" Width="150" />

<DatePicker Height="25" HorizontalAlignment="Left" Margin="124,111,0,0" Name="dtpEmployment" VerticalAlignment="Top" Width="150" />

<Label Content="Salary:" Height="28" HorizontalAlignment="Left" Margin="298,11,0,0" Name="Label6" VerticalAlignment="Top" />

<Label Content="Username:" Height="28" HorizontalAlignment="Left" Margin="298,46,0,0" Name="Label7" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="378,11,0,0" Name="txtSalary" VerticalAlignment="Top" Width="150" Grid.ColumnSpan="2" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="378,43,0,0" Name="txtUsername" VerticalAlignment="Top" Width="150" Grid.ColumnSpan="2" />

<Label Content="Password:" Height="21" HorizontalAlignment="Left" Margin="298,80,0,0" Name="Label8" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="378,77,0,0" Name="txtPassword" VerticalAlignment="Top" Width="150" Grid.ColumnSpan="2" />

<Button Content="Modify" Height="28" HorizontalAlignment="Left" Margin="378,109,0,0" Name="btnModify" VerticalAlignment="Top" Width="150" Grid.ColumnSpan="2" />

<Label Content="ID:" Height="27" HorizontalAlignment="Left" Margin="10,11,0,0" Name="Label9" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="124,11,0,0" Name="txtID" VerticalAlignment="Top" Width="150" IsReadOnly="True" />

<Label Content="Age:" Height="28" HorizontalAlignment="Left" Margin="10,148,0,0" Name="Label10" VerticalAlignment="Top" />

<Label Height="28" HorizontalAlignment="Left" Margin="280,178,0,0" Name="lblError" VerticalAlignment="Top" Width="248" Foreground="Red" Grid.ColumnSpan="2" />

<Label Content="" Height="28" HorizontalAlignment="Left" Margin="280,178,0,0" Name="lblSuccess" VerticalAlignment="Top" Width="248" Foreground="Lime" Grid.ColumnSpan="2" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="124,148,0,0" Name="txtAge" VerticalAlignment="Top" Width="150" />

<Label Content="Gender:" Height="28" HorizontalAlignment="Left" Margin="9,180,0,0" Name="Label4" VerticalAlignment="Top" />

<RadioButton Content="Male" Height="18" HorizontalAlignment="Left" Margin="124,180,0,0" Name="rdbMale" VerticalAlignment="Top" Width="71" />

<RadioButton Content="Female" Height="18" HorizontalAlignment="Left" Margin="208,180,0,0" Name="rdbFemale" VerticalAlignment="Top" Width="66" />

<Button Content="Delete" Height="28" HorizontalAlignment="Left" Margin="378,146,0,0" Name="btnDelete" VerticalAlignment="Top" Width="150" Grid.ColumnSpan="2" />

</Grid>

</Grid>

</Page>

**listEmployee.xaml.vb**

Imports System.Data

Class listEmployee

Private db As DBTransaction = New DBTransaction

Private employeesTable As DataTable

'Sub that fills the datagrid with the employee's data from the datatable when the grid is loaded

Private Sub Grid1\_Loaded(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles Grid1.Loaded

employeesTable = db.getAllEmployees

dgEmployees.ItemsSource = employeesTable.DefaultView

End Sub

'Button used to modify the selected employee with the data in the fields

Private Sub btnModify\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnModify.Click

If txtID.Text.Length <> 0 Then

Dim id As Integer = CType(txtID.Text, Integer)

Dim fname As String = txtFname.Text

Dim lname As String = txtLname.Text

Dim gender As Char

If rdbMale.IsChecked = True Then

gender = "M"

Else

gender = "F"

End If

Dim employmentDate As Date = dtpEmployment.Text

Dim salary As String = txtSalary.Text

Dim age As String = txtAge.Text

Dim username As String = txtUsername.Text

Dim pass As String = txtPassword.Text

If fname.Length = 0 Or lname.Length = 0 Or username.Length = 0 Or pass.Length = 0 Or salary.Length = 0 Or age.Length = 0 Then

lblError.Content = "Please fill in all the fields!"

lblSuccess.Content = ""

Else

Dim rs As Integer = db.editEmployee(id, fname, lname, gender, age, employmentDate, CType(salary, Double), username, pass)

If rs = 1 Then

lblSuccess.Content = "Employee has been modified!"

lblError.Content = ""

dgEmployees.ItemsSource = db.getAllEmployees().DefaultView

clearFields()

Else

lblError.Content = "Something wrong has happened."

lblSuccess.Content = ""

End If

End If

Else

lblSuccess.Content = ""

lblError.Content = "Please select an employee."

End If

End Sub

'Sub used do fill the fields with the selected employee from the data grid

Private Sub dgEmployees\_MouseDoubleClick(sender As System.Object, e As System.Windows.Input.MouseButtonEventArgs) Handles dgEmployees.MouseDoubleClick

Dim dataRow As DataRowView = dgEmployees.SelectedItem

txtID.Text = dataRow.Item("employee\_id").ToString

txtFname.Text = dataRow.Item("employee\_first\_name").ToString

txtLname.Text = dataRow.Item("employee\_last\_name").ToString

If dataRow.Item("employee\_gender").ToString = "M" Then

rdbMale.IsChecked = True

Else

rdbFemale.IsChecked = True

End If

txtAge.Text = dataRow.Item("employee\_age").ToString

dtpEmployment.Text = dataRow.Item("employee\_date\_employment").ToString

txtSalary.Text = dataRow.Item("employee\_salary").ToString

txtUsername.Text = dataRow.Item(7).ToString

txtPassword.Text = dataRow.Item(8).ToString

lblError.Content = ""

lblSuccess.Content = ""

End Sub

'Sub used to clear the fields

Private Sub clearFields()

txtID.Text = ""

txtAge.Text = ""

txtFname.Text = ""

txtLname.Text = ""

txtPassword.Text = ""

txtSalary.Text = ""

txtUsername.Text = ""

End Sub

'Button used to delete from the database the selected employee from the datagrid

Private Sub btnDelete\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnDelete.Click

If txtID.Text.Length <> 0 Then

Dim id As Integer = CType(txtID.Text, Integer)

Dim result As Integer = MessageBox.Show("Do you really want do delete this employee?", "Delete employee", MessageBoxButton.YesNo)

If result = vbYes Then

Dim rs As Integer = db.deleteEmployee(id)

If rs = 1 Then

lblSuccess.Content = "Employee has beed deleted."

lblError.Content = ""

dgEmployees.ItemsSource = db.getAllEmployees().DefaultView

clearFields()

Else

lblError.Content = "Something wrong has happened."

lblSuccess.Content = ""

End If

End If

Else

lblError.Content = "Please select a employee."

lblSuccess.Content = ""

End If

End Sub

End Class

4. Listing/modifying/deleting a member

**listMember.xaml**

<Page x:Class="listMember"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="565" d:DesignWidth="1050"

Title="listMember">

<Grid Name="Grid1">

<DataGrid AutoGenerateColumns="False" x:Name="dgMembers" HorizontalAlignment="Left" Margin="10,10,0,0" VerticalAlignment="Top" Height="271" Width="1030" IsReadOnly="True">

<DataGrid.Columns>

<DataGridTextColumn Header="ID" Binding="{Binding Path=member\_id}"></DataGridTextColumn>

<DataGridTextColumn Header="First Name" Binding="{Binding Path=member\_first\_name}"></DataGridTextColumn>

<DataGridTextColumn Header="Last Name" Binding="{Binding Path=member\_last\_name}"></DataGridTextColumn>

<DataGridTextColumn Header="Age" Binding="{Binding Path=member\_age}"></DataGridTextColumn>

<DataGridTextColumn Header="Gender" Binding="{Binding Path=member\_gender}"></DataGridTextColumn>

<DataGridTextColumn Header="Registration Date" Binding="{Binding Path=member\_registration\_date, StringFormat=d}"></DataGridTextColumn>

<DataGridTextColumn Header="Films Bought" Binding="{Binding Path=member\_films\_bought}"></DataGridTextColumn>

<DataGridTextColumn Header="Games Bought" Binding="{Binding Path=member\_games\_bought}"></DataGridTextColumn>

<DataGridTextColumn Header="Phone Number" Binding="{Binding Path=member\_phone\_number}"></DataGridTextColumn>

<DataGridTextColumn Header="Email" Binding="{Binding Path=member\_email}"></DataGridTextColumn>

<DataGridTextColumn Header="Discount" Binding="{Binding Path=member\_discount}"></DataGridTextColumn>

</DataGrid.Columns>

</DataGrid>

<GroupBox Header="Modify" Height="253" HorizontalAlignment="Left" Margin="4,283,0,0" Name="GroupBox1" VerticalAlignment="Top" Width="560"></GroupBox>

<Grid Height="220" Margin="10,300,419,12">

<Grid.ColumnDefinitions>

<ColumnDefinition Width="619\*" />

<ColumnDefinition Width="2\*" />

</Grid.ColumnDefinitions>

<Label Content="First name:" Height="28" HorizontalAlignment="Left" Margin="5,51,0,0" Name="Label1" VerticalAlignment="Top" />

<Label Content="Last name:" Height="28" HorizontalAlignment="Left" Margin="6,93,0,0" Name="Label2" VerticalAlignment="Top" />

<Label Content="Registration:" Height="28" HorizontalAlignment="Left" Margin="279,11,0,0" Name="Label3" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="101,51,0,0" Name="txtFname" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="101,93,0,0" Name="txtLname" VerticalAlignment="Top" Width="150" />

<DatePicker Height="25" HorizontalAlignment="Left" Margin="378,11,0,0" Name="dtpRegistration" VerticalAlignment="Top" Width="150" />

<Label Content="E-mail:" Height="28" HorizontalAlignment="Left" Margin="279,54,0,0" Name="Label6" VerticalAlignment="Top" />

<Label Content="Phone:" Height="28" HorizontalAlignment="Left" Margin="279,96,0,0" Name="Label7" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="378,51,0,0" Name="txtEmail" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="378,93,0,0" Name="txtPhone" VerticalAlignment="Top" Width="150" />

<Button Content="Modify" Height="28" HorizontalAlignment="Left" Margin="279,133,0,0" Name="btnModify" VerticalAlignment="Top" Width="107" />

<Label Content="ID:" Height="27" HorizontalAlignment="Left" Margin="10,11,0,0" Name="Label9" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="101,11,0,0" Name="txtID" VerticalAlignment="Top" Width="150" IsReadOnly="True" />

<Label Content="Age:" Height="28" HorizontalAlignment="Left" Margin="5,135,0,0" Name="Label10" VerticalAlignment="Top" />

<Label Height="28" HorizontalAlignment="Left" Margin="265,178,0,0" Name="lblError" VerticalAlignment="Top" Width="263" Foreground="Red" />

<Label Content="" Height="28" HorizontalAlignment="Left" Margin="265,178,0,0" Name="lblSuccess" VerticalAlignment="Top" Width="263" Foreground="Lime" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="101,135,0,0" Name="txtAge" VerticalAlignment="Top" Width="150" />

<Label Content="Gender:" Height="28" HorizontalAlignment="Left" Margin="6,178,0,0" Name="Label4" VerticalAlignment="Top" />

<RadioButton Content="Male" Height="16" HorizontalAlignment="Left" Margin="101,178,0,0" Name="rdbMale" VerticalAlignment="Top" Width="71" />

<RadioButton Content="Female" Height="16" HorizontalAlignment="Left" Margin="185,178,0,0" Name="rdbFemale" VerticalAlignment="Top" Width="66" />

<Button Content="Delete" Height="28" HorizontalAlignment="Left" Margin="415,133,0,0" Name="btnDelete" VerticalAlignment="Top" Width="113" />

</Grid>

</Grid>

</Page>

**listMember.xaml.vb**

Imports System.Data

Class listMember

Private db As DBTransaction = New DBTransaction

Private membersTable As DataTable

''Sub that fills the datagrid with the member's data from the datatable when the grid is loaded

Private Sub Grid1\_Loaded(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles Grid1.Loaded

membersTable = db.getAllMembers

dgMembers.ItemsSource = membersTable.DefaultView

End Sub

'Button used to modify the selected member in the database with the new data in fields

Private Sub btnModify\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnModify.Click

If txtID.Text.Length <> 0 Then

Dim id As Integer = CType(txtID.Text, Integer)

Dim fname As String = txtFname.Text

Dim lname As String = txtLname.Text

Dim gender As Char

If rdbMale.IsChecked = True Then

gender = "M"

Else

gender = "F"

End If

Dim registrationDate As Date = dtpRegistration.Text

Dim age As String = txtAge.Text

Dim email As String = txtEmail.Text

Dim phone As String = txtPhone.Text

If fname.Length = 0 Or lname.Length = 0 Or email.Length = 0 Or phone.Length = 0 Or age.Length = 0 Then

lblError.Content = "Please fill in all the fields!"

lblSuccess.Content = ""

Else

Dim rs As Integer = db.editMember(id, fname, lname, gender, age, registrationDate, email, phone)

If rs = 1 Then

lblSuccess.Content = "Member has been modified successfuly!"

lblError.Content = ""

dgMembers.ItemsSource = db.getAllMembers().DefaultView

clearFields()

Else

lblError.Content = "Something wrong has happened. Please try again later!"

lblSuccess.Content = ""

End If

End If

Else

lblSuccess.Content = ""

lblError.Content = "Please select a member from the list."

End If

End Sub

'Sub used do fill the fields with the selected member from the data grid

Private Sub dgMembers\_MouseDoubleClick(sender As System.Object, e As System.Windows.Input.MouseButtonEventArgs) Handles dgMembers.MouseDoubleClick

Dim dataRow As DataRowView = dgMembers.SelectedItem

txtID.Text = dataRow.Item("member\_id").ToString

txtFname.Text = dataRow.Item("member\_first\_name").ToString

txtLname.Text = dataRow.Item("member\_last\_name").ToString

If dataRow.Item("member\_gender").ToString = "M" Then

rdbMale.IsChecked = True

Else

rdbFemale.IsChecked = True

End If

txtAge.Text = dataRow.Item("member\_age").ToString

dtpRegistration.Text = dataRow.Item("member\_registration\_date").ToString

txtEmail.Text = dataRow.Item("member\_email").ToString

txtPhone.Text = dataRow.Item("member\_phone\_number").ToString

End Sub

'Button used to delete the member from the database

Private Sub btnDelete\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnDelete.Click

If txtID.Text.Length <> 0 Then

Dim id As Integer = CType(txtID.Text, Integer)

Dim result As Integer = MessageBox.Show("Do you really want do delete this member?", "Delete member", MessageBoxButton.YesNo)

If result = vbYes Then

Dim rs As Integer = db.deleteMember(id)

If rs = 1 Then

lblSuccess.Content = "Member has beed deleted successfully."

lblError.Content = ""

dgMembers.ItemsSource = db.getAllMembers().DefaultView

Else

lblError.Content = "Something wrong has happened. Please try again later."

lblSuccess.Content = ""

End If

End If

Else

lblError.Content = "Please select a member from the list."

lblSuccess.Content = ""

End If

End Sub

'Sub used to clear all the fields

Sub clearFields()

txtAge.Text = ""

txtEmail.Text = ""

txtFname.Text = ""

txtID.Text = ""

txtLname.Text = ""

txtPhone.Text = ""

End Sub

End Class

**d) Selling an item**

1. Selling a movie

**sellMovie.xaml**

<Page x:Class="sellMovie"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="480" d:DesignWidth="750"

Title="sellMovie" Name="lblError">

<Grid Name="Grid1">

<DataGrid AutoGenerateColumns="False" Height="200" HorizontalAlignment="Left" Margin="12,40,0,0" Name="dgMovies" VerticalAlignment="Top" Width="316" IsReadOnly="True">

<DataGrid.Columns >

<DataGridTextColumn Header="ID" Binding="{Binding Path=film\_id}"> </DataGridTextColumn>

<DataGridTextColumn Header="Title" Binding="{Binding Path=film\_title}"></DataGridTextColumn>

<DataGridTextColumn Header="Stock" Binding="{Binding Path=film\_stock}"></DataGridTextColumn>

<DataGridTextColumn Header="Price" Binding="{Binding Path=film\_price}"></DataGridTextColumn>

<DataGridTextColumn Header="Cost" Binding="{Binding Path=film\_cost}"></DataGridTextColumn>

</DataGrid.Columns>

</DataGrid>

<DataGrid AutoGenerateColumns="False" Height="200" HorizontalAlignment="Left" Margin="363,39,0,0" Name="dgMembers" VerticalAlignment="Top" Width="325" IsReadOnly="True">

<DataGrid.Columns >

<DataGridTextColumn Header="ID" Binding="{Binding Path=member\_id}"></DataGridTextColumn>

<DataGridTextColumn Header="First name" Binding="{Binding Path=member\_first\_name}"></DataGridTextColumn>

<DataGridTextColumn Header="Last name" Binding="{Binding Path=member\_last\_name}"></DataGridTextColumn>

<DataGridTextColumn Header="Movies bought" Binding="{Binding Path=member\_films\_bought}"></DataGridTextColumn>

<DataGridTextColumn Header="Discount" Binding="{Binding Path=member\_discount}"></DataGridTextColumn>

<DataGridTextColumn Header="Registration date" Binding="{Binding Path=member\_registration\_date, StringFormat=d}"></DataGridTextColumn>

</DataGrid.Columns>

</DataGrid>

<Label Content="Filter:" Height="19" HorizontalAlignment="Left" Margin="12,249,0,0" Name="Label1" VerticalAlignment="Top" Width="37" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="55,245,0,0" Name="txtFilmFilter" VerticalAlignment="Top" Width="273" />

<Label Content="Filter:" Height="21" HorizontalAlignment="Left" Margin="363,247,0,0" Name="Label2" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="406,245,0,0" Name="txtMemberFilter" VerticalAlignment="Top" Width="282" />

<GroupBox Header="Sell Movies" Height="186" HorizontalAlignment="Left" Margin="12,279,0,0" Name="GroupBox1" VerticalAlignment="Top" Width="715">

<Grid>

<Label Content="Film ID:" Height="28" HorizontalAlignment="Left" Margin="6,18,0,0" Name="Label3" VerticalAlignment="Top" />

<Label Content="Stock:" Height="28" HorizontalAlignment="Left" Margin="6,52,0,0" Name="Label4" VerticalAlignment="Top" />

<Label Content="Price:" Height="28" HorizontalAlignment="Left" Margin="6,86,0,0" Name="Label5" VerticalAlignment="Top" />

<Label Content="Cost:" Height="28" HorizontalAlignment="Left" Margin="6,120,0,0" Name="Label6" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="66,18,0,0" Name="txtFilmId" VerticalAlignment="Top" Width="163" IsReadOnly="True" />

<TextBox Height="23" HorizontalAlignment="Left" IsReadOnly="True" Margin="66,52,0,0" Name="txtStock" VerticalAlignment="Top" Width="163" />

<TextBox Height="23" HorizontalAlignment="Left" IsReadOnly="True" Margin="66,86,0,0" Name="txtPrice" VerticalAlignment="Top" Width="163" />

<TextBox Height="23" HorizontalAlignment="Left" IsReadOnly="True" Margin="66,120,0,0" Name="txtCost" VerticalAlignment="Top" Width="163" />

<Label Content="Member ID:" Height="28" HorizontalAlignment="Left" Margin="248,18,0,0" Name="Label7" VerticalAlignment="Top" />

<Label Content="Movies bought: " Height="28" HorizontalAlignment="Left" Margin="248,52,0,0" Name="Label8" VerticalAlignment="Top" />

<Label Content="Registration Date:" Height="28" HorizontalAlignment="Left" Margin="248,86,0,0" Name="Label9" VerticalAlignment="Top" />

<Label Content="Discount:" Height="28" HorizontalAlignment="Left" Margin="248,120,0,0" Name="Label10" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" IsReadOnly="True" Margin="360,18,0,0" Name="txtMemberId" VerticalAlignment="Top" Width="178" />

<TextBox Height="23" HorizontalAlignment="Left" IsReadOnly="True" Margin="360,52,0,0" Name="txtMoviesBought" VerticalAlignment="Top" Width="178" />

<TextBox Height="23" HorizontalAlignment="Left" IsReadOnly="True" Margin="360,83,0,0" Name="txtRegistrationDate" VerticalAlignment="Top" Width="178" />

<TextBox Height="23" HorizontalAlignment="Left" IsReadOnly="True" Margin="360,120,0,0" Name="txtDiscount" VerticalAlignment="Top" Width="178" />

<Button Content="Clear Fields" Height="28" HorizontalAlignment="Left" Margin="556,18,0,0" Name="Button1" VerticalAlignment="Top" Width="137" />

<Button Content="Sell Movie" HorizontalAlignment="Left" Margin="556,96,0,40" Name="btnSellMovie" Width="137" Height="28" />

<Button Content="Check Discount" Height="28" HorizontalAlignment="Left" Margin="556,57,0,0" Name="btnCheck" VerticalAlignment="Top" Width="137" />

<Label Height="28" HorizontalAlignment="Left" Margin="556,129,0,0" Name="lblSuccess" VerticalAlignment="Top" Width="123" Foreground="Lime" />

<Label Content="" Height="28" HorizontalAlignment="Left" Margin="556,129,0,0" Name="lblError2" VerticalAlignment="Top" Width="123" Foreground="Red" />

</Grid>

</GroupBox>

<Label Content="Film Stock" HorizontalAlignment="Left" Margin="12,10,0,0" VerticalAlignment="Top" FontSize="18"/>

<Label Content="Member List" HorizontalAlignment="Left" Margin="364,10,0,0" VerticalAlignment="Top" FontSize="18"/>

</Grid>

</Page>

**sellMovie.xaml.vb**

Imports System.Data

Class sellMovie

Private db As DBTransaction

Private dtableMovies As DataTable

Private dtableMembers As DataTable

'Button that clears all the fields

Private Sub Button1\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles Button1.Click

clearFields()

End Sub

'Sub that clears all the fields

Private Sub clearFields()

txtFilmId.Text = ""

txtCost.Text = ""

txtDiscount.Text = ""

txtFilmFilter.Text = ""

txtMemberId.Text = ""

txtMoviesBought.Text = ""

txtPrice.Text = ""

txtRegistrationDate.Text = ""

txtStock.Text = ""

End Sub

'Sub that fills two data grids when the grid is loaded. One with the movies in the database and the other with the members in the database

Private Sub Grid1\_Loaded(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles Grid1.Loaded

db = New DBTransaction()

dtableMovies = New DataTable()

dtableMovies = db.getAllMovies()

dgMovies.ItemsSource = dtableMovies.DefaultView

dtableMembers = New DataTable()

dtableMembers = db.getAllMembers()

dgMembers.ItemsSource = dtableMembers.DefaultView

lblError2.Content = ""

lblSuccess.Content = ""

End Sub

'Sub that filters the list depending on what you write in the textbox

Private Sub txtFilmFilter\_TextChanged(sender As System.Object, e As System.Windows.Controls.TextChangedEventArgs) Handles txtFilmFilter.TextChanged

dtableMovies.DefaultView.RowFilter = "film\_title LIKE '\*" & txtFilmFilter.Text & "\*'"

End Sub

'Sub that filters the list depending on what you write in the textbox

Private Sub txtMemberFilter\_TextChanged(sender As System.Object, e As System.Windows.Controls.TextChangedEventArgs) Handles txtMemberFilter.TextChanged

dtableMembers.DefaultView.RowFilter = "member\_first\_name LIKE '\*" & txtMemberFilter.Text & "\*'"

End Sub

'Sub that fills the fields for the movies with the selected movie

Private Sub dgMovies\_MouseDoubleClick(sender As System.Object, e As System.Windows.Input.MouseButtonEventArgs) Handles dgMovies.MouseDoubleClick

Dim dataRow As DataRowView = dgMovies.SelectedItem

txtFilmId.Text = dataRow.Item("film\_id").ToString()

txtStock.Text = dataRow.Item("film\_stock").ToString()

txtPrice.Text = dataRow.Item("film\_price").ToString()

txtCost.Text = dataRow.Item("film\_cost").ToString()

lblError2.Content = ""

lblSuccess.Content = ""

If txtMemberId.Text.Length <> 0 Then

Dim moviesBought As Integer = CType(txtMoviesBought.Text, Integer)

If moviesBought Mod 10 = 0 And moviesBought > 0 Then

txtPrice.Text = "0"

ElseIf CType(txtDiscount.Text.ToString, Integer) <> 0 And txtPrice.Text.Length <> 0 Then

Dim pr As Double = CType(txtPrice.Text.ToString, Double)

pr = pr \* 0.9

txtPrice.Text = pr.ToString

Else

txtPrice.Text = dataRow.Item("film\_price")

End If

End If

End Sub

'Sub that fills the fields for the members with the selected member

Private Sub dgMembers\_MouseDoubleClick(sender As System.Object, e As System.Windows.Input.MouseButtonEventArgs) Handles dgMembers.MouseDoubleClick

Dim dataRow As DataRowView = dgMembers.SelectedItem

txtMemberId.Text = dataRow.Item("member\_id").ToString()

txtMoviesBought.Text = dataRow.Item("member\_films\_bought").ToString()

txtRegistrationDate.Text = dataRow.Item("member\_registration\_date").ToString()

txtDiscount.Text = dataRow.Item("member\_discount")

lblSuccess.Content = ""

lblError2.Content = ""

Dim moviesBought As Integer = CType(txtMoviesBought.Text, Integer)

If moviesBought Mod 10 = 0 And moviesBought > 0 Then

txtPrice.Text = "0"

ElseIf CType(txtDiscount.Text.ToString, Integer) <> 0 And txtPrice.Text.Length <> 0 Then

Dim pr As Double = getRealPrice()

pr = pr \* 0.9

txtPrice.Text = pr.ToString

Else

txtPrice.Text = getRealPrice().ToString

End If

End Sub

'Button that checks for discount

Private Sub btnCheck\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnCheck.Click

If txtMoviesBought.Text.Length <> 0 Then

Dim moviesBought As Integer = CType(txtMoviesBought.Text, Integer)

If moviesBought Mod 10 = 0 And moviesBought > 0 Then

lblSuccess.Content = "Success"

lblError2.Content = ""

txtPrice.Text = "0"

Else

lblError2.Content = "No discount"

lblSuccess.Content = ""

End If

Else

lblError2.Content = "Select a member"

lblSuccess.Content = ""

End If

End Sub

'Button used to sell a movie

Private Sub btnSellMovie\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnSellMovie.Click

If txtFilmId.Text.Length <> 0 And CType(txtStock.Text.ToString, Integer) <> 0 Then

Dim filmId As Integer = CType(txtFilmId.Text, Integer)

Dim stock As Integer = CType(txtStock.Text, Integer)

Dim price As Double = CType(txtPrice.Text, Integer)

Dim cost As Double = CType(txtCost.Text, Integer)

Dim memberId As Integer = 0

If txtMemberId.Text.Length <> 0 Then

memberId = CType(txtMemberId.Text, Integer)

End If

If db.sellMovie(MainWindow.getId(), filmId, Date.Now, price, cost, memberId) = 3 And db.decrementFilmStock(filmId) = 1 Then

lblSuccess.Content = "Sold"

lblError2.Content = ""

clearFields()

dgMovies.ItemsSource = db.getAllMovies.DefaultView

dgMembers.ItemsSource = db.getAllMembers.DefaultView

Else

lblSuccess.Content = ""

lblError2.Content = "Wrong"

End If

Else

lblError2.Content = "No stock"

lblSuccess.Content = ""

End If

End Sub

'Gets the movie real price

Private Function getRealPrice() As Double

Dim drt As DataRowView = dgMovies.SelectedItem

Dim price As Double = drt.Item("film\_price")

Return price

End Function

End Class

2. Selling a game

**sellGame.xaml**

<Page x:Class="sellGame"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="480" d:DesignWidth="750"

Title="sellGame">

<Grid Loaded="Grid\_Loaded\_1">

<DataGrid IsReadOnly="True" AutoGenerateColumns="False" x:Name="dgGames" HorizontalAlignment="Left" Margin="10,39,0,0" VerticalAlignment="Top" Width="350" Height="200">

<DataGrid.Columns>

<DataGridTextColumn Header="ID" Binding="{Binding Path=game\_id}"></DataGridTextColumn>

<DataGridTextColumn Header="Title" Binding="{Binding Path=game\_title}"></DataGridTextColumn>

<DataGridTextColumn Header="Price" Binding="{Binding Path=game\_price}"></DataGridTextColumn>

<DataGridTextColumn Header="Cost" Binding="{Binding Path=game\_cost}"></DataGridTextColumn>

<DataGridTextColumn Header="Stock New" Binding="{Binding Path=game\_stock\_new}"></DataGridTextColumn>

<DataGridTextColumn Header="Stock Used" Binding="{Binding Path=game\_stock\_used}"></DataGridTextColumn>

</DataGrid.Columns>

</DataGrid>

<Label Content="Games Stock" HorizontalAlignment="Left" Margin="10,10,0,0" VerticalAlignment="Top" FontSize="18"/>

<DataGrid IsReadOnly="True" AutoGenerateColumns="False" x:Name="dgMembers" HorizontalAlignment="Left" Margin="390,39,0,0" VerticalAlignment="Top" Width="350" Height="200">

<DataGrid.Columns >

<DataGridTextColumn Header="ID" Binding="{Binding Path=member\_id}"></DataGridTextColumn>

<DataGridTextColumn Header="First name" Binding="{Binding Path=member\_first\_name}"></DataGridTextColumn>

<DataGridTextColumn Header="Last name" Binding="{Binding Path=member\_last\_name}"></DataGridTextColumn>

<DataGridTextColumn Header="Games bought" Binding="{Binding Path=member\_games\_bought}"></DataGridTextColumn>

<DataGridTextColumn Header="Discount" Binding="{Binding Path=member\_discount}"></DataGridTextColumn>

<DataGridTextColumn Header="Registration Date" Binding="{Binding Path=member\_registration\_date, StringFormat=d}"></DataGridTextColumn>

</DataGrid.Columns>

</DataGrid>

<Label Content="Member List" HorizontalAlignment="Left" Margin="390,10,0,0" VerticalAlignment="Top" FontSize="18"/>

<Label Content="Filter:" HorizontalAlignment="Left" Margin="10,249,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtGameFilter" HorizontalAlignment="Left" Height="23" Margin="44,244,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="316"/>

<Label Content="Filter:" HorizontalAlignment="Left" Margin="390,249,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtMemberFilter" HorizontalAlignment="Left" Height="23" Margin="440,246,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="300"/>

<GroupBox Header="Sell Games" HorizontalAlignment="Left" Margin="10,278,0,0" VerticalAlignment="Top" Height="192" Width="730">

<Grid>

<Label Content="Game ID:" Height="28" HorizontalAlignment="Left" Margin="6,18,0,0" Name="Label3" VerticalAlignment="Top" />

<Label Content="Stock(New):" Height="28" HorizontalAlignment="Left" Margin="6,52,0,0" Name="Label4" VerticalAlignment="Top" />

<Label Content="Price:" Height="28" HorizontalAlignment="Left" Margin="6,112,0,0" Name="Label5" VerticalAlignment="Top" />

<Label Content="Cost:" Height="28" HorizontalAlignment="Left" Margin="6,140,0,0" Name="Label6" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="79,18,0,0" x:Name="txtGameID" VerticalAlignment="Top" Width="163" IsReadOnly="True" />

<TextBox Height="23" HorizontalAlignment="Left" IsReadOnly="True" Margin="79,52,0,0" x:Name="txtStockN" VerticalAlignment="Top" Width="163" />

<TextBox Height="23" HorizontalAlignment="Left" IsReadOnly="True" Margin="79,112,0,0" Name="txtPrice" VerticalAlignment="Top" Width="163" />

<TextBox Height="23" HorizontalAlignment="Left" IsReadOnly="True" Margin="79,142,0,0" Name="txtCost" VerticalAlignment="Top" Width="163" />

<Label Content="Member ID:" Height="28" HorizontalAlignment="Left" Margin="261,18,0,0" Name="Label7" VerticalAlignment="Top" />

<Label Content="Games bought: " Height="28" HorizontalAlignment="Left" Margin="261,52,0,0" Name="Label8" VerticalAlignment="Top" />

<Label Content="Registration Date:" Height="28" HorizontalAlignment="Left" Margin="261,86,0,0" Name="Label9" VerticalAlignment="Top" />

<Label Content="Discount:" Height="28" HorizontalAlignment="Left" Margin="261,120,0,0" Name="Label10" VerticalAlignment="Top" />

<TextBox IsReadOnly="True" Height="23" HorizontalAlignment="Left" Margin="373,18,0,0" Name="txtMemberId" VerticalAlignment="Top" Width="178" />

<TextBox IsReadOnly="True" Height="23" HorizontalAlignment="Left" Margin="373,52,0,0" x:Name="txtGamesBought" VerticalAlignment="Top" Width="178" />

<TextBox IsReadOnly="True" Height="23" HorizontalAlignment="Left" Margin="373,83,0,0" Name="txtRegistrationDate" VerticalAlignment="Top" Width="178" />

<TextBox IsReadOnly="True" Height="23" HorizontalAlignment="Left" Margin="373,120,0,0" Name="txtDiscount" VerticalAlignment="Top" Width="178" />

<Button Content="Clear Fields" Height="28" HorizontalAlignment="Left" Margin="569,18,0,0" Name="Button1" VerticalAlignment="Top" Width="129" />

<Button Content="Sell Game" HorizontalAlignment="Left" Margin="569,95,0,47" x:Name="btnSellGame" Width="129" Height="28" />

<Button Content="Check Discount" Height="28" HorizontalAlignment="Left" Margin="569,57,0,0" Name="btnCheck" VerticalAlignment="Top" Width="129" />

<Label Foreground="Green" Height="28" HorizontalAlignment="Left" Margin="569,135,0,0" Name="lblSuccess" VerticalAlignment="Top" Width="129" />

<Label Foreground="Red" Content="" Height="28" HorizontalAlignment="Left" Margin="569,139,0,0" Name="lblError2" VerticalAlignment="Top" Width="129" />

<Label Content="Stock(Used):" Height="27" HorizontalAlignment="Left" Margin="6,85,0,0" x:Name="Label4\_Copy" VerticalAlignment="Top" />

<TextBox IsReadOnly="True" Height="23" HorizontalAlignment="Left" Margin="79,82,0,0" x:Name="txtStockUsed" VerticalAlignment="Top" Width="163" />

<RadioButton IsEnabled="False" x:Name="rbtNew" Content="New" HorizontalAlignment="Left" Margin="373,147,0,0" VerticalAlignment="Top"/>

<RadioButton IsEnabled="False" x:Name="rbtUsed" Content="Used" HorizontalAlignment="Left" Margin="431,148,0,0" VerticalAlignment="Top"/>

</Grid>

</GroupBox>

</Grid>

</Page>

**sellGame.xaml.vb**

Imports System.Data

Class sellGame

Private db As DBTransaction

Private dtableGames As DataTable

Private dtableMembers As DataTable

Private dr1 As DataRowView

'Sub that fills two data grids when the grid is loaded. One with the games in the database and the other with the members in the database

Private Sub Grid\_Loaded\_1(sender As Object, e As RoutedEventArgs)

db = New DBTransaction()

dtableGames = New DataTable()

dr1 = Nothing

dtableGames = db.getAllGames()

dgGames.ItemsSource = dtableGames.DefaultView

dtableMembers = New DataTable()

dtableMembers = db.getAllMembers()

dgMembers.ItemsSource = dtableMembers.DefaultView

lblError2.Content = ""

lblSuccess.Content = ""

End Sub

'Sub that filters the list depending on what you write in the textbox

Private Sub txtGameFilter\_TextChanged(sender As Object, e As TextChangedEventArgs) Handles txtGameFilter.TextChanged

dtableGames.DefaultView.RowFilter = "game\_title LIKE '\*" & txtGameFilter.Text & "\*'"

End Sub

'Sub that filters the list depending on what you write in the textbox

Private Sub txtMemberFilter\_TextChanged(sender As Object, e As TextChangedEventArgs) Handles txtMemberFilter.TextChanged

dtableMembers.DefaultView.RowFilter = "member\_first\_name LIKE '\*" & txtMemberFilter.Text & "\*'"

End Sub

'Sub that fills the fields for the games with the selected game

Private Sub dgGames\_MouseDoubleClick(sender As Object, e As MouseButtonEventArgs) Handles dgGames.MouseDoubleClick

dr1 = dgGames.SelectedItem

txtGameID.Text = dr1.Item("game\_id")

txtStockN.Text = dr1.Item("game\_stock\_new")

txtStockUsed.Text = dr1.Item("game\_stock\_used")

txtPrice.Text = dr1.Item("game\_price")

txtCost.Text = dr1.Item("game\_cost")

lblError2.Content = ""

lblSuccess.Content = ""

rbtNew.IsEnabled = True

rbtUsed.IsEnabled = True

If txtMemberId.Text.Length <> 0 Then

Dim gamesBought As Integer = CType(txtGamesBought.Text, Integer)

If gamesBought Mod 10 = 0 And gamesBought > 0 Then

txtPrice.Text = "0"

ElseIf CType(txtDiscount.Text.ToString, Integer) <> 0 And txtPrice.Text.Length <> 0 Then

Dim pr As Double = CType(txtPrice.Text.ToString, Double)

pr = pr \* 0.9

txtPrice.Text = pr.ToString

Else

txtPrice.Text = dr1.Item("game\_price")

End If

End If

End Sub

'Sub that fills the fields for the members with the selected member

Private Sub dgMembers\_MouseDoubleClick(sender As Object, e As MouseButtonEventArgs) Handles dgMembers.MouseDoubleClick

Dim dr As DataRowView = dgMembers.SelectedItem

txtMemberId.Text = dr.Item("member\_id")

txtRegistrationDate.Text = dr.Item("member\_registration\_date")

txtGamesBought.Text = dr.Item("member\_games\_bought")

txtDiscount.Text = dr.Item("member\_discount")

If txtMemberId.Text.Length <> 0 Then

Dim gamesBought As Integer = CType(txtGamesBought.Text, Integer)

If gamesBought Mod 10 = 0 And gamesBought > 0 Then

txtPrice.Text = "0"

ElseIf CType(txtDiscount.Text.ToString, Integer) <> 0 And txtPrice.Text.Length <> 0 Then

Dim pr As Double = getRealPrice()

pr = pr \* 0.9

txtPrice.Text = pr.ToString

Else

txtPrice.Text = getRealPrice().ToString

End If

End If

lblError2.Content = ""

lblSuccess.Content = ""

End Sub

'Sub that sets the price to zero if the game is used

Private Sub rbtUsed\_Checked(sender As Object, e As RoutedEventArgs) Handles rbtUsed.Checked

Dim pr As Double = CType(txtPrice.Text.ToString, Double)

Dim cs As Double = CType(txtCost.Text.ToString, Double)

If pr <> 0 Then

pr = pr - pr \* 0.3

txtPrice.Text = pr.ToString

cs = cs - cs \* 0.3

txtCost.Text = cs.ToString

End If

End Sub

'Sub that sets the normal price if the game is new

Private Sub rbtNew\_Checked(sender As Object, e As RoutedEventArgs) Handles rbtNew.Checked

Dim pr As Double = txtPrice.Text.ToString()

txtPrice.Text = pr

End Sub

'Sub that clears all the fields

Private Sub clearFields()

txtCost.Text = ""

txtDiscount.Text = ""

txtGameID.Text = ""

txtGamesBought.Text = ""

txtMemberId.Text = ""

txtPrice.Text = ""

txtRegistrationDate.Text = ""

txtStockN.Text = ""

txtStockUsed.Text = ""

rbtNew.IsEnabled = False

rbtUsed.IsEnabled = False

End Sub

'Button that clears all the fields

Private Sub Button1\_Click(sender As Object, e As RoutedEventArgs) Handles Button1.Click

clearFields()

End Sub

'Button that sells game

Private Sub btnSellGame\_Click(sender As Object, e As RoutedEventArgs) Handles btnSellGame.Click

If txtGameID.Text.Length <> 0 Then

If rbtNew.IsChecked = False And rbtUsed.IsChecked = False Then

lblError2.Content = ""

lblError2.Content = "Select New/Used"

Else

Dim st As Boolean = True

If rbtNew.IsChecked Then

st = True

Else

st = False

End If

If st = True Then

If CType(txtStockN.Text.ToString, Integer) = 0 Then

lblError2.Content = "No Stock"

lblSuccess.Content = ""

Else

Dim mid As Integer = 0

If txtMemberId.Text.Length <> 0 Then

mid = CType(txtMemberId.Text, Integer)

End If

Dim gid As Integer = CType(txtGameID.Text, Integer)

Dim eid As Integer = MainWindow.getId()

Dim price As Double = CType(txtPrice.Text, Double)

Dim cost As Double = CType(txtCost.Text, Double)

Dim rs As Integer = db.sellGame(gid, eid, mid, price, cost, st)

If rs > 0 Then

lblSuccess.Content = "Sold"

lblError2.Content = ""

clearFields()

dgGames.ItemsSource = db.getAllGames().DefaultView

dgMembers.ItemsSource = db.getAllMembers().DefaultView

Else

lblError2.Content = "Error"

lblSuccess.Content = ""

End If

End If

Else

If CType(txtStockUsed.Text.ToString, Integer) = 0 Then

lblError2.Content = "No Stock"

lblSuccess.Content = ""

Else

Dim mid As Integer = 0

If txtMemberId.Text.Length <> 0 Then

mid = CType(txtMemberId.Text, Integer)

End If

Dim gid As Integer = CType(txtGameID.Text, Integer)

Dim eid As Integer = MainWindow.getId()

Dim price As Double = CType(txtPrice.Text, Double)

Dim cost As Double = CType(txtCost.Text, Double)

Dim rs As Integer = db.sellGame(gid, eid, mid, price, cost, st)

If rs > 0 Then

lblSuccess.Content = "Sold"

lblError2.Content = ""

clearFields()

dgGames.ItemsSource = db.getAllGames().DefaultView

dgMembers.ItemsSource = db.getAllMembers().DefaultView

Else

lblError2.Content = "Error"

lblSuccess.Content = ""

End If

End If

End If

End If

End If

End Sub

'Button that checks for discount for a member

Private Sub btnCheck\_Click(sender As Object, e As RoutedEventArgs) Handles btnCheck.Click

If txtGamesBought.Text.Length <> 0 Then

Dim gamesBought As Integer = CType(txtGamesBought.Text, Integer)

If gamesBought Mod 10 = 0 And gamesBought > 0 Then

lblSuccess.Content = "Success"

lblError2.Content = ""

txtPrice.Text = "0"

Else

lblError2.Content = "No discount"

lblSuccess.Content = ""

End If

Else

lblError2.Content = "Select a member"

lblSuccess.Content = ""

End If

End Sub

'Gets the game real price

Private Function getRealPrice() As Double

Dim drt As DataRowView = dgGames.SelectedItem

Dim price As Double = drt.Item("game\_price")

Return price

End Function

End Class

**e) Rental process**

1. Renting a movie

**rentMovie.xaml**

<Page x:Class="returnMovie"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="550" d:DesignWidth="780"

Title="returnMovie">

<Grid Loaded="Grid\_Loaded\_1">

<Label Content="Select Film" HorizontalAlignment="Left" Margin="10,10,0,0" VerticalAlignment="Top" FontSize="18"/>

<DataGrid x:Name="dgMovies" IsReadOnly="True" AutoGenerateColumns="False" HorizontalAlignment="Left" Margin="10,39,0,0" VerticalAlignment="Top" Width="350" Height="250">

<DataGrid.Columns>

<DataGridTextColumn Width="70" Header="ID" Binding="{Binding Path=film\_id}"></DataGridTextColumn>

<DataGridTextColumn Width="140" Header="Title" Binding="{Binding Path=film\_title}"></DataGridTextColumn>

<DataGridTextColumn Width="140" Header="Stock" Binding="{Binding Path=film\_stock}"></DataGridTextColumn>

</DataGrid.Columns>

</DataGrid>

<DataGrid IsReadOnly="True" AutoGenerateColumns="False" x:Name="dgMembers" HorizontalAlignment="Left" Margin="390,39,0,0" VerticalAlignment="Top" Width="350" Height="250">

<DataGrid.Columns>

<DataGridTextColumn Width="70" Header="ID" Binding="{Binding Path=member\_id}"></DataGridTextColumn>

<DataGridTextColumn Width="140" Header="First Name" Binding="{Binding Path=member\_first\_name}"></DataGridTextColumn>

<DataGridTextColumn Width="140" Header="Last Name" Binding="{Binding Path=member\_last\_name}"></DataGridTextColumn>

</DataGrid.Columns>

</DataGrid>

<Label Content="Select Member" HorizontalAlignment="Left" Margin="403,10,0,0" VerticalAlignment="Top" FontSize="18"/>

<Label Content="Filter" HorizontalAlignment="Left" Margin="10,298,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtMovieFilter" HorizontalAlignment="Left" Height="23" Margin="54,294,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="306"/>

<Label Content="Filter" HorizontalAlignment="Left" Margin="390,297,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtMemberFilter" HorizontalAlignment="Left" Height="23" Margin="434,293,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="306"/>

<GroupBox Header="GroupBox" HorizontalAlignment="Left" Margin="10,332,0,0" VerticalAlignment="Top" Width="750" Height="208">

<Grid>

<TextBox x:Name="txtFilmId" IsReadOnly="True" HorizontalAlignment="Left" Height="23" Margin="75,10,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

<Label Content="Film Title" HorizontalAlignment="Left" Margin="10,49,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtFilmTitle" IsReadOnly="True" HorizontalAlignment="Left" Height="23" Margin="75,46,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

<Label Content="Film Stock" HorizontalAlignment="Left" Margin="10,85,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtFilmStock" IsReadOnly="True" HorizontalAlignment="Left" Height="23" Margin="75,82,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

<Label Content="Member ID" HorizontalAlignment="Left" Margin="271,10,0,0" VerticalAlignment="Top"/>

<Label Content="First Name" HorizontalAlignment="Left" Margin="271,46,0,0" VerticalAlignment="Top" Width="60"/>

<Label Content="Rented" HorizontalAlignment="Left" Margin="271,85,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtMemberID" IsReadOnly="True" HorizontalAlignment="Left" Height="23" Margin="357,7,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

<TextBox x:Name="txtFirstName" IsReadOnly="True" HorizontalAlignment="Left" Height="23" Margin="357,46,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

<TextBox x:Name="txtRented" IsReadOnly="True" HorizontalAlignment="Left" Height="23" Margin="357,82,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

<Button x:Name="btnRent" Content="Rent" HorizontalAlignment="Left" Margin="547,43,0,0" VerticalAlignment="Top" Width="181"/>

<Label x:Name="lblError" Foreground="Red" Content="" HorizontalAlignment="Left" Margin="547,82,0,0" VerticalAlignment="Top" Width="181" FontSize="16"/>

<Label x:Name="lblSuccess" Foreground="Green" Content="" HorizontalAlignment="Left" Margin="547,82,0,0" VerticalAlignment="Top" Width="181" FontSize="16"/>

<Label Content="Price" HorizontalAlignment="Left" Margin="547,14,0,0" VerticalAlignment="Top"/>

<TextBox Name="txtPrice" IsReadOnly="True" Text="100 LEK" HorizontalAlignment="Left" Height="23" Margin="591,15,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="137"/>

</Grid>

</GroupBox>

<Label Content="Film ID" HorizontalAlignment="Left" Margin="27,362,0,0" VerticalAlignment="Top"/>

</Grid>

</Page>

**rentMovie.xaml.vb**

Imports System.Data

Class returnMovie

Private db As DBTransaction

Private dTableMembers As DataTable

Private dTableMovies As DataTable

'Sub that fills two tables with data from the database when the grid is loaded. One with the movies and the other with the members

Private Sub Grid\_Loaded\_1(sender As Object, e As RoutedEventArgs)

db = New DBTransaction()

dTableMembers = New DataTable()

dTableMovies = New DataTable()

dTableMovies = db.getAllMovies()

dTableMembers = db.getAllMembers()

dgMovies.ItemsSource = dTableMovies.DefaultView

dgMembers.ItemsSource = dTableMembers.DefaultView

lblError.Content = ""

lblSuccess.Content = ""

End Sub

'Sub that fills the fields for the movie with the selected movie

Private Sub dgMovies\_MouseDoubleClick(sender As Object, e As MouseButtonEventArgs) Handles dgMovies.MouseDoubleClick

Dim dr As DataRowView = dgMovies.SelectedItem

txtFilmId.Text = dr.Item("film\_id")

txtFilmTitle.Text = dr.Item("film\_title")

txtFilmStock.Text = dr.Item("film\_stock")

lblError.Content = ""

lblSuccess.Content = ""

End Sub

'Sub that fills the fields for the members with the selected member

Private Sub dgMembers\_MouseDoubleClick(sender As Object, e As MouseButtonEventArgs) Handles dgMembers.MouseDoubleClick

Dim dr As DataRowView = dgMembers.SelectedItem

txtMemberID.Text = dr.Item("member\_id")

txtFirstName.Text = dr.Item("member\_first\_name")

Dim result As Integer = 0

result = db.getRentedMoviesFromMemberId(CType(txtMemberID.Text.ToString, Integer))

txtRented.Text = result

lblError.Content = ""

lblSuccess.Content = ""

End Sub

'Event handler for text changing in the textbox which handles data table filtering

Private Sub txtMovieFilter\_TextChanged(sender As Object, e As TextChangedEventArgs) Handles txtMovieFilter.TextChanged

dTableMovies.DefaultView.RowFilter = "film\_title LIKE '\*" & txtMovieFilter.Text & "\*'"

End Sub

'Event handler for text changing in the textbox which handles data table filtering

Private Sub txtMemberFilter\_TextChanged(sender As Object, e As TextChangedEventArgs) Handles txtMemberFilter.TextChanged

dTableMembers.DefaultView.RowFilter = "member\_first\_name LIKE '\*" & txtMemberFilter.Text & "\*'"

End Sub

'Button that's used to rent a movie to a member

Private Sub btnRent\_Click(sender As Object, e As RoutedEventArgs) Handles btnRent.Click

If txtMemberID.Text.Length <> 0 And txtFilmId.Text.Length <> 0 Then

If CType(txtRented.Text, Integer) < 2 Then

If CType(txtFilmStock.Text.ToString, Integer) <> 0 Then

Dim fID As Integer = CType(txtFilmId.Text, Integer)

Dim mID As Integer = CType(txtMemberID.Text, Integer)

Dim pr As String = txtPrice.Text.ToString()

pr = pr.Substring(0, pr.IndexOf("L"))

Dim price As Double = CType(pr, Double)

Dim rs As Integer = db.rentMovie(fID, Mid, price)

Dim rs1 As Integer = db.decrementFilmStock(fID)

If rs = 1 And rs1 = 1 Then

dTableMovies = db.getAllMovies()

dgMovies.ItemsSource = dTableMovies.DefaultView

lblSuccess.Content = "OK"

lblError.Content = ""

clearFields()

Else

lblError.Content = "Try again later"

lblSuccess.Content = ""

End If

Else

lblError.Content = "No Stock"

lblSuccess.Content = ""

End If

Else

lblError.Content = "Limit rents"

lblSuccess.Content = ""

End If

Else

lblError.Content = "Film & Member?"

lblSuccess.Content = ""

End If

End Sub

'Sub that's used to clear the fields

Private Sub clearFields()

txtFilmStock.Text = ""

txtFilmTitle.Text = ""

txtFirstName.Text = ""

txtMemberID.Text = ""

txtRented.Text = ""

txtFilmId.Text = ""

End Sub

End Class

2. Rented movies (returning movie, postponing deadline)

**rentedMovies.xaml**

<Page x:Class="rentedMovies"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="550" d:DesignWidth="780"

Title="rentedMovies">

<Grid Loaded="Grid\_Loaded\_1">

<DataGrid x:Name="dgRentedFilms" IsReadOnly="True" AutoGenerateColumns="False" HorizontalAlignment="Left" Margin="10,42,0,0" VerticalAlignment="Top" Width="760" Height="340">

<DataGrid.Columns>

<DataGridTextColumn Header="Member ID" Binding="{Binding Path=member\_id}"></DataGridTextColumn>

<DataGridTextColumn Header="First Name" Binding="{Binding Path=member\_first\_name}"></DataGridTextColumn>

<DataGridTextColumn Header="Last Name" Binding="{Binding Path=member\_last\_name}"></DataGridTextColumn>

<DataGridTextColumn Header="Film ID" Binding="{Binding Path=film\_id}"></DataGridTextColumn>

<DataGridTextColumn Header="Title" Binding="{Binding Path=film\_title}"></DataGridTextColumn>

<DataGridTextColumn Header="Stock" Binding="{Binding Path=film\_stock}"></DataGridTextColumn>

<DataGridTextColumn Header="Rent Date" Binding="{Binding Path=rent\_date, StringFormat=d}"></DataGridTextColumn>

<DataGridTextColumn Header="Return Date" Binding="{Binding Path=return\_date, StringFormat=d}"></DataGridTextColumn>

</DataGrid.Columns>

</DataGrid>

<Label Content="Rented Movies" HorizontalAlignment="Left" Margin="321,10,0,0" VerticalAlignment="Top" FontSize="18"/>

<GroupBox Header="GroupBox" HorizontalAlignment="Left" Margin="10,387,0,0" VerticalAlignment="Top" Height="153" Width="760">

<Grid>

<Label Content="Member ID" HorizontalAlignment="Left" Margin="10,10,0,0" VerticalAlignment="Top"/>

<Label Content="First Name" HorizontalAlignment="Left" Margin="10,42,0,0" VerticalAlignment="Top"/>

<Label Content="First Title" HorizontalAlignment="Left" Margin="10,101,0,0" VerticalAlignment="Top"/>

<Label Content="Film ID" HorizontalAlignment="Left" Margin="10,71,0,0" VerticalAlignment="Top"/>

<TextBox IsReadOnly="True" x:Name="txtMemberID" HorizontalAlignment="Left" Height="23" Margin="95,7,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

<TextBox IsReadOnly="True" x:Name="txtFirstName" HorizontalAlignment="Left" Height="23" Margin="95,37,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

<TextBox IsReadOnly="True" x:Name="txtFilmID" HorizontalAlignment="Left" Height="23" Margin="95,67,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

<TextBox IsReadOnly="True" x:Name="txtTitle" HorizontalAlignment="Left" Height="23" Margin="95,98,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

<Label Content="Rent Date" HorizontalAlignment="Left" Margin="280,10,0,0" VerticalAlignment="Top"/>

<Label Content="Return Date" HorizontalAlignment="Left" Margin="280,42,0,0" VerticalAlignment="Top"/>

<TextBox IsReadOnly="True" x:Name="txtRentDate" HorizontalAlignment="Left" Height="23" Margin="369,7,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

<TextBox IsReadOnly="True" x:Name="txtReturnDate" HorizontalAlignment="Left" Height="23" Margin="369,37,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

<Button x:Name="btnReturn" Content="Return Movie" HorizontalAlignment="Left" Margin="555,7,0,0" VerticalAlignment="Top" Width="183"/>

<Button x:Name="btnPostpone" Content="Postpone" HorizontalAlignment="Left" Margin="555,42,0,0" VerticalAlignment="Top" Width="183"/>

<Label x:Name="lblError" Foreground="Red" Content="" HorizontalAlignment="Left" Margin="555,85,0,0" VerticalAlignment="Top" Width="183" Height="23"/>

<Label x:Name="lblSuccess" Foreground="Green" Content="" HorizontalAlignment="Left" Margin="555,85,0,0" VerticalAlignment="Top" Width="183" Height="23"/>

</Grid>

</GroupBox>

</Grid>

</Page>

**rentedMovies.xaml.vb**

Imports System.Data

Imports System.Windows.Threading

Class rentedMovies

Private db As DBTransaction

Private dTableRentedFilms As DataTable

'Sub that fills the data grid with the list of all rented movies when the grid is loaded

Private Sub Grid\_Loaded\_1(sender As Object, e As RoutedEventArgs)

db = New DBTransaction()

dTableRentedFilms = New DataTable()

dTableRentedFilms = db.getAllRentedMovies()

dgRentedFilms.ItemsSource = dTableRentedFilms.DefaultView

lblError.Content = ""

lblSuccess.Content = ""

clearFields()

End Sub

'Sub that fills the fields with the data of the selected movie

Private Sub dgRentedFilms\_MouseDoubleClick(sender As Object, e As MouseButtonEventArgs) Handles dgRentedFilms.MouseDoubleClick

Dim dr As DataRowView = dgRentedFilms.SelectedItem

txtFilmID.Text = dr.Item("film\_id")

txtFirstName.Text = dr.Item("member\_first\_name")

txtMemberID.Text = dr.Item("member\_id")

txtRentDate.Text = dr.Item("rent\_date")

txtReturnDate.Text = dr.Item("return\_date")

txtTitle.Text = dr.Item("film\_title")

lblError.Content = ""

lblSuccess.Content = ""

End Sub

'Sub that clears the fields

Private Sub clearFields()

txtFilmID.Text = ""

txtFirstName.Text = ""

txtMemberID.Text = ""

txtRentDate.Text = ""

txtReturnDate.Text = ""

txtTitle.Text = ""

End Sub

'Button that returns a movie

Private Sub btnReturn\_Click(sender As Object, e As RoutedEventArgs) Handles btnReturn.Click

If txtFilmID.Text.Length <> 0 And txtMemberID.Text.Length <> 0 Then

Dim fID As Integer = CType(txtFilmID.Text, Integer)

Dim mID As Integer = CType(txtMemberID.Text, Integer)

Dim rDate As Date = CType(txtRentDate.Text, Date)

Dim rs As Integer = db.returnMovie(fID, mID, rDate)

Dim rs1 As Integer = db.incrementMovieStock(fID)

If rs > 0 Then

dgRentedFilms.ItemsSource = Nothing

dTableRentedFilms.Dispose()

dTableRentedFilms = db.getAllRentedMovies()

dgRentedFilms.ItemsSource = dTableRentedFilms.DefaultView

lblSuccess.Content = "Returned"

lblError.Content = ""

clearFields()

Else

lblError.Content = "Please try again."

lblSuccess.Content = ""

End If

Else

lblError.Content = "Please select a rent."

lblSuccess.Content = ""

End If

End Sub

'Button that postpones the movie's deadline with one day

Private Sub btnPostpone\_Click(sender As Object, e As RoutedEventArgs) Handles btnPostpone.Click

If txtFilmID.Text.Length <> 0 And txtMemberID.Text.Length <> 0 Then

Dim fID As Integer = CType(txtFilmID.Text, Integer)

Dim mID As Integer = CType(txtMemberID.Text, Integer)

Dim rDate As Date = CType(txtRentDate.Text, Date)

Dim rs As Integer = db.postponeRent(fID, mID, rDate)

If rs > 0 Then

dTableRentedFilms.Clear()

dTableRentedFilms = db.getAllRentedMovies()

dgRentedFilms.ItemsSource = dTableRentedFilms.DefaultView

lblSuccess.Content = "Postponed"

lblError.Content = ""

clearFields()

Else

lblError.Content = "Please try again"

lblSuccess.Content = ""

End If

Else

lblError.Content = "Please select a rent"

lblSuccess.Content = ""

End If

End Sub

End Class

**f) Sales report**

1. Administrator sales

**salesMovies.xaml**

<Page x:Class="salesMovies"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="400" d:DesignWidth="700"

Title="salesMovies">

<Grid Name="Grid1">

<GroupBox Header="Select interval" Height="193" HorizontalAlignment="Left" Margin="12,12,0,0" Name="GroupBox1" VerticalAlignment="Top" Width="248">

<Grid>

<Label Content="Begin date:" Height="28" HorizontalAlignment="Left" Margin="6,6,0,0" Name="Label1" VerticalAlignment="Top" />

<Label Content="End date:" Height="28" HorizontalAlignment="Left" Margin="6,44,0,0" Name="Label2" VerticalAlignment="Top" />

<DatePicker Height="25" HorizontalAlignment="Left" Margin="82,6,0,0" Name="dtpBegin" VerticalAlignment="Top" Width="148" />

<DatePicker Height="25" HorizontalAlignment="Left" Margin="82,44,0,0" Name="dtpEnd" VerticalAlignment="Top" Width="148" />

<Button Content="Show sales" Height="29" HorizontalAlignment="Left" Margin="50,89,0,0" Name="btnShowSales" VerticalAlignment="Top" Width="117" />

<Button Content="Generate PDF" Height="29" HorizontalAlignment="Left" Margin="50,132,0,0" x:Name="btnReport" VerticalAlignment="Top" Width="117" />

</Grid>

</GroupBox>

<DataGrid AutoGenerateColumns="False" Height="342" HorizontalAlignment="Left" Margin="275,12,0,0" Name="dgSales" VerticalAlignment="Top" Width="413" >

<DataGrid.Columns >

<DataGridTextColumn Header="Title" Binding="{Binding Path=film\_title}"></DataGridTextColumn>

<DataGridTextColumn Header="Price" Binding="{Binding Path=film\_price}"></DataGridTextColumn>

<DataGridTextColumn Header="Cost" Binding="{Binding Path=film\_cost}"></DataGridTextColumn>

<DataGridTextColumn Header="Sale date" Binding="{Binding Path=film\_sale\_date, StringFormat=d}"></DataGridTextColumn>

</DataGrid.Columns>

</DataGrid>

<Label Content="Total sales:" Height="28" HorizontalAlignment="Left" Margin="275,360,0,0" Name="Label3" VerticalAlignment="Top" />

<TextBox IsReadOnly="True" Height="23" HorizontalAlignment="Left" Margin="347,360,0,0" Name="txtTotalSales" VerticalAlignment="Top" Width="341" />

</Grid>

</Page>

**salesMovies.xaml.vb**

Imports System.Data

Imports iTextSharp.text

Imports iTextSharp.text.pdf

Imports System.IO

Class salesMovies

Private db As DBTransaction

Private dTableSales As DataTable

'Sub that's used to initialize some variable to be filled with the movies' sales

Private Sub Grid1\_Loaded(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles Grid1.Loaded

db = New DBTransaction()

dTableSales = New DataTable()

dgSales.ItemsSource = Nothing

dtpEnd.SelectedDate = Date.Now

dtpBegin.SelectedDate = Date.Now.AddDays(-1)

btnReport.IsEnabled = False

End Sub

'Button used to show movies' sales between the selected interval

Private Sub btnShowSales\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnShowSales.Click

Dim date1 As Date = dtpBegin.Text

Dim date2 As Date = dtpEnd.Text

dTableSales.Clear()

dTableSales = db.getAllmovieSalesTable(date1, date2)

dgSales.ItemsSource = dTableSales.DefaultView()

Dim total As Double = 0.0

If dTableSales.Rows.Count <> 0 Then

total = dTableSales.Compute("SUM(film\_price)", "")

End If

txtTotalSales.Text = total.ToString()

btnReport.IsEnabled = True

End Sub

'Sub fired when clicked the button generate in order to generate a PDF report of the movies' sales for the selected date interval

Private Sub btnReport\_Click(sender As Object, e As RoutedEventArgs) Handles btnReport.Click

Dim fileName As String = InputBox("Enter the filename:", "Report name:")

fileName = fileName & ".pdf"

Dim path As String = Utility.getDrive() & ":\Reports\" & fileName

While File.Exists(path)

fileName = InputBox("File already exists. Enter a new one:", "Report name:")

fileName = fileName & ".pdf"

path = Utility.getDrive() & ":\Reports\" & fileName

End While

Dim pdfRepoert As New Document()

Dim pdfWriter As PdfWriter = pdfWriter.GetInstance(pdfRepoert, New FileStream(path, FileMode.Create))

pdfRepoert.Open()

Dim pdfTable1 As New PdfPTable(2)

Dim cellOne As PdfPCell = New PdfPCell(New Phrase("Video Store Movie Sales"))

cellOne.Colspan = 2

cellOne.HorizontalAlignment = 1

cellOne.Border = Rectangle.NO\_BORDER

pdfTable1.AddCell(cellOne)

Dim cellTwo As PdfPCell = New PdfPCell(New Phrase("Begin date: " & dtpBegin.Text.ToString()))

cellTwo.Border = Rectangle.NO\_BORDER

pdfTable1.AddCell(cellTwo)

Dim cellThree As PdfPCell = New PdfPCell(New Phrase("End date: " & dtpEnd.Text.ToString()))

cellThree.Border = Rectangle.NO\_BORDER

pdfTable1.AddCell(cellThree)

pdfRepoert.Add(pdfTable1)

pdfRepoert.Add(New Paragraph(" "))

pdfRepoert.Add(New Paragraph(" "))

Dim salesTable As New PdfPTable(4)

salesTable.AddCell("Title")

salesTable.AddCell("Price")

salesTable.AddCell("Cost")

salesTable.AddCell("Sale Date")

Dim index As Integer = 0

While index < dTableSales.Rows.Count

salesTable.AddCell(dTableSales.Rows(index).Item("film\_title"))

salesTable.AddCell(dTableSales.Rows(index).Item("film\_price"))

salesTable.AddCell(dTableSales.Rows(index).Item("film\_cost"))

salesTable.AddCell(dTableSales.Rows(index).Item("film\_sale\_date"))

index += 1

End While

pdfRepoert.Add(salesTable)

pdfRepoert.Add(New Paragraph(" "))

pdfRepoert.Add(New Paragraph(" "))

Dim totalSales As New PdfPTable(2)

totalSales.WidthPercentage = 50

Dim c1 As PdfPCell = New PdfPCell(New Phrase("Total Sales:"))

c1.Border = Rectangle.NO\_BORDER

Dim c2 As PdfPCell = New PdfPCell(New Phrase(txtTotalSales.Text.ToString() & " LEK"))

c2.Border = Rectangle.NO\_BORDER

totalSales.AddCell(c1)

totalSales.AddCell(c2)

pdfRepoert.Add(totalSales)

pdfRepoert.Close()

System.Diagnostics.Process.Start(path)

End Sub

End Class

**salesGames.xaml**

<Page x:Class="salesGames"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="400" d:DesignWidth="700"

Title="salesGames">

<Grid Loaded="Grid\_Loaded\_1">

<GroupBox Header="Select interval" Height="180" HorizontalAlignment="Left" Margin="12,12,0,0" Name="GroupBox1" VerticalAlignment="Top" Width="248">

<Grid Height="160" VerticalAlignment="Top" Margin="0,0,0,-2">

<Label Content="Begin date:" Height="28" HorizontalAlignment="Left" Margin="6,6,0,0" Name="Label1" VerticalAlignment="Top" />

<Label Content="End date:" Height="28" HorizontalAlignment="Left" Margin="6,44,0,0" Name="Label2" VerticalAlignment="Top" />

<DatePicker Height="25" HorizontalAlignment="Left" Margin="82,6,0,0" Name="dtpBegin" VerticalAlignment="Top" Width="148" />

<DatePicker Height="25" HorizontalAlignment="Left" Margin="82,44,0,0" Name="dtpEnd" VerticalAlignment="Top" Width="148" />

<Button Content="Show sales" Height="29" HorizontalAlignment="Left" Margin="47,89,0,0" Name="btnShowSales" VerticalAlignment="Top" Width="129" />

<Button Content="Generate Report" Height="29" HorizontalAlignment="Left" Margin="47,123,0,0" x:Name="btnReport" VerticalAlignment="Top" Width="129" />

</Grid>

</GroupBox>

<DataGrid AutoGenerateColumns="False" Height="342" HorizontalAlignment="Left" Margin="275,12,0,0" Name="dgSales" VerticalAlignment="Top" Width="413" >

<DataGrid.Columns >

<DataGridTextColumn Header="Title" Binding="{Binding Path=game\_title}"></DataGridTextColumn>

<DataGridTextColumn Header="Price" Binding="{Binding Path=game\_price}"></DataGridTextColumn>

<DataGridTextColumn Header="Cost" Binding="{Binding Path=game\_cost}"></DataGridTextColumn>

<DataGridTextColumn Header="Sale date" Binding="{Binding Path=game\_sale\_date, StringFormat=d}"></DataGridTextColumn>

</DataGrid.Columns>

</DataGrid>

<Label Content="Total sales:" Height="28" HorizontalAlignment="Left" Margin="275,360,0,0" Name="Label3" VerticalAlignment="Top" />

<TextBox IsReadOnly="True" Height="23" HorizontalAlignment="Left" Margin="347,360,0,0" Name="txtTotalSales" VerticalAlignment="Top" Width="341" />

</Grid>

</Page>

**salesGames.xaml.vb**

Imports System.Data

Imports iTextSharp.text

Imports iTextSharp.text.pdf

Imports System.IO

Class salesGames

Private db As DBTransaction

Private dTableSales As DataTable

'Sub that's used to initialize some variable to be filled with the games' sales

Private Sub Grid\_Loaded\_1(sender As Object, e As RoutedEventArgs)

db = New DBTransaction()

dTableSales = New DataTable()

dgSales.ItemsSource = Nothing

dtpEnd.SelectedDate = Date.Now

dtpBegin.SelectedDate = Date.Now.AddDays(-1)

btnReport.IsEnabled = False

End Sub

'Button used to show games' sales between the selected interval

Private Sub btnShowSales\_Click(sender As Object, e As RoutedEventArgs) Handles btnShowSales.Click

Dim date1 As Date = dtpBegin.Text

Dim date2 As Date = dtpEnd.Text

dTableSales.Clear()

dTableSales = db.getAllGameSalesTable(date1, date2)

dgSales.ItemsSource = dTableSales.DefaultView()

Dim total As Double = 0.0

If dTableSales.Rows.Count <> 0 Then

total = dTableSales.Compute("SUM(game\_price)", "")

End If

txtTotalSales.Text = total.ToString()

btnReport.IsEnabled = True

End Sub

'Sub fired when clicked the button generate in order to generate a PDF report of the games' sales for the selected date interval

Private Sub btnReport\_Click(sender As Object, e As RoutedEventArgs) Handles btnReport.Click

Dim fileName As String = InputBox("Enter the filename:", "Report name:")

fileName = fileName & ".pdf"

Dim path As String = Utility.getDrive() & ":\Reports\" & fileName

While File.Exists(path)

fileName = InputBox("File already exists. Enter a new one:", "Report name:")

fileName = fileName & ".pdf"

path = Utility.getDrive() & ":\Reports\" & fileName

End While

Dim pdfRepoert As New Document()

Dim pdfWriter As PdfWriter = pdfWriter.GetInstance(pdfRepoert, New FileStream(path, FileMode.Create))

pdfRepoert.Open()

Dim pdfTable1 As New PdfPTable(2)

Dim cellOne As PdfPCell = New PdfPCell(New Phrase("Video Store Game Sales"))

cellOne.Colspan = 2

cellOne.HorizontalAlignment = 1

cellOne.Border = Rectangle.NO\_BORDER

pdfTable1.AddCell(cellOne)

Dim cellTwo As PdfPCell = New PdfPCell(New Phrase("Begin date: " & dtpBegin.Text.ToString()))

cellTwo.Border = Rectangle.NO\_BORDER

pdfTable1.AddCell(cellTwo)

Dim cellThree As PdfPCell = New PdfPCell(New Phrase("End date: " & dtpEnd.Text.ToString()))

cellThree.Border = Rectangle.NO\_BORDER

pdfTable1.AddCell(cellThree)

pdfRepoert.Add(pdfTable1)

pdfRepoert.Add(New Paragraph(" "))

pdfRepoert.Add(New Paragraph(" "))

Dim salesTable As New PdfPTable(4)

salesTable.AddCell("Title")

salesTable.AddCell("Price")

salesTable.AddCell("Cost")

salesTable.AddCell("Sale Date")

Dim index As Integer = 0

While index < dTableSales.Rows.Count

salesTable.AddCell(dTableSales.Rows(index).Item("game\_title"))

salesTable.AddCell(dTableSales.Rows(index).Item("game\_price"))

salesTable.AddCell(dTableSales.Rows(index).Item("game\_cost"))

salesTable.AddCell(dTableSales.Rows(index).Item("game\_sale\_date"))

index += 1

End While

pdfRepoert.Add(salesTable)

pdfRepoert.Add(New Paragraph(" "))

pdfRepoert.Add(New Paragraph(" "))

Dim totalSales As New PdfPTable(2)

totalSales.WidthPercentage = 50

Dim c1 As PdfPCell = New PdfPCell(New Phrase("Total Sales:"))

c1.Border = Rectangle.NO\_BORDER

Dim c2 As PdfPCell = New PdfPCell(New Phrase(txtTotalSales.Text.ToString() & " LEK"))

c2.Border = Rectangle.NO\_BORDER

totalSales.AddCell(c1)

totalSales.AddCell(c2)

pdfRepoert.Add(totalSales)

pdfRepoert.Close()

System.Diagnostics.Process.Start(path)

End Sub

End Class

**AccountingPanel.xaml**

<Page x:Class="AccountingPanel"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="720"

Title="AccountingPanel">

<Grid Loaded="Grid\_Loaded\_1">

<Label Content="" HorizontalAlignment="Left" Name="lblMonth" FontSize="18" Margin="325,10,0,0" VerticalAlignment="Top"/>

<Label Content="Total Movie Sales:" FontSize="14" HorizontalAlignment="Left" Margin="10,49,0,0" VerticalAlignment="Top"/>

<TextBox Name="txtMovieSales" FontSize="14" HorizontalAlignment="Left" Height="23" Margin="147,49,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150" IsReadOnly="True"/>

<Label Content="Total Game Sales:" FontSize="14" HorizontalAlignment="Left" Margin="10,93,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtGameSales" FontSize="14" HorizontalAlignment="Left" Height="23" Margin="147,89,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150" IsReadOnly="True"/>

<Label Content="Total Movie Rents:" FontSize="14" HorizontalAlignment="Left" Margin="10,132,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtMovieRents" FontSize="14" HorizontalAlignment="Left" Height="23" Margin="147,132,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150" IsReadOnly="True"/>

<Label Content="Employee Salaries:" FontSize="14" HorizontalAlignment="Left" Margin="10,253,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtEmployeeSalaries" FontSize="14" HorizontalAlignment="Left" Height="23" Margin="147,249,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150" IsReadOnly="True"/>

<Label Content="Employee Bonuses:" FontSize="14" HorizontalAlignment="Left" Margin="10,289,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtEmployeeBonuses" FontSize="14" HorizontalAlignment="Left" Height="23" Margin="147,289,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150" IsReadOnly="True"/>

<DataGrid AutoGenerateColumns="False" IsReadOnly="True" HorizontalAlignment="Left" Name="dgEmployees" Margin="325,50,0,0" VerticalAlignment="Top" Width="365" Height="182">

<DataGrid.Columns>

<DataGridTextColumn Header="First Name" Binding="{Binding Path=employee\_first\_name}"></DataGridTextColumn>

<DataGridTextColumn Header="Last Name" Binding="{Binding Path=employee\_last\_name}"></DataGridTextColumn>

<DataGridTextColumn Header="Salary" Binding="{Binding Path=employee\_salary}"></DataGridTextColumn>

<DataGridTextColumn Header="Bonus" Binding="{Binding Path=employee\_bonus}"></DataGridTextColumn>

</DataGrid.Columns>

</DataGrid>

<Label Content="Total Movie Cost:" FontSize="14" HorizontalAlignment="Left" Margin="10,173,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtMovieCost" FontSize="14" HorizontalAlignment="Left" Height="23" Margin="147,169,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150" IsReadOnly="True"/>

<TextBox x:Name="txtGameCost" FontSize="14" HorizontalAlignment="Left" Height="23" Margin="147,209,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150" IsReadOnly="True"/>

<Label Content="Total Game Cost:" FontSize="14" HorizontalAlignment="Left" Margin="10,213,0,0" VerticalAlignment="Top"/>

<Label Content="Energy:" FontSize="14" HorizontalAlignment="Left" Margin="10,331,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtEnergy" FontSize="14" HorizontalAlignment="Left" Height="23" Margin="147,327,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

<TextBox x:Name="txtInternet" FontSize="14" HorizontalAlignment="Left" Height="23" Margin="147,368,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

<Label Content="Internet:" FontSize="14" HorizontalAlignment="Left" Margin="10,372,0,0" VerticalAlignment="Top"/>

<Button Content="Calculate" Name="btnCalculate" HorizontalAlignment="Left" Margin="378,329,0,0" VerticalAlignment="Top" Width="92"/>

<Button Content="Generate Report" Name="btnGenerate" HorizontalAlignment="Left" Margin="512,329,0,0" VerticalAlignment="Top" Width="134"/>

<Label Content="Month Income:" FontSize="14" HorizontalAlignment="Left" Margin="325,374,0,0" VerticalAlignment="Top"/>

<TextBox IsReadOnly="True" HorizontalAlignment="Left" Height="23" Margin="445,370,0,0" TextWrapping="Wrap" Text="" Name="txtIncome" VerticalAlignment="Top" Width="245"/>

<Label Content="" Name="lblStatus" HorizontalAlignment="Left" Margin="302,289,0,0" VerticalAlignment="Top" Height="23" Width="408" Foreground="Red"/>

</Grid>

</Page>

**AccountingPannel.xaml.vb**

Imports System.Data

Imports System.IO

Imports iTextSharp.text

Imports iTextSharp.text.pdf

Class AccountingPanel

Private db As DBTransaction

Private dTable As DataTable

Dim salaries As Double = 0.0

Dim bonuses As Double = 0.0

Dim index As Integer = 0

Dim income As Double = 0.0

Dim mSales As Double

Dim mCost As Double

Dim gSales As Double

Dim gCost As Double

Dim mRents As Double

'Subroutine fired when loading the window in order to fill needed information

Private Sub Grid\_Loaded\_1(sender As Object, e As RoutedEventArgs)

Dim thisMonth As Integer = Month(Date.Now())

db = New DBTransaction()

dTable = New DataTable()

dTable = db.getAllEmployees()

While index < dTable.Rows.Count

salaries += dTable.Rows(index).Item("employee\_salary")

bonuses += dTable.Rows(index).Item("employee\_bonus")

index += 1

End While

Dim mSales As Double = db.getTotalMovieSales(Date.Now())

Dim mCost As Double = db.getTotalMoviesCost(Date.Now())

Dim gSales As Double = db.getTotalGamesSales(Date.Now())

Dim gCost As Double = db.getTotalGamesCost(Date.Now())

Dim mRents As Double = db.getTotalMovieRents(Date.Now())

lblMonth.Content = "Accouting Panel For Month: " & MonthName(Month(Now))

txtMovieSales.Text = mSales.ToString()

txtGameSales.Text = gSales.ToString()

txtMovieCost.Text = mCost.ToString()

txtGameCost.Text = gCost.ToString()

txtMovieRents.Text = mRents.ToString()

dgEmployees.ItemsSource = dTable.DefaultView()

txtEmployeeBonuses.Text = bonuses.ToString()

txtEmployeeSalaries.Text = salaries.ToString()

End Sub

'Subroutine fired when clicked the button calculate in order to calculate month income

Private Sub btnCalculate\_Click(sender As Object, e As RoutedEventArgs) Handles btnCalculate.Click

If txtInternet.Text.Length = 0 Or txtEnergy.Text.Length = 0 Then

lblStatus.Content = "Please fill the internet and energy."

Else

Dim internet As Double = CType(txtInternet.Text, Double)

Dim energy As Double = CType(txtEnergy.Text, Double)

income = (mSales + gSales + mRents) - (mCost + gCost + energy + internet + salaries + bonuses)

txtIncome.Text = "Total income this month: " & income & " LEK"

End If

End Sub

'Subroutine fired when clicked the button generate in order to generate a PDF report and close this month accounting

Private Sub btnGenerate\_Click(sender As Object, e As RoutedEventArgs) Handles btnGenerate.Click

Dim mr As Integer = MessageBox.Show("Do you want to close this month accounting generating a month report and reseting employees bonuses?", "Accounting", MessageBoxButton.YesNo)

If mr = vbYes Then

If txtIncome.Text.Length <> 0 Then

Dim fileName As String = InputBox("Enter the filename:", "Report name:")

fileName = fileName & ".pdf"

Dim path As String = Utility.getDrive() & ":\Reports\" & fileName

While File.Exists(path)

fileName = InputBox("File already exists. Enter a new one:", "Report name:")

fileName = fileName & ".pdf"

path = Utility.getDrive() & ":\Reports\" & fileName

End While

Dim pdfRepoert As New Document()

Dim pdfWriter As PdfWriter = pdfWriter.GetInstance(pdfRepoert, New FileStream(path, FileMode.Create))

pdfRepoert.Open()

Dim pdfTable1 As New PdfPTable(2)

Dim cellOne As PdfPCell = New PdfPCell(New Phrase("Video Store Month Report"))

cellOne.Colspan = 2

cellOne.HorizontalAlignment = 1

cellOne.Border = Rectangle.NO\_BORDER

pdfTable1.AddCell(cellOne)

pdfRepoert.Add(pdfTable1)

pdfRepoert.Add(New Paragraph(" "))

pdfRepoert.Add(New Paragraph(" "))

Dim p1 As Paragraph = New Paragraph("Total Movie Sales: " & txtMovieSales.Text.ToString() & " LEK")

Dim p2 As Paragraph = New Paragraph("Total Games Sales: " & txtGameSales.Text.ToString() & " LEK")

Dim p3 As Paragraph = New Paragraph("Total Movie Rents: " & txtMovieRents.Text.ToString() & " LEK")

Dim p4 As Paragraph = New Paragraph("Total Movie Costs: " & txtMovieCost.Text.ToString() & " LEK")

Dim p5 As Paragraph = New Paragraph("Total Game Costs: " & txtGameCost.Text.ToString() & " LEK")

Dim p6 As Paragraph = New Paragraph("Employee Salaries: " & txtEmployeeSalaries.Text.ToString() & " LEK")

Dim p7 As Paragraph = New Paragraph("Employee Bonuses: " & txtEmployeeBonuses.Text.ToString() & " LEK")

Dim p8 As Paragraph = New Paragraph("Energy Bill: " & txtEnergy.Text.ToString() & " LEK")

Dim p9 As Paragraph = New Paragraph("Internet Bill: " & txtInternet.Text.ToString() & " LEK")

Dim p10 As Paragraph = New Paragraph("Month Income: " & txtIncome.Text.ToString() & " LEK")

pdfRepoert.Add(p1)

pdfRepoert.Add(p2)

pdfRepoert.Add(p3)

pdfRepoert.Add(p4)

pdfRepoert.Add(p5)

pdfRepoert.Add(p6)

pdfRepoert.Add(p7)

pdfRepoert.Add(p8)

pdfRepoert.Add(p9)

pdfRepoert.Add(p10)

pdfRepoert.Add(New Paragraph(" "))

pdfRepoert.Add(New Paragraph(" "))

pdfRepoert.Close()

System.Diagnostics.Process.Start(path)

db.zeroBonuses()

clearFields()

Else

lblStatus.Content = "Please fill the internet and energy bill and calculate this month total income."

End If

End If

End Sub

'Sub to clear all text fields after the report for this month is generated

Private Sub clearFields()

txtEmployeeBonuses.Text = ""

txtEmployeeSalaries.Text = ""

txtEnergy.Text = ""

txtGameCost.Text = ""

txtGameSales.Text = ""

txtIncome.Text = ""

txtInternet.Text = ""

txtMovieCost.Text = ""

txtMovieRents.Text = ""

txtMovieSales.Text = ""

End Sub

End Class

2. Employee sales

**filmSales.xaml**

<Page x:Class="MySales"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="550" d:DesignWidth="750"

Title="MySales">

<Grid Loaded="Grid\_Loaded\_1">

<DataGrid IsReadOnly="True" AutoGenerateColumns="False" x:Name="dgSales" HorizontalAlignment="Left" Margin="10,31,0,0" VerticalAlignment="Top" Width="730" Height="406">

<DataGrid.Columns>

<DataGridTextColumn Header="Film Title" Binding="{Binding Path=film\_title}"></DataGridTextColumn>

<DataGridTextColumn Header="Sale Date" Binding="{Binding Path=film\_sale\_date, StringFormat=d}"></DataGridTextColumn>

<DataGridTextColumn Header="Film Price" Binding="{Binding Path=film\_price}"></DataGridTextColumn>

</DataGrid.Columns>

</DataGrid>

<Label Content="My Sales" HorizontalAlignment="Left" Margin="334,4,0,0" VerticalAlignment="Top" FontSize="18"/>

<Label Content="Total Sales" HorizontalAlignment="Left" Margin="12,461,0,0" VerticalAlignment="Top"/>

<TextBox IsReadOnly="True" x:Name="txtSales" HorizontalAlignment="Left" Height="23" Margin="87,461,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

</Grid>

</Page>

**filmSales.xaml.vb**

Imports System.Data

Class MySales

Private db As DBTransaction

Private fSalesDataTable As DataTable

'Sub that fills the table with all the sales of the movies of the employee logged in

Private Sub Grid\_Loaded\_1(sender As Object, e As RoutedEventArgs)

db = New DBTransaction()

fSalesDataTable = New DataTable()

fSalesDataTable = db.getMyFilmSales(MainWindow.getId())

dgSales.ItemsSource = fSalesDataTable.DefaultView

Dim total As Integer = fSalesDataTable.Rows.Count

txtSales.Text = total.ToString

End Sub

End Class

**gameSales.xaml**

<Page x:Class="gameSales"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="550" d:DesignWidth="750"

Title="gameSales">

<Grid Loaded="Grid\_Loaded\_1">

<DataGrid IsReadOnly="True" AutoGenerateColumns="False" x:Name="dgSales" HorizontalAlignment="Left" Margin="10,31,-440,-137" VerticalAlignment="Top" Width="730" Height="406">

<DataGrid.Columns>

<DataGridTextColumn Header="Game Title" Binding="{Binding Path=game\_title}"/>

<DataGridTextColumn Header="Sale Date" Binding="{Binding Path=game\_sale\_date, StringFormat=d}"/>

<DataGridTextColumn Header="Game Price" Binding="{Binding Path=game\_price}"/>

</DataGrid.Columns>

</DataGrid>

<Label Content="My Sales" HorizontalAlignment="Left" Margin="334,4,-104,0" VerticalAlignment="Top" FontSize="18"/>

<Label Content="Total Sales" HorizontalAlignment="Left" Margin="12,461,0,-177" VerticalAlignment="Top"/>

<TextBox IsReadOnly="True" x:Name="txtSales" HorizontalAlignment="Left" Height="23" Margin="87,461,0,-184" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

</Grid>

</Page>

**gameSales.xaml.vb**

Imports System.Data

Class gameSales

Private db As DBTransaction

Private gSalesDataTable As DataTable

'Sub that fill the table with all the sales of the games of the employee logged in

Private Sub Grid\_Loaded\_1(sender As Object, e As RoutedEventArgs)

db = New DBTransaction()

gSalesDataTable = New DataTable()

gSalesDataTable = db.getMyGameSales(MainWindow.getId())

dgSales.ItemsSource = gSalesDataTable.DefaultView

Dim total As Integer = gSalesDataTable.Rows.Count

txtSales.Text = total.ToString

End Sub

End Class

**g)Email a member**

**emailForRents.xaml**

<Page x:Class="emailForRents"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="600"

Title="emailForRents">

<Grid Loaded="Grid\_Loaded\_1">

<Label Content="Send email to passed rents" HorizontalAlignment="Left" Margin="10,10,0,0" VerticalAlignment="Top" FontSize="18"/>

<DataGrid IsReadOnly="True" AutoGenerateColumns="False" x:Name="dgMembers" HorizontalAlignment="Left" Margin="10,39,0,0" VerticalAlignment="Top" Width="580" Height="250">

<DataGrid.Columns>

<DataGridTextColumn Header="First Name" Binding="{Binding Path=member\_first\_name}"></DataGridTextColumn>

<DataGridTextColumn Header="Last Name" Binding="{Binding Path=member\_last\_name}"></DataGridTextColumn>

<DataGridTextColumn Header="Email" Binding="{Binding Path=member\_email}"></DataGridTextColumn>

<DataGridTextColumn Header="Title" Binding="{Binding Path=film\_title}"></DataGridTextColumn>

</DataGrid.Columns>

</DataGrid>

<GroupBox Header="Send Email" HorizontalAlignment="Left" Margin="10,294,0,0" VerticalAlignment="Top" Width="580" Height="146">

<Grid HorizontalAlignment="Left" Height="124" VerticalAlignment="Top" Width="568">

<Label Content="First Name:" HorizontalAlignment="Left" Margin="10,14,0,0" VerticalAlignment="Top"/>

<Label Content="Last Name" HorizontalAlignment="Left" Margin="10,44,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtFirstName" HorizontalAlignment="Left" Height="23" Margin="93,10,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

<TextBox x:Name="txtLastName" HorizontalAlignment="Left" Height="23" Margin="93,39,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

<Label Content="Email:" HorizontalAlignment="Left" Margin="10,72,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtEmail" HorizontalAlignment="Left" Height="23" Margin="93,67,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

<Label Content="Film Title:" HorizontalAlignment="Left" Margin="10,98,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtTitle" HorizontalAlignment="Left" Height="23" Margin="93,95,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="150"/>

<Button x:Name="btnSend" Content="Send Email" HorizontalAlignment="Left" Margin="298,7,0,0" VerticalAlignment="Top" Width="121"/>

<Label x:Name="lblError" Foreground="Red" Content="" HorizontalAlignment="Left" Margin="298,44,0,0" VerticalAlignment="Top" Width="121" Height="22"/>

<Label x:Name="lblSuccess" Foreground="Green" Content="" HorizontalAlignment="Left" Margin="298,44,0,0" VerticalAlignment="Top" Width="121" Height="22"/>

</Grid>

</GroupBox>

</Grid>

</Page>

**emailForRents.xaml.vb**

Imports System.Data

Imports System.Net.Mail

Class emailForRents

Private db As DBTransaction

Private dTableRents As DataTable

'Sub used to fill the table with all the passed rents when the page is opened

Private Sub Grid\_Loaded\_1(sender As Object, e As RoutedEventArgs)

db = New DBTransaction()

dTableRents = New DataTable()

dTableRents = db.getPassedRents()

dgMembers.ItemsSource = dTableRents.DefaultView

lblError.Content = ""

lblSuccess.Content = ""

End Sub

'Filling the fields with the data of the selected member

Private Sub dgMembers\_MouseDoubleClick(sender As Object, e As MouseButtonEventArgs) Handles dgMembers.MouseDoubleClick

Dim dr As DataRowView = dgMembers.SelectedItem

txtEmail.Text = dr.Item("member\_email")

txtFirstName.Text = dr.Item("member\_first\_name")

txtLastName.Text = dr.Item("member\_last\_name")

txtTitle.Text = dr.Item("film\_title")

lblError.Content = ""

lblSuccess.Content = ""

End Sub

'Button used to send an email to a member that has not returned the movie in time

Private Sub btnSend\_Click(sender As Object, e As RoutedEventArgs) Handles btnSend.Click

If txtTitle.Text.Length <> 0 Then

Dim fName As String = txtFirstName.Text.ToString()

Dim lName As String = txtLastName.Text.ToString()

Dim email As String = txtEmail.Text.ToString()

Dim title As String = txtTitle.Text.ToString()

Dim message As New MailMessage

Dim smtp As New SmtpClient

message.To.Add(email)

message.From = New MailAddress("video.store2000@gmail.com", "Video Store")

message.Subject = "Rent Reminder"

message.Body = "Hi " & fName & " " & lName & vbCrLf & "The deadline for returning the movies has passed. Please return the following movie as soon as possible: " & title

smtp.Host = "smtp.gmail.com"

smtp.Port = 587

smtp.EnableSsl = True

smtp.Credentials = New Net.NetworkCredential("video.store2000@gmail.com", "17May2000")

smtp.Send(message)

lblSuccess.Content = "Email sent."

lblError.Content = ""

Else

lblError.Content = "Select a rent."

lblSuccess.Content = ""

End If

End Sub

End Class

**h) Guest panel**

**NewMovies.xaml**

<Page x:Class="NewMovies"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="410" d:DesignWidth="500"

Title="NewMovies">

<Grid Name="Grid1">

<Label Content="ID:" Height="28" HorizontalAlignment="Left" Margin="12,12,0,0" Name="Label1" VerticalAlignment="Top" />

<Label Content="Title:" Height="28" HorizontalAlignment="Left" Margin="12,45,0,0" Name="Label2" VerticalAlignment="Top" />

<Label Content="Categories:" Height="28" HorizontalAlignment="Left" Margin="12,77,0,0" Name="Label3" VerticalAlignment="Top" />

<Label Content="Release Date:" Height="28" HorizontalAlignment="Left" Margin="12,110,0,0" Name="Label4" VerticalAlignment="Top" />

<Label Content="Runtime:" Height="28" HorizontalAlignment="Left" Margin="12,145,0,0" Name="Label5" VerticalAlignment="Top" />

<Label Content="Rating:" Height="28" HorizontalAlignment="Left" Margin="12,177,0,0" Name="Label6" VerticalAlignment="Top" />

<Label Content="Description:" Height="28" HorizontalAlignment="Left" Margin="12,208,0,0" Name="Label7" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="94,12,0,0" IsEnabled="False" Name="txtID" VerticalAlignment="Top" Width="150" Text="{Binding Path=film\_id}" />

<TextBox Height="23" Text="{Binding Path=film\_title}" HorizontalAlignment="Left" Margin="94,45,0,0" Name="txtTitle" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=film\_categories}" HorizontalAlignment="Left" Margin="94,77,0,0" Name="txtCategories" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=film\_release\_date, StringFormat=d}" HorizontalAlignment="Left" Margin="94,110,0,0" Name="txtRelease" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=film\_runtime}" HorizontalAlignment="Left" Margin="94,142,0,0" Name="txtRuntime" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=film\_rating}" HorizontalAlignment="Left" Margin="94,175,0,0" Name="txtRating" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<RichTextBox ScrollViewer.VerticalScrollBarVisibility="Visible" Height="100" HorizontalAlignment="Left" Margin="94,216,0,0" IsReadOnly="True" VerticalAlignment="Top" Width="150" >

<FlowDocument>

<Paragraph>

<Run x:Name="txtDesc" Text="{Binding Path=film\_description}" />

</Paragraph>

</FlowDocument>

</RichTextBox>

<Image Height="304" HorizontalAlignment="Left" Margin="259,12,0,0" Name="imgFilm" Stretch="Fill" VerticalAlignment="Top" Width="220" />

<TextBox Height="25" HorizontalAlignment="Left" Margin="198,364,0,0" Name="txtPosition" VerticalAlignment="Top" Width="120" />

<Button Content="Previous" Height="25" HorizontalAlignment="Left" Margin="106,365,0,0" Name="btnPrev" VerticalAlignment="Top" Width="86" />

<Button Content="First" Height="25" HorizontalAlignment="Left" Margin="28,365,0,0" Name="btnFirst" VerticalAlignment="Top" Width="75" />

<Button Content="Next" Height="25" HorizontalAlignment="Left" Margin="324,364,0,0" Name="btnNext" VerticalAlignment="Top" Width="75" />

<Button Content="Last" Height="25" HorizontalAlignment="Left" Margin="404,363,0,0" Name="btnLast" VerticalAlignment="Top" Width="75" />

</Grid>

</Page>

**NewMovies.xaml.vb**

Imports System.Data

Imports System.Data.SqlClient

Imports System.IO

Class NewMovies

Private db As DBTransaction

Private moviesTable As datatable

Private view As CollectionView

'Sub fired when page is loaded and calls two others subs to fill the page

Private Sub Grid1\_Loaded(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles Grid1.Loaded

fillDataTable()

showPosition()

End Sub

'Sub to show navigation in the page

Private Sub showPosition()

txtPosition.Text = view.CurrentPosition + 1 & " of " & view.Count

showPhoto()

End Sub

'Sub to fill the data table with new movies from database

Private Sub fillDataTable()

moviesTable = New DataTable()

db = New DBTransaction()

moviesTable = db.getNewMoviesTable()

view = CollectionViewSource.GetDefaultView(moviesTable)

DataContext = view

End Sub

'Button that moves to the next in the row

Private Sub btnNext\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnNext.Click

If view.CurrentPosition + 1 < view.Count Then

view.MoveCurrentToNext()

showPosition()

End If

End Sub

'Button that moves to the last in the row

Private Sub btnLast\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnLast.Click

view.MoveCurrentToLast()

showPosition()

End Sub

'Button that moves to the previous in the row

Private Sub btnPrev\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnPrev.Click

If view.CurrentPosition - 1 > -1 Then

view.MoveCurrentToPrevious()

showPosition()

End If

End Sub

'Button that moves to the first in the row

Private Sub btnFirst\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnFirst.Click

view.MoveCurrentToFirst()

showPosition()

End Sub

'Sub that shows the photo of the movie in the image field

Private Sub showPhoto()

Dim dr As DataRow = moviesTable.Rows(view.CurrentPosition)

Dim img() As Byte = dr.Item("film\_image")

Dim bitMap As BitmapImage = New BitmapImage()

Dim ms As MemoryStream = New MemoryStream(img)

bitMap.BeginInit()

bitMap.StreamSource = ms

bitMap.EndInit()

imgFilm.Source = bitMap

End Sub

End Class

**NewGames.xaml**

<Page x:Class="NewGames"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="410" d:DesignWidth="500"

Title="NewGames">

<Grid Name="Grid1">

<Label Content="ID:" Height="28" HorizontalAlignment="Left" Margin="12,12,0,0" Name="Label1" VerticalAlignment="Top" />

<Label Content="Title:" Height="28" HorizontalAlignment="Left" Margin="12,45,0,0" Name="Label2" VerticalAlignment="Top" />

<Label Content="Categories:" Height="28" HorizontalAlignment="Left" Margin="12,77,0,0" Name="Label3" VerticalAlignment="Top" />

<Label Content="Release Date:" Height="28" HorizontalAlignment="Left" Margin="12,110,0,0" Name="Label4" VerticalAlignment="Top" />

<Label Content="Producer:" Height="28" HorizontalAlignment="Left" Margin="12,145,0,0" Name="Label5" VerticalAlignment="Top" />

<Label Content="Rating:" Height="28" HorizontalAlignment="Left" Margin="12,177,0,0" Name="Label6" VerticalAlignment="Top" />

<Label Content="Description:" Height="28" HorizontalAlignment="Left" Margin="12,208,0,0" Name="Label7" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="94,12,0,0" IsEnabled="False" Name="txtID" VerticalAlignment="Top" Width="150" Text="{Binding Path=game\_id}" />

<TextBox Height="23" Text="{Binding Path=game\_title}" HorizontalAlignment="Left" Margin="94,45,0,0" Name="txtTitle" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=game\_categories}" HorizontalAlignment="Left" Margin="94,77,0,0" Name="txtCategories" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=game\_release\_date, StringFormat=d}" HorizontalAlignment="Left" Margin="94,110,0,0" Name="txtRelease" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=game\_producer}" HorizontalAlignment="Left" Margin="94,142,0,0" Name="txtProducer" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=game\_rating}" HorizontalAlignment="Left" Margin="94,175,0,0" Name="txtRating" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<RichTextBox ScrollViewer.VerticalScrollBarVisibility="Visible" Height="100" HorizontalAlignment="Left" Margin="94,216,0,0" IsReadOnly="True" VerticalAlignment="Top" Width="150" >

<FlowDocument>

<Paragraph>

<Run x:Name="txtDesc" Text="{Binding Path=game\_description}" />

</Paragraph>

</FlowDocument>

</RichTextBox>

<Image Height="304" HorizontalAlignment="Left" Margin="259,12,0,0" Name="imgGame" Stretch="Fill" VerticalAlignment="Top" Width="220" />

<TextBox Height="25" HorizontalAlignment="Left" Margin="198,364,0,0" Name="txtPosition" VerticalAlignment="Top" Width="120" />

<Button Content="Previous" Height="25" HorizontalAlignment="Left" Margin="106,365,0,0" Name="btnPrev" VerticalAlignment="Top" Width="86" />

<Button Content="First" Height="25" HorizontalAlignment="Left" Margin="28,365,0,0" Name="btnFirst" VerticalAlignment="Top" Width="75" />

<Button Content="Next" Height="25" HorizontalAlignment="Left" Margin="324,364,0,0" Name="btnNext" VerticalAlignment="Top" Width="75" />

<Button Content="Last" Height="25" HorizontalAlignment="Left" Margin="404,363,0,0" Name="btnLast" VerticalAlignment="Top" Width="75" />

</Grid>

</Page>

**NewGames.xaml.vb**

Imports System.Data

Imports System.Data.SqlClient

Imports System.IO

Class NewGames

Private db As DBTransaction

Private gamesTable As DataTable

Private view As CollectionView

'Sub fired when page is loaded and calls two others subs to fill the page

Private Sub Grid1\_Loaded(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles Grid1.Loaded

fillDataTable()

showPosition()

End Sub

'Sub to show navigation in the page

Private Sub showPosition()

txtPosition.Text = view.CurrentPosition + 1 & " of " & view.Count

showPhoto()

End Sub

'Sub to fill the data table with new games from database

Private Sub fillDataTable()

gamesTable = New DataTable()

db = New DBTransaction()

gamesTable = db.getNewGamesTable()

view = CollectionViewSource.GetDefaultView(gamesTable)

DataContext = view

End Sub

'Button that moves to the next in row

Private Sub btnNext\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnNext.Click

If view.CurrentPosition + 1 < view.Count Then

view.MoveCurrentToNext()

showPosition()

End If

End Sub

'Button that moves to the last in row

Private Sub btnLast\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnLast.Click

view.MoveCurrentToLast()

showPosition()

End Sub

'Button that moves to the previous in row

Private Sub btnPrev\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnPrev.Click

If view.CurrentPosition - 1 > -1 Then

view.MoveCurrentToPrevious()

showPosition()

End If

End Sub

'Button that moves to the first in row

Private Sub btnFirst\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnFirst.Click

view.MoveCurrentToFirst()

showPosition()

End Sub

'Sub that shows the photo of the game in the field for the image

Private Sub showPhoto()

Dim dr As DataRow = gamesTable.Rows(view.CurrentPosition)

Dim img() As Byte = dr.Item("game\_image")

Dim bitMap As BitmapImage = New BitmapImage()

Dim ms As MemoryStream = New MemoryStream(img)

bitMap.BeginInit()

bitMap.StreamSource = ms

bitMap.EndInit()

imgGame.Source = bitMap

End Sub

End Class

**TopRatedMovies.xaml**

<Page x:Class="TopRatedMovies"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="410" d:DesignWidth="500"

Title="TopRatedMovies">

<Grid Name="Grid1">

<Label Content="ID:" Height="28" HorizontalAlignment="Left" Margin="12,12,0,0" Name="Label1" VerticalAlignment="Top" />

<Label Content="Title:" Height="28" HorizontalAlignment="Left" Margin="12,45,0,0" Name="Label2" VerticalAlignment="Top" />

<Label Content="Categories:" Height="28" HorizontalAlignment="Left" Margin="12,77,0,0" Name="Label3" VerticalAlignment="Top" />

<Label Content="Release Date:" Height="28" HorizontalAlignment="Left" Margin="12,110,0,0" Name="Label4" VerticalAlignment="Top" />

<Label Content="Runtime:" Height="28" HorizontalAlignment="Left" Margin="12,145,0,0" Name="Label5" VerticalAlignment="Top" />

<Label Content="Rating:" Height="28" HorizontalAlignment="Left" Margin="12,177,0,0" Name="Label6" VerticalAlignment="Top" />

<Label Content="Description:" Height="28" HorizontalAlignment="Left" Margin="12,208,0,0" Name="Label7" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="94,12,0,0" IsEnabled="False" Name="txtID" VerticalAlignment="Top" Width="150" Text="{Binding Path=film\_id}" />

<TextBox Height="23" Text="{Binding Path=film\_title}" HorizontalAlignment="Left" Margin="94,45,0,0" Name="txtTitle" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=film\_categories}" HorizontalAlignment="Left" Margin="94,77,0,0" Name="txtCategories" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=film\_release\_date, StringFormat=d}" HorizontalAlignment="Left" Margin="94,110,0,0" Name="txtRelease" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=film\_runtime}" HorizontalAlignment="Left" Margin="94,142,0,0" Name="txtRuntime" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=film\_rating}" HorizontalAlignment="Left" Margin="94,175,0,0" Name="txtRating" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<RichTextBox ScrollViewer.VerticalScrollBarVisibility="Visible" Height="100" HorizontalAlignment="Left" Margin="94,216,0,0" IsReadOnly="True" VerticalAlignment="Top" Width="150" >

<FlowDocument>

<Paragraph>

<Run x:Name="txtDesc" Text="{Binding Path=film\_description}" />

</Paragraph>

</FlowDocument>

</RichTextBox>

<Image Height="304" HorizontalAlignment="Left" Margin="259,12,0,0" Name="imgFilm" Stretch="Fill" VerticalAlignment="Top" Width="220" />

<TextBox Height="25" HorizontalAlignment="Left" Margin="198,364,0,0" Name="txtPosition" VerticalAlignment="Top" Width="120" />

<Button Content="Previous" Height="25" HorizontalAlignment="Left" Margin="106,365,0,0" Name="btnPrev" VerticalAlignment="Top" Width="86" />

<Button Content="First" Height="25" HorizontalAlignment="Left" Margin="28,365,0,0" Name="btnFirst" VerticalAlignment="Top" Width="75" />

<Button Content="Next" Height="25" HorizontalAlignment="Left" Margin="324,364,0,0" Name="btnNext" VerticalAlignment="Top" Width="75" />

<Button Content="Last" Height="25" HorizontalAlignment="Left" Margin="404,363,0,0" Name="btnLast" VerticalAlignment="Top" Width="75" />

</Grid>

</Page>

**TopRatedMovies.xaml.vb**

Imports System.Data

Imports System.Data.SqlClient

Imports System.IO

Class TopRatedMovies

Private db As DBTransaction

Private moviesTable As DataTable

Private view As CollectionView

'Sub fired when page is loaded and calls two others subs to fill the page

Private Sub Grid1\_Loaded(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles Grid1.Loaded

fillDataTable()

showPosition()

End Sub

'Sub to show navigation in the page

Private Sub showPosition()

txtPosition.Text = view.CurrentPosition + 1 & " of " & view.Count

showPhoto()

End Sub

'Sub to fill the data table with top rated movies from database

Private Sub fillDataTable()

moviesTable = New DataTable()

db = New DBTransaction()

moviesTable = db.getMostRateMoview()

view = CollectionViewSource.GetDefaultView(moviesTable)

DataContext = view

End Sub

'Button that moves to the next in the row

Private Sub btnNext\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnNext.Click

If view.CurrentPosition + 1 < view.Count Then

view.MoveCurrentToNext()

showPosition()

End If

End Sub

'Button that moves to the last in the row

Private Sub btnLast\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnLast.Click

view.MoveCurrentToLast()

showPosition()

End Sub

'Button that moves to the previous in the row

Private Sub btnPrev\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnPrev.Click

If view.CurrentPosition - 1 > -1 Then

view.MoveCurrentToPrevious()

showPosition()

End If

End Sub

'Button that moves to the first in the row

Private Sub btnFirst\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnFirst.Click

view.MoveCurrentToFirst()

showPosition()

End Sub

'Sub that shows the photo of the movie in the image field

Private Sub showPhoto()

Dim dr As DataRow = moviesTable.Rows(view.CurrentPosition)

Dim img() As Byte = dr.Item("film\_image")

Dim bitMap As BitmapImage = New BitmapImage()

Dim ms As MemoryStream = New MemoryStream(img)

bitMap.BeginInit()

bitMap.StreamSource = ms

bitMap.EndInit()

imgFilm.Source = bitMap

End Sub

End Class

**TopRatedGames.xaml**

<Page x:Class="TopRatedGames"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="410" d:DesignWidth="500"

Title="TopRatedGames">

<Grid Name="Grid1">

<Label Content="ID:" Height="28" HorizontalAlignment="Left" Margin="12,12,0,0" Name="Label1" VerticalAlignment="Top" />

<Label Content="Title:" Height="28" HorizontalAlignment="Left" Margin="12,45,0,0" Name="Label2" VerticalAlignment="Top" />

<Label Content="Categories:" Height="28" HorizontalAlignment="Left" Margin="12,77,0,0" Name="Label3" VerticalAlignment="Top" />

<Label Content="Release Date:" Height="28" HorizontalAlignment="Left" Margin="12,110,0,0" Name="Label4" VerticalAlignment="Top" />

<Label Content="Runtime:" Height="28" HorizontalAlignment="Left" Margin="12,145,0,0" Name="Label5" VerticalAlignment="Top" />

<Label Content="Rating:" Height="28" HorizontalAlignment="Left" Margin="12,177,0,0" Name="Label6" VerticalAlignment="Top" />

<Label Content="Description:" Height="28" HorizontalAlignment="Left" Margin="12,208,0,0" Name="Label7" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="94,12,0,0" IsEnabled="False" Name="txtID" VerticalAlignment="Top" Width="150" Text="{Binding Path=game\_id}" />

<TextBox Height="23" Text="{Binding Path=game\_title}" HorizontalAlignment="Left" Margin="94,45,0,0" Name="txtTitle" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=game\_categories}" HorizontalAlignment="Left" Margin="94,77,0,0" Name="txtCategories" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=game\_release\_date, StringFormat=d}" HorizontalAlignment="Left" Margin="94,110,0,0" Name="txtRelease" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=game\_producer}" HorizontalAlignment="Left" Margin="94,142,0,0" Name="txtProducer" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=game\_rating}" HorizontalAlignment="Left" Margin="94,175,0,0" Name="txtRating" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<RichTextBox ScrollViewer.VerticalScrollBarVisibility="Visible" Height="100" HorizontalAlignment="Left" Margin="94,216,0,0" IsReadOnly="True" VerticalAlignment="Top" Width="150" >

<FlowDocument>

<Paragraph>

<Run x:Name="txtDesc" Text="{Binding Path=game\_description}" />

</Paragraph>

</FlowDocument>

</RichTextBox>

<Image Height="304" HorizontalAlignment="Left" Margin="259,12,0,0" Name="imgGame" Stretch="Fill" VerticalAlignment="Top" Width="220" />

<TextBox Height="25" HorizontalAlignment="Left" Margin="198,364,0,0" Name="txtPosition" VerticalAlignment="Top" Width="120" />

<Button Content="Previous" Height="25" HorizontalAlignment="Left" Margin="106,365,0,0" Name="btnPrev" VerticalAlignment="Top" Width="86" />

<Button Content="First" Height="25" HorizontalAlignment="Left" Margin="28,365,0,0" Name="btnFirst" VerticalAlignment="Top" Width="75" />

<Button Content="Next" Height="25" HorizontalAlignment="Left" Margin="324,364,0,0" Name="btnNext" VerticalAlignment="Top" Width="75" />

<Button Content="Last" Height="25" HorizontalAlignment="Left" Margin="404,363,0,0" Name="btnLast" VerticalAlignment="Top" Width="75" />

</Grid>

</Page>

**TopRatedGames.xaml.vb**

Imports System.Data

Imports System.Data.SqlClient

Imports System.IO

Class TopRatedGames

Private db As DBTransaction

Private gamesTable As DataTable

Private view As CollectionView

'Sub fired when page is loaded and calls two others subs to fill the page

Private Sub Grid1\_Loaded(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles Grid1.Loaded

fillDataTable()

showPosition()

End Sub

'Sub to show navigation in the page

Private Sub showPosition()

txtPosition.Text = view.CurrentPosition + 1 & " of " & view.Count

showPhoto()

End Sub

'Sub to fill the data table with top rated games from database

Private Sub fillDataTable()

gamesTable = New DataTable()

db = New DBTransaction()

gamesTable = db.getMostRateGame()

view = CollectionViewSource.GetDefaultView(gamesTable)

DataContext = view

End Sub

'Button that moves to the next in the row

Private Sub btnNext\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnNext.Click

If view.CurrentPosition + 1 < view.Count Then

view.MoveCurrentToNext()

showPosition()

End If

End Sub

'Button that moves to the last in the row

Private Sub btnLast\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnLast.Click

view.MoveCurrentToLast()

showPosition()

End Sub

'Button that moves to the previous in the row

Private Sub btnPrev\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnPrev.Click

If view.CurrentPosition - 1 > -1 Then

view.MoveCurrentToPrevious()

showPosition()

End If

End Sub

'Button that moves to the first in the row

Private Sub btnFirst\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnFirst.Click

view.MoveCurrentToFirst()

showPosition()

End Sub

'Sub that shows the photo of the game in the image field

Private Sub showPhoto()

Dim dr As DataRow = gamesTable.Rows(view.CurrentPosition)

Dim img() As Byte = dr.Item("game\_image")

Dim bitMap As BitmapImage = New BitmapImage()

Dim ms As MemoryStream = New MemoryStream(img)

bitMap.BeginInit()

bitMap.StreamSource = ms

bitMap.EndInit()

imgGame.Source = bitMap

End Sub

End Class

**TopSoldMovies.xaml**

<Page x:Class="TopSoldMovies"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="410" d:DesignWidth="500"

Title="TopSoldMovies">

<Grid>

<Label Content="ID:" Height="28" HorizontalAlignment="Left" Margin="12,12,0,0" Name="Label1" VerticalAlignment="Top" />

<Label Content="Title:" Height="28" HorizontalAlignment="Left" Margin="12,45,0,0" Name="Label2" VerticalAlignment="Top" />

<Label Content="Categories:" Height="28" HorizontalAlignment="Left" Margin="12,77,0,0" Name="Label3" VerticalAlignment="Top" />

<Label Content="Release Date:" Height="28" HorizontalAlignment="Left" Margin="12,110,0,0" Name="Label4" VerticalAlignment="Top" />

<Label Content="Runtime:" Height="28" HorizontalAlignment="Left" Margin="12,145,0,0" Name="Label5" VerticalAlignment="Top" />

<Label Content="Rating:" Height="28" HorizontalAlignment="Left" Margin="12,177,0,0" Name="Label6" VerticalAlignment="Top" />

<Label Content="Description:" Height="28" HorizontalAlignment="Left" Margin="12,208,0,0" Name="Label7" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="94,12,0,0" IsEnabled="False" Name="txtID" VerticalAlignment="Top" Width="150" Text="{Binding Path=film\_id}" />

<TextBox Height="23" Text="{Binding Path=film\_title}" HorizontalAlignment="Left" Margin="94,45,0,0" Name="txtTitle" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=film\_categories}" HorizontalAlignment="Left" Margin="94,77,0,0" Name="txtCategories" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=film\_release\_date, StringFormat=d}" HorizontalAlignment="Left" Margin="94,110,0,0" Name="txtRelease" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=film\_runtime}" HorizontalAlignment="Left" Margin="94,142,0,0" Name="txtProducer" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=film\_rating}" HorizontalAlignment="Left" Margin="94,175,0,0" Name="txtRating" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<RichTextBox ScrollViewer.VerticalScrollBarVisibility="Visible" Height="100" HorizontalAlignment="Left" Margin="94,216,0,0" IsReadOnly="True" VerticalAlignment="Top" Width="150" >

<FlowDocument>

<Paragraph>

<Run x:Name="txtDesc" Text="{Binding Path=film\_description}" />

</Paragraph>

</FlowDocument>

</RichTextBox>

<Image Height="304" HorizontalAlignment="Left" Margin="259,12,0,0" x:Name="imgFilm" Stretch="Fill" VerticalAlignment="Top" Width="220" />

<TextBox Height="25" HorizontalAlignment="Left" Margin="198,364,0,0" Name="txtPosition" VerticalAlignment="Top" Width="120" />

<Button Content="Previous" Height="25" HorizontalAlignment="Left" Margin="106,365,0,0" Name="btnPrev" VerticalAlignment="Top" Width="86" />

<Button Content="First" Height="25" HorizontalAlignment="Left" Margin="28,365,0,0" Name="btnFirst" VerticalAlignment="Top" Width="75" />

<Button Content="Next" Height="25" HorizontalAlignment="Left" Margin="324,364,0,0" Name="btnNext" VerticalAlignment="Top" Width="75" />

<Button Content="Last" Height="25" HorizontalAlignment="Left" Margin="404,363,0,0" Name="btnLast" VerticalAlignment="Top" Width="75" />

</Grid>

</Page>

**TopSoldMovies.xaml.vb**

Imports System.Data

Imports System.Data.SqlClient

Imports System.IO

Class TopSoldMovies

Private db As DBTransaction

Private moviesTable As DataTable

Private view As CollectionView

'Sub to show navigation in the page

Private Sub showPosition()

txtPosition.Text = view.CurrentPosition + 1 & " of " & view.Count

showPhoto()

End Sub

'Sub to fill the data table with top sold movies from database

Private Sub fillDataTable()

moviesTable = New DataTable()

db = New DBTransaction()

moviesTable = db.getMostSoldMoview()

view = CollectionViewSource.GetDefaultView(moviesTable)

DataContext = view

End Sub

'Button that moves to the next in the row

Private Sub btnNext\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnNext.Click

If view.CurrentPosition + 1 < view.Count Then

view.MoveCurrentToNext()

showPosition()

End If

End Sub

'Button that moves to the last in the row

Private Sub btnLast\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnLast.Click

view.MoveCurrentToLast()

showPosition()

End Sub

'Button that moves to the previous in the row

Private Sub btnPrev\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnPrev.Click

If view.CurrentPosition - 1 > -1 Then

view.MoveCurrentToPrevious()

showPosition()

End If

End Sub

'Button that moves to the first in the row

Private Sub btnFirst\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnFirst.Click

view.MoveCurrentToFirst()

showPosition()

End Sub

'Sub that shows the photo of the movie in the image field

Private Sub showPhoto()

Dim dr As DataRow = moviesTable.Rows(view.CurrentPosition)

Dim img() As Byte = dr.Item("film\_image")

Dim bitMap As BitmapImage = New BitmapImage()

Dim ms As MemoryStream = New MemoryStream(img)

bitMap.BeginInit()

bitMap.StreamSource = ms

bitMap.EndInit()

imgFilm.Source = bitMap

End Sub

'Sub fired when page is loaded and calls two others subs to fill the page

Private Sub Page\_Loaded(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles MyBase.Loaded

fillDataTable()

showPosition()

End Sub

End Class

**TopSoldGames.xaml**

<Page x:Class="TopSoldGames"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="410" d:DesignWidth="500"

Title="TopSoldGames">

<Grid Loaded="Grid\_Loaded\_1">

<Label Content="ID:" Height="28" HorizontalAlignment="Left" Margin="12,12,0,0" Name="Label1" VerticalAlignment="Top" />

<Label Content="Title:" Height="28" HorizontalAlignment="Left" Margin="12,45,0,0" Name="Label2" VerticalAlignment="Top" />

<Label Content="Categories:" Height="28" HorizontalAlignment="Left" Margin="12,77,0,0" Name="Label3" VerticalAlignment="Top" />

<Label Content="Release Date:" Height="28" HorizontalAlignment="Left" Margin="12,110,0,0" Name="Label4" VerticalAlignment="Top" />

<Label Content="Runtime:" Height="28" HorizontalAlignment="Left" Margin="12,145,0,0" Name="Label5" VerticalAlignment="Top" />

<Label Content="Rating:" Height="28" HorizontalAlignment="Left" Margin="12,177,0,0" Name="Label6" VerticalAlignment="Top" />

<Label Content="Description:" Height="28" HorizontalAlignment="Left" Margin="12,208,0,0" Name="Label7" VerticalAlignment="Top" />

<TextBox Height="23" HorizontalAlignment="Left" Margin="94,12,0,0" IsEnabled="False" Name="txtID" VerticalAlignment="Top" Width="150" Text="{Binding Path=game\_id}" />

<TextBox Height="23" Text="{Binding Path=game\_title}" HorizontalAlignment="Left" Margin="94,45,0,0" Name="txtTitle" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=game\_categories}" HorizontalAlignment="Left" Margin="94,77,0,0" Name="txtCategories" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=game\_release\_date, StringFormat=d}" HorizontalAlignment="Left" Margin="94,110,0,0" Name="txtRelease" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=game\_producer}" HorizontalAlignment="Left" Margin="94,142,0,0" Name="txtProducer" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<TextBox Height="23" Text="{Binding Path=game\_rating}" HorizontalAlignment="Left" Margin="94,175,0,0" Name="txtRating" IsEnabled="False" VerticalAlignment="Top" Width="150" />

<RichTextBox ScrollViewer.VerticalScrollBarVisibility="Visible" IsReadOnly="True" Height="100" HorizontalAlignment="Left" Margin="94,216,0,0" VerticalAlignment="Top" Width="150" >

<FlowDocument>

<Paragraph>

<Run x:Name="txtDesc" Text="{Binding Path=game\_description}" />

</Paragraph>

</FlowDocument>

</RichTextBox>

<Image Height="304" HorizontalAlignment="Left" Margin="259,12,0,0" Name="imgGame" Stretch="Fill" VerticalAlignment="Top" Width="220" />

<TextBox Height="25" HorizontalAlignment="Left" Margin="198,364,0,0" Name="txtPosition" VerticalAlignment="Top" Width="120" />

<Button Content="Previous" Height="25" HorizontalAlignment="Left" Margin="106,365,0,0" Name="btnPrev" VerticalAlignment="Top" Width="86" />

<Button Content="First" Height="25" HorizontalAlignment="Left" Margin="28,365,0,0" Name="btnFirst" VerticalAlignment="Top" Width="75" />

<Button Content="Next" Height="25" HorizontalAlignment="Left" Margin="324,364,0,0" Name="btnNext" VerticalAlignment="Top" Width="75" />

<Button Content="Last" Height="25" HorizontalAlignment="Left" Margin="404,363,0,0" Name="btnLast" VerticalAlignment="Top" Width="75" />

</Grid>

</Page>

**TopSoldGames.xaml.vb**

Imports System.Data

Imports System.Data.SqlClient

Imports System.IO

Class TopSoldGames

Private db As DBTransaction

Private gamesTable As DataTable

Private view As CollectionView

'Sub to show navigation in the page

Private Sub showPosition()

txtPosition.Text = view.CurrentPosition + 1 & " of " & view.Count

showPhoto()

End Sub

'Sub to fill the data table with top sold games from database

Private Sub fillDataTable()

gamesTable = New DataTable()

db = New DBTransaction()

gamesTable = db.getMostSoldGames()

view = CollectionViewSource.GetDefaultView(gamesTable)

DataContext = view

End Sub

'Button that moves to the next in the row

Private Sub btnNext\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnNext.Click

If view.CurrentPosition + 1 < view.Count Then

view.MoveCurrentToNext()

showPosition()

End If

End Sub

'Button that moves to the last in the row

Private Sub btnLast\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnLast.Click

view.MoveCurrentToLast()

showPosition()

End Sub

'Button that moves to the previous in the row

Private Sub btnPrev\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnPrev.Click

If view.CurrentPosition - 1 > -1 Then

view.MoveCurrentToPrevious()

showPosition()

End If

End Sub

'Button that moves to the first in the row

Private Sub btnFirst\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnFirst.Click

view.MoveCurrentToFirst()

showPosition()

End Sub

'Sub that shows the photo of the movie in the image field

Private Sub showPhoto()

Dim dr As DataRow = gamesTable.Rows(view.CurrentPosition)

Dim img() As Byte = dr.Item("game\_image")

Dim bitMap As BitmapImage = New BitmapImage()

Dim ms As MemoryStream = New MemoryStream(img)

bitMap.BeginInit()

bitMap.StreamSource = ms

bitMap.EndInit()

imgGame.Source = bitMap

End Sub

'Sub fired when page is loaded and calls two others subs to fill the page

Private Sub Grid\_Loaded\_1(sender As Object, e As RoutedEventArgs)

fillDataTable()

showPosition()

End Sub

End Class

**SearchMovie.xaml**

<Page x:Class="SearchMovie"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="700"

Title="SearchMovie">

<Grid Loaded="Grid\_Loaded\_1">

<DataGrid AutoGenerateColumns="False" IsReadOnly="True" x:Name="dgMovies" HorizontalAlignment="Left" Margin="10,39,0,0" VerticalAlignment="Top" Width="680" Height="360">

<DataGrid.Columns>

<DataGridTextColumn Header="ID" Binding="{Binding Path=film\_id}"></DataGridTextColumn>

<DataGridTextColumn Header="Title" Binding="{Binding Path=film\_title}"></DataGridTextColumn>

<DataGridTextColumn Header="Category" Binding="{Binding Path=film\_categories}"></DataGridTextColumn>

<DataGridTextColumn Header="Release Date" Binding="{Binding Path=film\_release\_date, StringFormat=d}"></DataGridTextColumn>

<DataGridTextColumn Header="Description" Width="220" Binding="{Binding Path=film\_description}"></DataGridTextColumn>

<DataGridTemplateColumn Header="Image">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Image Width="80" Height="80" Source="{Binding Path=film\_image}">

</Image>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

<Label Content="All Movies" FontSize="18" HorizontalAlignment="Left" Margin="10,10,0,0" VerticalAlignment="Top"/>

<Label Content="Filter By Title:" FontSize="16" HorizontalAlignment="Left" Margin="10,415,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtFilter" HorizontalAlignment="Left" Height="23" Margin="119,413,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="200"/>

</Grid>

</Page>

**SearchMovie.xaml.vb**

Imports System.Data

Class SearchMovie

Private db As DBTransaction

Private dTableMovies As DataTable

'Sub that filters the list depending on what you write in the textbox

Private Sub txtFilter\_TextChanged(sender As Object, e As TextChangedEventArgs) Handles txtFilter.TextChanged

dTableMovies.DefaultView.RowFilter = "film\_title LIKE '\*" & txtFilter.Text.ToString() & "\*'"

End Sub

'Sub that fills the data grid with all the movies in the database

Private Sub Grid\_Loaded\_1(sender As Object, e As RoutedEventArgs)

db = New DBTransaction()

dTableMovies = New DataTable()

dTableMovies = db.getAllMovies()

dgMovies.ItemsSource = dTableMovies.DefaultView

End Sub

End Class

**SearchGame.xaml**

<Page x:Class="SearchGame"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="450" d:DesignWidth="700"

Title="SearchGame">

<Grid Loaded="Grid\_Loaded\_1">

<Label Content="All Games" FontSize="18" HorizontalAlignment="Left" Margin="10,10,0,0" VerticalAlignment="Top"/>

<DataGrid AutoGenerateColumns="False" IsEnabled="True" x:Name="dgGames" HorizontalAlignment="Left" Margin="10,39,0,0" VerticalAlignment="Top" Width="680" Height="368">

<DataGrid.Columns>

<DataGridTextColumn Header="ID" Binding="{Binding Path=game\_id}"></DataGridTextColumn>

<DataGridTextColumn Header="Title" Binding="{Binding Path=game\_title}"></DataGridTextColumn>

<DataGridTextColumn Header="Category" Binding="{Binding Path=game\_categories}"></DataGridTextColumn>

<DataGridTextColumn Header="Release Date" Binding="{Binding Path=game\_release\_date, StringFormat=d}"></DataGridTextColumn>

<DataGridTextColumn Header="Rating" Binding="{Binding Path=game\_rating}"></DataGridTextColumn>

<DataGridTextColumn Header="Description" Width="320" Binding="{Binding Path=game\_description}"></DataGridTextColumn>

<DataGridTemplateColumn Header="Image">

<DataGridTemplateColumn.CellTemplate>

<DataTemplate>

<Image Width="80" Height="80" Source="{Binding Path=game\_image}">

</Image>

</DataTemplate>

</DataGridTemplateColumn.CellTemplate>

</DataGridTemplateColumn>

</DataGrid.Columns>

</DataGrid>

<Label Content="Filter By Title:" FontSize="16" HorizontalAlignment="Left" Margin="10,420,0,0" VerticalAlignment="Top"/>

<TextBox x:Name="txtFilter" HorizontalAlignment="Left" Height="23" Margin="109,420,0,0" TextWrapping="Wrap" Text="" VerticalAlignment="Top" Width="200"/>

</Grid>

</Page>

**SearchGame.xaml.vb**

Imports System.Data

Class SearchGame

Private db As DBTransaction

Private dTableGames As DataTable

'Sub that filters the list depending on what you write in the textbox

Private Sub txtFilter\_TextChanged(sender As Object, e As TextChangedEventArgs) Handles txtFilter.TextChanged

dTableGames.DefaultView.RowFilter = "game\_title LIKE '\*" & txtFilter.Text.ToString() & "\*'"

End Sub

'Sub that fills the data grid with all the games in the database

Private Sub Grid\_Loaded\_1(sender As Object, e As RoutedEventArgs)

db = New DBTransaction()

dTableGames = New DataTable()

dTableGames = db.getAllGames()

dgGames.ItemsSource = dTableGames.DefaultView

End Sub

End Class

**i) Help**

**manual.xaml**

<Page x:Class="manual"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="565" d:DesignWidth="1050"

Title="manual">

<Grid VerticalAlignment="Top">

<Label Content="Troubleshooting guide" FontSize="20" HorizontalAlignment="Left" Margin="390,22,0,-12" VerticalAlignment="Top"/>

<Label Content="a) Logging in" FontSize="18" HorizontalAlignment="Left" Margin="26,62,0,-25" VerticalAlignment="Top"/>

<Label FontSize="14" Content="During the logging in as an administrator or employee if the window says at the right – bottom corner: 'Invalid credentials', it means &#10;that the username or password or both are not written correctly. You should check again what you've written in both those boxes." HorizontalAlignment="Left" Margin="26,91,0,-74" VerticalAlignment="Top" Height="60" Width="998"/>

<Label Content="b) Adding a new item to the program" FontSize="18" HorizontalAlignment="Left" Margin="26,156,0,-79" VerticalAlignment="Top"/>

<Label FontSize="14" Content="When you add a new item to the program and you click the 'add', a label may appear saying: 'Please fill in all the fields'. You should &#10;check again if you have left any of those fields unfilled. " HorizontalAlignment="Left" Margin="26,185,0,-37" VerticalAlignment="Top" Height="49" Width="998"/>

<Label Content="c) Modifying/deleting something from the program" FontSize="18" HorizontalAlignment="Left" Margin="26,239,0,-122" VerticalAlignment="Top"/>

<Label FontSize="14" Content="When you click 'modify' or 'delete' without selecting any item from the list the program says: 'Please select an item from the list'. &#10;You should double click the item that you want to modify/delete and then proceed.

Once you've selected an item and you want &#10;to modify it, you should fill in all the fields before clicking 'modify' button or otherwise the program will require you to fill in all the fields.

" HorizontalAlignment="Left" Margin="26,268,0,-185" VerticalAlignment="Top" Height="74" Width="998"/>

<Label FontSize="18" Content="d) No sales are shown to the sales report of the administrator" HorizontalAlignment="Left" Margin="26,347,0,-190" VerticalAlignment="Top"/>

<Label FontSize="14" Content="Check the time interval again because you might have chosen a time interval with no sales." HorizontalAlignment="Left" Margin="26,376,0,-230" VerticalAlignment="Top" Height="51" Width="998"/>

<Label FontSize="18" Content="e) Program crushes" HorizontalAlignment="Left" Margin="26,432,0,-269" VerticalAlignment="Top"/>

<Label FontSize="14" Content="If the program crushes for no reason it's probably because you have put the wrong type of data in one of the fields of adding an item or modifying an item." HorizontalAlignment="Left" Margin="26,466,0,-320" VerticalAlignment="Top" Height="51" Width="998"/>

</Grid>

</Page>

**manual.xaml.vb**

Class manual

Dim manual As RichTextBox = New RichTextBox()

End Class

1. Changing password for administrator

**adminAccount.xaml**

<Page x:Class="adminAccount"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="300" d:DesignWidth="300"

Title="adminAccount">

<Grid>

<Label Content="New Password:" HorizontalAlignment="Left" Margin="10,10,0,0" VerticalAlignment="Top" Width="100"/>

<Label Content="Enter Again:" HorizontalAlignment="Left" Margin="10,46,0,0" VerticalAlignment="Top"/>

<PasswordBox x:Name="psd1" HorizontalAlignment="Left" Margin="116,10,0,0" VerticalAlignment="Top" Width="174"/>

<PasswordBox x:Name="psd2" HorizontalAlignment="Left" Margin="116,40,0,0" VerticalAlignment="Top" Width="174"/>

<Button x:Name="btnModify" Content="Modify" HorizontalAlignment="Left" Margin="103,84,0,0" VerticalAlignment="Top" Width="75"/>

<Label x:Name="lblError" Foreground="Red" Content="" HorizontalAlignment="Left" Margin="10,118,0,0" VerticalAlignment="Top" Width="280" Height="25" FontSize="18"/>

<Label x:Name="lblSuccess" Foreground="Green" Content="" HorizontalAlignment="Left" Margin="10,118,0,0" VerticalAlignment="Top" Width="280" Height="25" FontSize="18"/>

</Grid>

</Page>

**adminAccount.xaml.vb**

Class adminAccount

Private db As DBTransaction

'Button used for changing the password of the administrator in the database

Private Sub btnModify\_Click(sender As Object, e As RoutedEventArgs) Handles btnModify.Click

db = New DBTransaction()

Dim pas As String = psd1.Password

Dim pas1 As String = psd2.Password

If pas.Length = 0 Or pas1.Length = 0 Then

lblError.Content = "Please fill in both passwords."

Else

If pas <> pas1 Then

lblError.Content = "Password must match."

Else

Dim rs As Integer = db.modifyAdminPassword(pas)

If rs = 1 Then

lblSuccess.Content = "Password modified."

Else

lblError.Content = "Try again later."

End If

End If

End If

End Sub

End Class

2. Changing password for employee

**myAccount.xaml**

<Page x:Class="myAccount"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="300" d:DesignWidth="300"

Title="myAccount">

<Grid>

<Label Content="New Password:" HorizontalAlignment="Left" Margin="10,10,0,0" VerticalAlignment="Top" Width="84"/>

<Label Content="Enter Again:" HorizontalAlignment="Left" Margin="10,46,0,0" VerticalAlignment="Top"/>

<PasswordBox x:Name="psd1" HorizontalAlignment="Left" Margin="116,10,0,0" VerticalAlignment="Top" Width="174"/>

<PasswordBox x:Name="psd2" HorizontalAlignment="Left" Margin="116,40,0,0" VerticalAlignment="Top" Width="174"/>

<Button x:Name="btnModify" Content="Modify" HorizontalAlignment="Left" Margin="103,84,0,0" VerticalAlignment="Top" Width="75"/>

<Label x:Name="lblError" Foreground="Red" Content="" HorizontalAlignment="Left" Margin="10,118,0,0" VerticalAlignment="Top" Width="280" Height="25" FontSize="18"/>

<Label x:Name="lblSuccess" Foreground="Green" Content="" HorizontalAlignment="Left" Margin="10,118,0,0" VerticalAlignment="Top" Width="280" Height="25" FontSize="18"/>

</Grid>

</Page>

**myAccount.xaml.vb**

Class myAccount

Private db As DBTransaction

'Button used to modify the the password of the logged in employee

Private Sub btnModify\_Click(sender As Object, e As RoutedEventArgs) Handles btnModify.Click

db = New DBTransaction()

Dim pas As String = psd1.Password

Dim pas1 As String = psd2.Password

Dim id As Integer = MainWindow.getId()

If pas.Length = 0 Or pas1.Length = 0 Then

lblError.Content = "Please fill in both passwords."

Else

If pas <> pas1 Then

lblError.Content = "Password must match."

Else

Dim rs As Integer = db.modifyEmployeePassword(pas, id)

If rs = 1 Then

lblSuccess.Content = "Password modified."

Else

lblError.Content = "Try again later."

End If

End If

End If

End Sub

End Class

**j) Logging out**

**logout.xaml**

<Page x:Class="logOut"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

mc:Ignorable="d"

d:DesignHeight="300" d:DesignWidth="300"

Title="logOut">

<Grid>

<Button Content="Log Out" Name="btnLogOut" HorizontalAlignment="Left" Margin="98,130,0,0" VerticalAlignment="Top" Width="108"/>

</Grid>

</Page>

**logout.xaml.vb**

Imports FirstFloor.ModernUI.Windows.Controls

Class logOut

'Button used to close the current window and open the log-in window

Private Sub btnLogOut\_Click(sender As Object, e As RoutedEventArgs) Handles btnLogOut.Click

Dim newWindow As New LoginWindow()

newWindow.Show()

newWindow.Width = 250

newWindow.Height = 350

Dim mainMenu(1) As FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(1) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(1).DisplayName = "Log In As:"

newWindow.MenuLinkGroups.Add(mainMenu(1))

Dim admin As New FirstFloor.ModernUI.Presentation.Link

admin.DisplayName = "Administrator"

admin.Source = New Uri("/AdministratorLogin.xaml", UriKind.Relative)

Dim emp As New FirstFloor.ModernUI.Presentation.Link

emp.DisplayName = "Employee"

emp.Source = New Uri("/EmployeeLogin.xaml", UriKind.Relative)

Dim guest As New FirstFloor.ModernUI.Presentation.Link

guest.DisplayName = "Guest"

guest.Source = New Uri("/GuestLogin.xaml", UriKind.Relative)

mainMenu(1).Links.Add(admin)

mainMenu(1).Links.Add(emp)

mainMenu(1).Links.Add(guest)

Dim currentWindow As ModernWindow = ModernWindow.GetWindow(Me)

currentWindow.Close()

End Sub

End Class

**iii) Creating the setup for the program**

Figures below show step-by-step how is the setup of this program created.

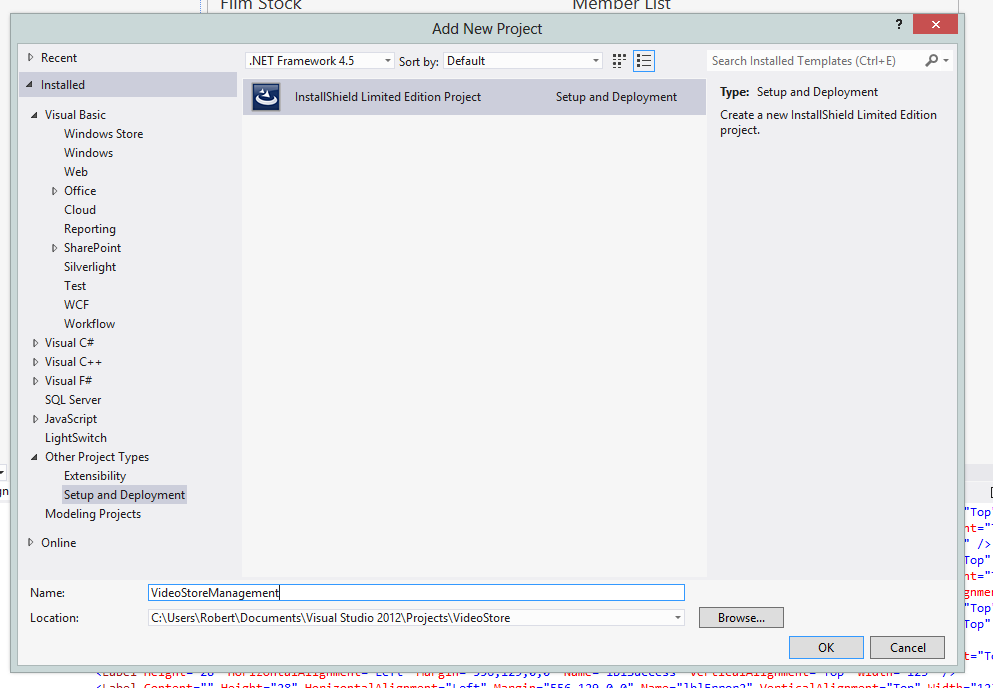


Fig. 7.1

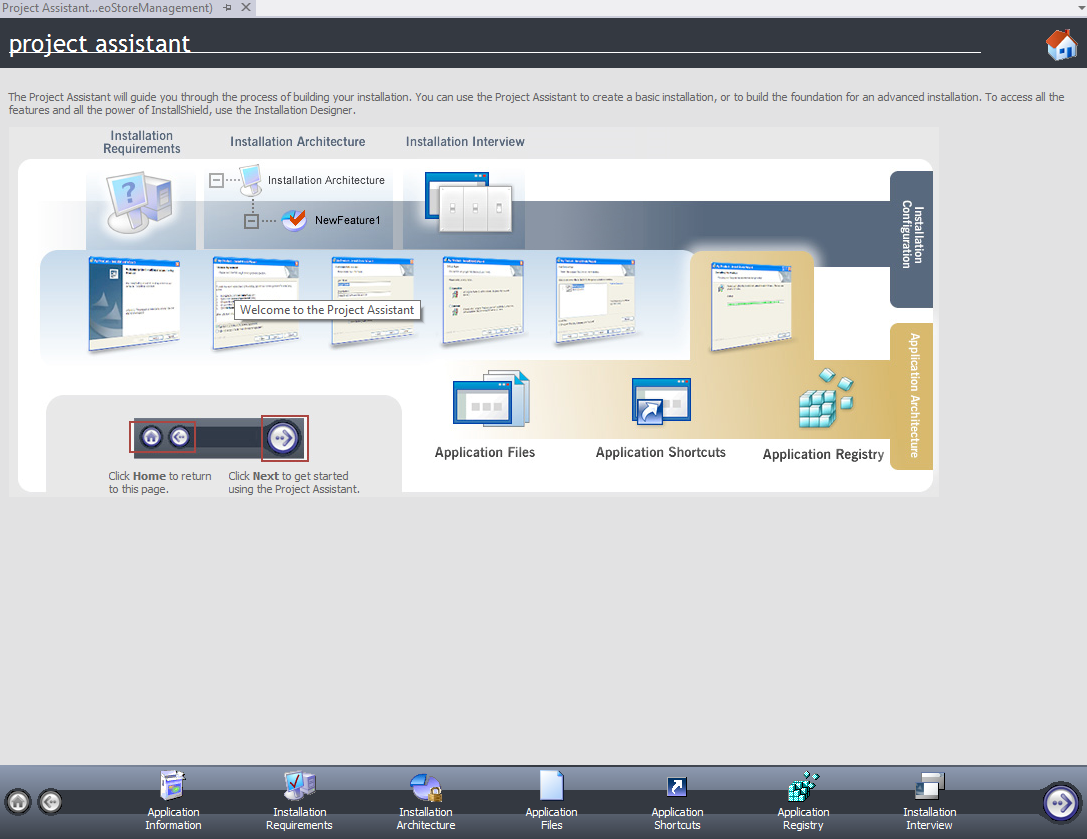


Fig. 7.2

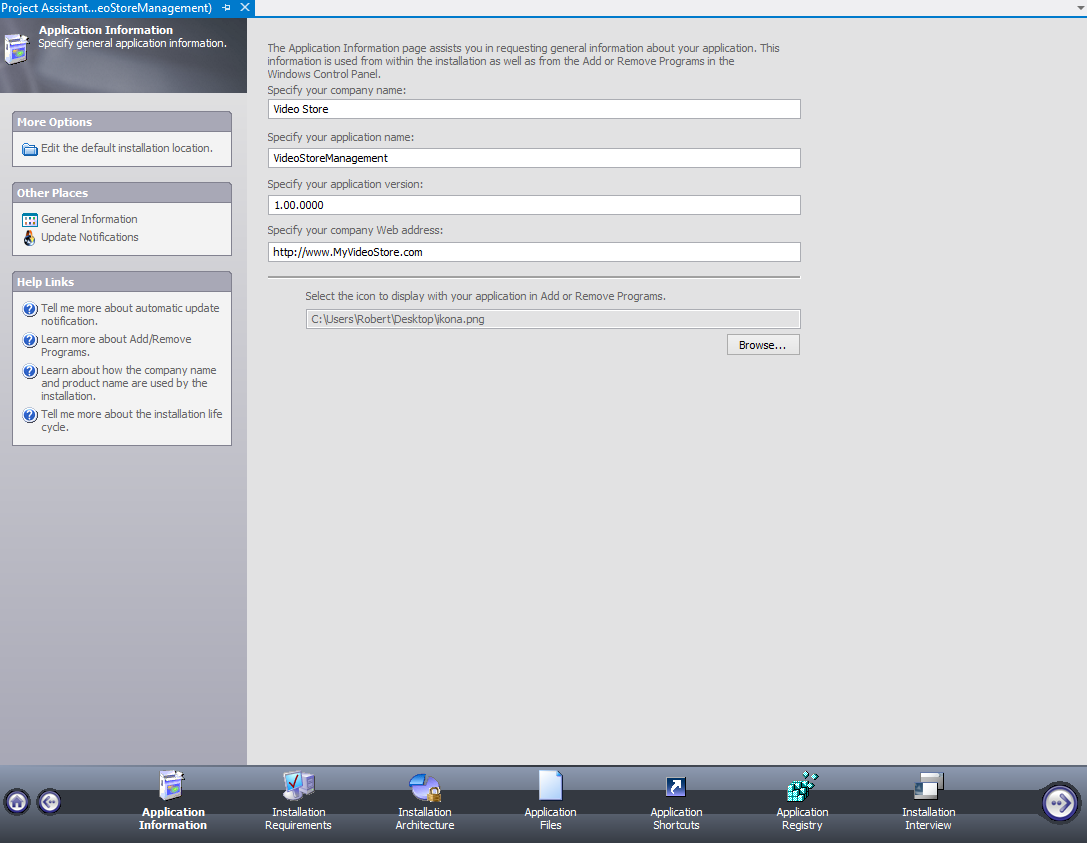


Fig. 7.3

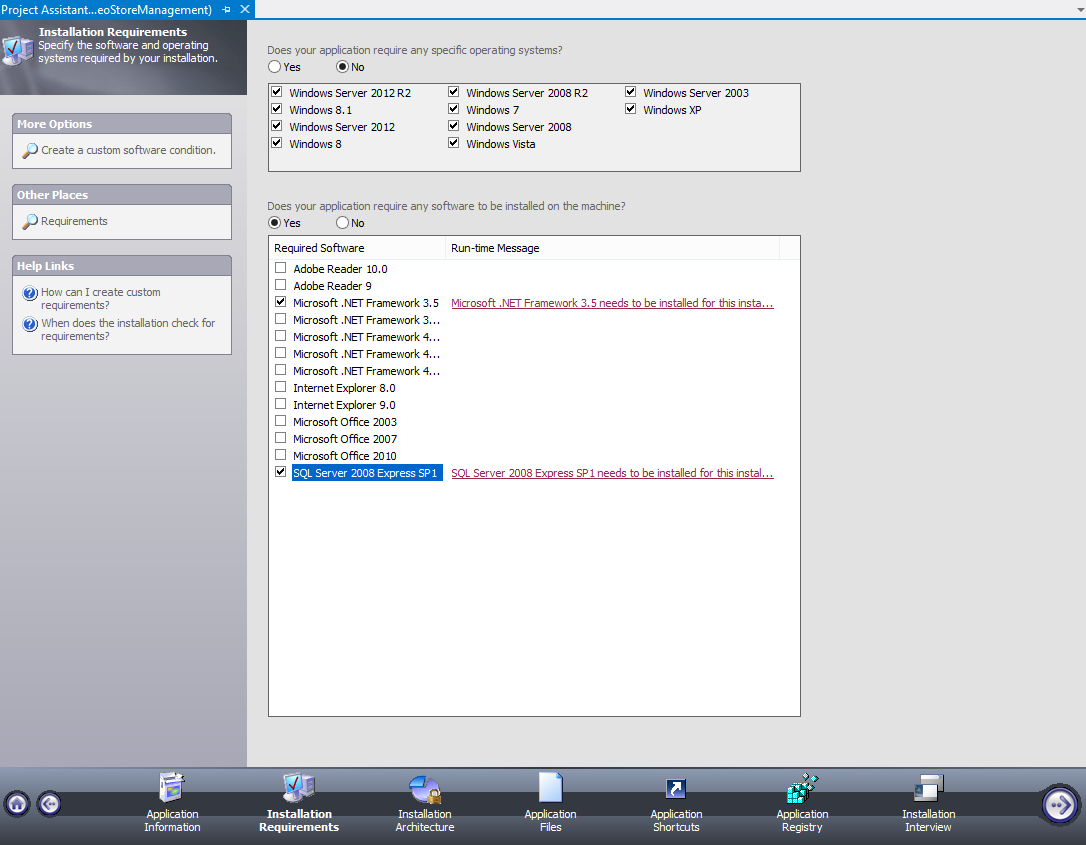


Fig. 7.4

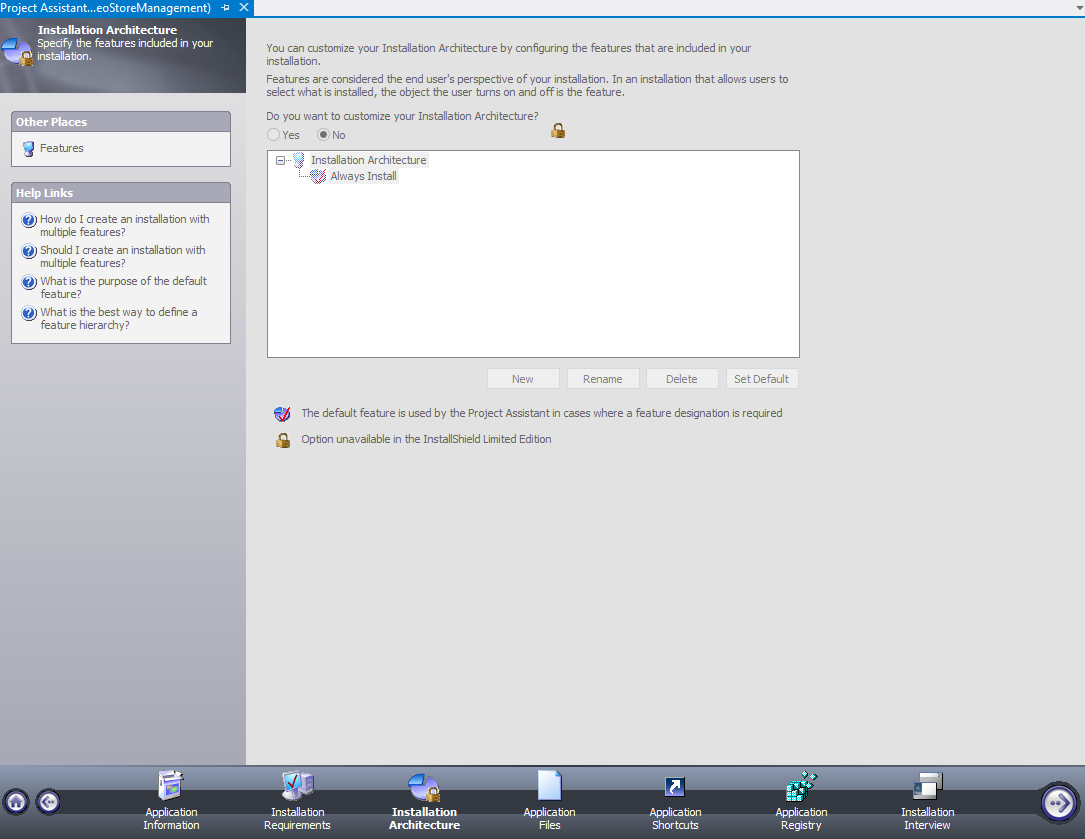


Fig. 7.5

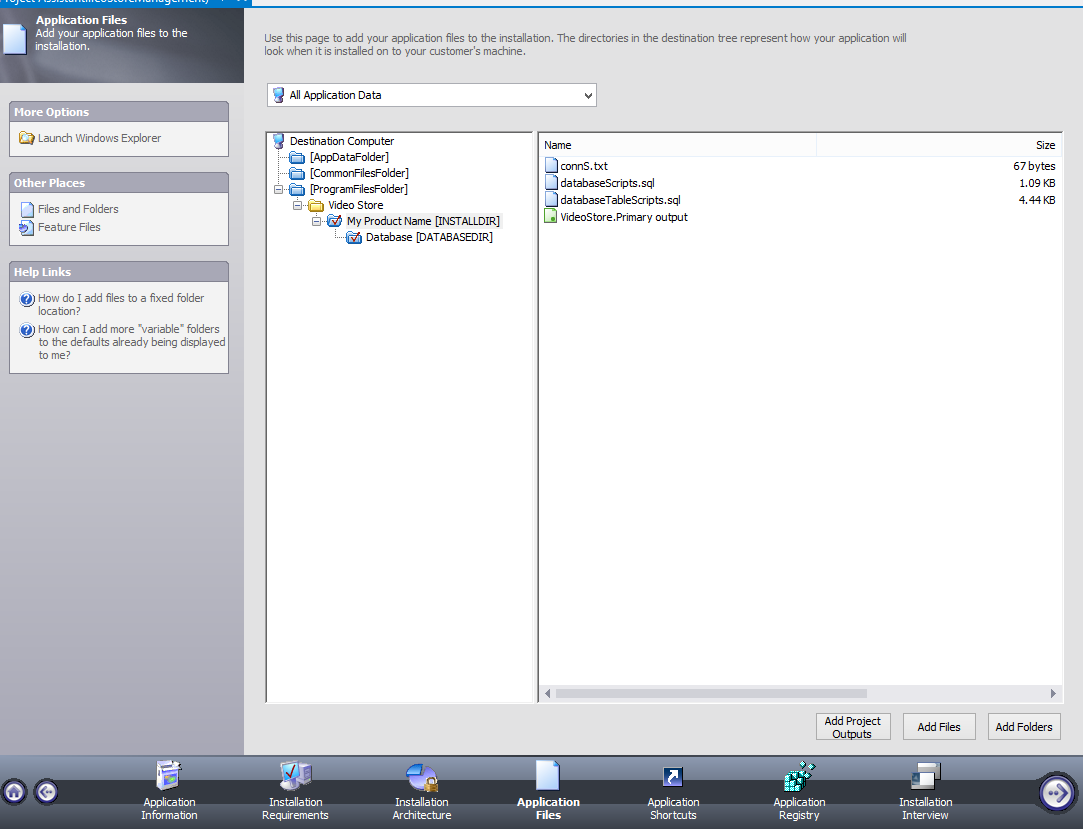
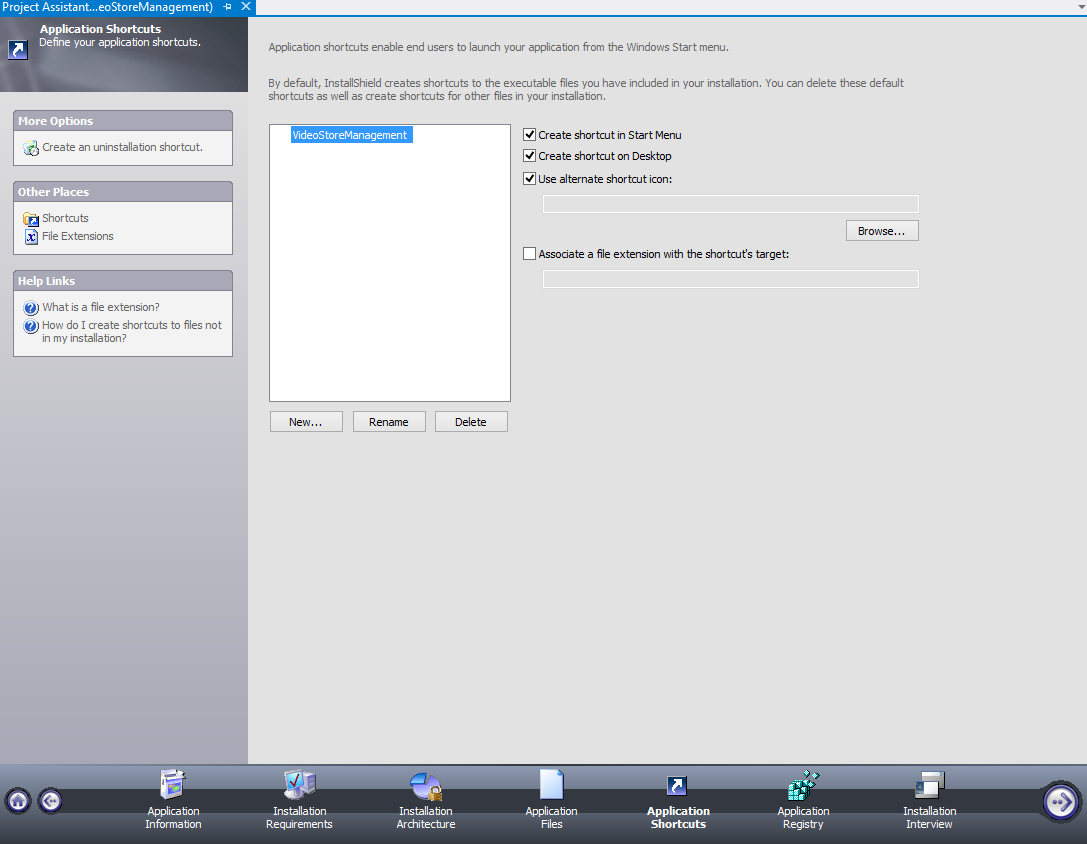


Fig. 7.6

Fig. 7.7

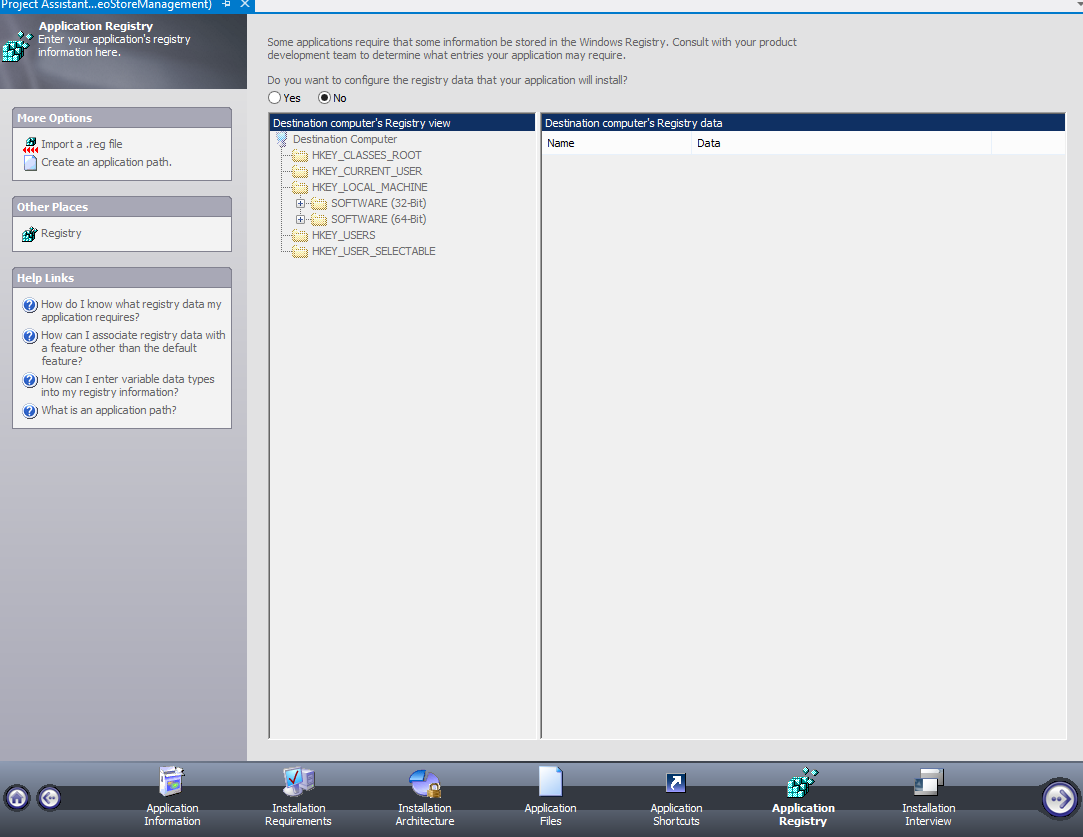


Fig. 7.8

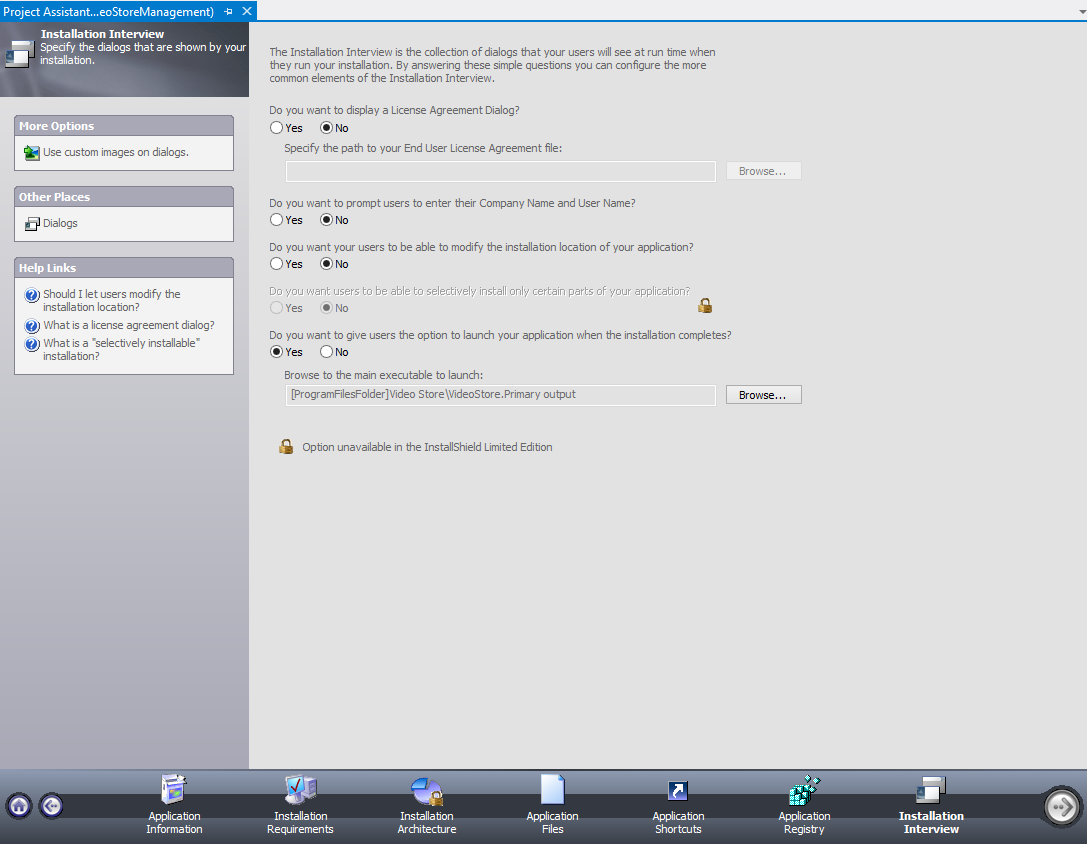


Fig. 7.9

After these steps the setup for this program is ready and now the program can be installed to any computer that fulfills the software and hardware requirements for this program.

**8. Testing**

**i) Alpha testing**

During this process most of the objectives that are set in the test plan are tested at this phase. If any error is found when the program is executed, it’s corrected directly.

Some errors in the database connections where found during this testing, most of which where grammatical errors in the SQL query, but they were immediately corrected.

**a) Adding a product**

Private Sub btnAddMovie\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnAddMovie.Click

'Getting the data from the inputs of the client

Dim name As String = txtName.Text

Dim category As String = txtCategory.Text

Dim releaseDate As Date = dtpReleaseDate.Text

Dim runtime As String = txtRuntime.Text

Dim rating As String = txtRating.Text

Dim txtRange As New TextRange(txtDescription.Document.ContentStart, txtDescription.Document.ContentEnd)

Dim description As String = txtRange.Text

Dim stock As String = txtStock.Text

Dim price As String = txtPrice.Text

Dim cost As String = txtCost.Text

Dim img() As Byte = Nothing

If txtImage.Text.Length <> 0 Then

Dim fs As FileStream = New FileStream(txtImage.Text, FileMode.Open, FileAccess.Read)

Dim br As BinaryReader = New BinaryReader(fs)

img = br.ReadBytes(fs.Length())

End If

'Checking if any of the fields is not filled

If name.Length = 0 Or category.Length = 0 Or runtime.Length = 0 Or rating.Length = 0 Or

description.Length = 0 Or stock.Length = 0 Or price.Length = 0 Or cost.Length = 0 Then

lblError.Content = "Please fill in all the fields!"

Else

'Creating a class that gets as parameters the inputs of the client

Dim newMovie As Movies = New Movies(name, category, releaseDate, runtime, CType(rating, Double),

description, CType(stock, Integer), CType(price, Double), CType(cost, Double), img)

'Checking it the connected with the database and the query was executed successfully

Dim db As DBTransaction = New DBTransaction

If db.addMovie(newMovie) = 1 Then

lblError.Content = ""

lblSuccess.Content = "Insert is successful"

resetFields()

Else

lblError.Content = "Something happened. Please try again later."

End If

End If

End Sub

'Getting an image for the movie

Private Sub btnBrowse\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnBrowse.Click

Dim dlg As Microsoft.Win32.OpenFileDialog = New Microsoft.Win32.OpenFileDialog

dlg.DefaultExt = ".jpg"

dlg.Filter = "Image File (\*.png, \*.jpg)|\*.png; \*.jpg"

Dim result As Boolean = dlg.ShowDialog

If result = True Then

txtImage.Text = dlg.FileName

End If

End Sub

'Movies class from with all the data needed for a movie

Public Class Movies

Private cName As String

Private cCategory As String

Private cReleaseDate As Date

Private cRuntime As String

Private cRating As Double

Private cDescription As String

Private cStock As Integer

Private cPrice As Double

Private cCost As Double

Private cImage() As Byte

Public Sub New(ByVal name As String, ByVal category As String, ByVal releaseDate As Date, ByVal runtime As Integer, ByVal rating As Double,

ByVal description As String, ByVal stock As Integer, ByVal price As Double, ByVal cost As Double, ByVal img() As Byte)

cName = name

cCategory = category

cReleaseDate = releaseDate

cRuntime = runtime

cRating = rating

cDescription = description

cStock = stock

cPrice = price

cCost = cost

cImage = img

End Sub

Public Function getName() As String

Return cName

End Function

Public Function getCategory() As String

Return cCategory

End Function

Public Function getReleaseDate() As Date

Return cReleaseDate

End Function

Public Function getRuntime() As String

Return cRuntime

End Function

Public Function getRating() As Double

Return cRating

End Function

Public Function getDescription() As String

Return cDescription

End Function

Public Function getStock() As String

Return cStock

End Function

Public Function getPrice() As String

Return cPrice

End Function

Public Function getCost() As String

Return cCost

End Function

Public Function getImage() As Byte()

Return cImage

End Function

End Class

'The function used to insert the movie with all its data in the database table that returns 1 if executed successfully or 0 if not

Public Function addMovie(ByVal newMovie As Movies) As Integer

objCommand.CommandText = "INSERT INTO Films(film\_title, film\_categories, film\_release\_date " &

",film\_runtime, film\_rating, film\_description, film\_times\_sold, film\_stock, film\_price, film\_cost, film\_image)" &

"VALUES(@title, @category, @releasedate, @runtime, @rating, @description, @times, @stock, @price, @cost, @image)"

objCommand.Parameters.AddWithValue("@title", newMovie.getName())

objCommand.Parameters.AddWithValue("@category", newMovie.getCategory())

objCommand.Parameters.AddWithValue("@releasedate", newMovie.getReleaseDate())

objCommand.Parameters.AddWithValue("@runtime", newMovie.getRuntime())

objCommand.Parameters.AddWithValue("@rating", newMovie.getRating())

objCommand.Parameters.AddWithValue("@description", newMovie.getDescription())

objCommand.Parameters.AddWithValue("@times", 0)

objCommand.Parameters.AddWithValue("@stock", newMovie.getStock())

objCommand.Parameters.AddWithValue("@price", newMovie.getPrice())

objCommand.Parameters.AddWithValue("@cost", newMovie.getCost())

objCommand.Parameters.AddWithValue("@image", newMovie.getImage())

'Executing the query

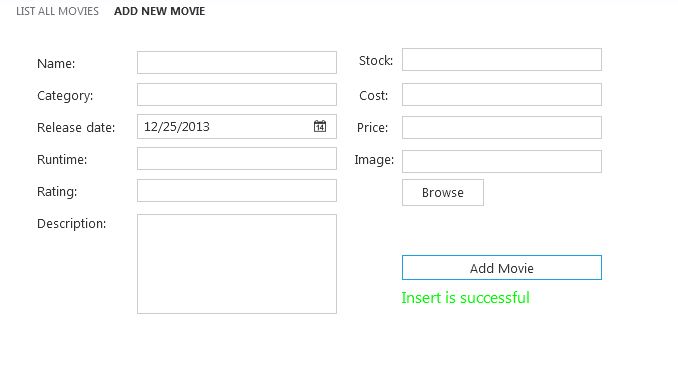
objConnection.Open()

Dim result As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

Return result

End Function

 Fig 8.1 Movie added successfully in the database

**b) Listing a movie**

Class listMovies

'Declaring two type of objects

Private db As DBTransaction = New DBTransaction

Private moviesTable As DataTable

'Getting all the data from the database an showing it on a data grid

Private Sub Grid1\_Loaded(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles Grid1.Loaded

moviesTable = db.getAllMovies()

dgMovies.ItemsSource = moviesTable.DefaultView

End Sub

'The function used for getting the data of movies from the database

Public Function getAllMovies() As DataTable

Dim moviesTable As DataTable = New DataTable()

objAdapter.SelectCommand.CommandText = "SELECT film\_id, film\_title, film\_categories, film\_release\_date, film\_runtime," &

"film\_rating, film\_description, film\_stock, film\_price, film\_cost, film\_image FROM Films"

'Executing the query

objConnection.Open()

objAdapter.Fill(moviesTable)

objConnection.Close()

Return moviesTable

End Function

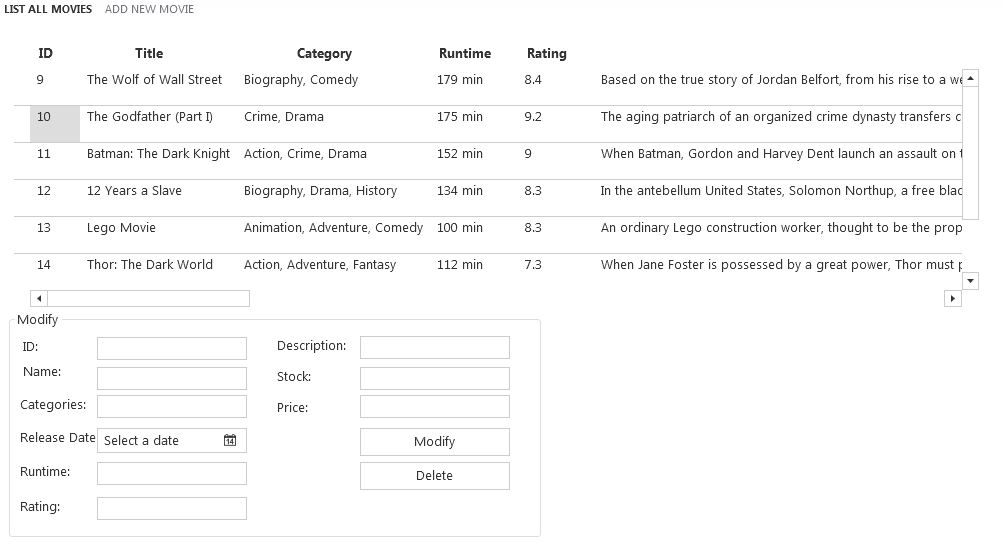


Fig. 8.2 A list of all the movies in the database

**c) Modifying an item in the database**

'Filling all the fields for modification from the datagrid

Private Sub dgMovies\_MouseDoubleClick(sender As System.Object, e As System.Windows.Input.MouseButtonEventArgs) Handles dgMovies.MouseDoubleClick

Dim dataRow As DataRowView = dgMovies.SelectedItem

txtID.Text = dataRow.Item(0).ToString()

txtName.Text = dataRow.Item(1).ToString()

txtCategories.Text = dataRow.Item(2).ToString()

dtpRelease.Text = dataRow.Item(3).ToString()

txtRuntime.Text = dataRow.Item(4).ToString()

txtRating.Text = dataRow.Item(5).ToString()

txtDescripton.Text = dataRow.Item(6).ToString()

txtStock.Text = dataRow.Item(7).ToString()

txtPrice.Text = dataRow.Item(8).ToString()

lblError.Content = ""

lblSuccess.Content = ""

End Sub

'Modifying the movie

Private Sub btnModify\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnModify.Click

If txtID.Text.Length <> 0 Then

Dim id As Integer = CType(txtID.Text, Integer)

Dim name As String = txtName.Text

Dim category As String = txtCategories.Text

Dim releaseDate As Date = dtpRelease.Text

Dim runtime As String = txtRuntime.Text

Dim rating As String = txtRating.Text

Dim description As String = txtDescripton.Text

Dim stock As String = txtStock.Text

Dim price As String = txtPrice.Text

Dim img() As Byte = Nothing

'Checking if all the fields are filled

If name.Length = 0 Or category.Length = 0 Or runtime.Length = 0 Or rating.Length = 0 Or

description.Length = 0 Or stock.Length = 0 Or price.Length = 0 Or txtImage.Text.Length = 0 Then

lblError.Content = "Please fill in all the fields!"

lblSuccess.Content = ""

Else

Dim fs As FileStream = New FileStream(txtImage.Text, FileMode.Open, FileAccess.Read)

Dim br As BinaryReader = New BinaryReader(fs)

img = br.ReadBytes(fs.Length())

'Declaring a variable as integer that results if the query executed successfully

Dim rs As Integer = db.editMovie(id, name, category, releaseDate, runtime, CType(rating, Double), description,

CType(stock, Double), CType(price, Double), img)

'Checking if the movie is modified successfully

If rs = 1 Then

lblSuccess.Content = "Movie has been modified successfuly!"

lblError.Content = ""

dgMovies.ItemsSource = db.getAllMovies().DefaultView

clearFields()

Else

lblError.Content = "Something wrong has happened. Please try again later!"

lblSuccess.Content = ""

End If

End If

Else

lblError.Content = "Please select a movie from the list."

lblSuccess.Content = ""

End If

End Sub

'Function used for modifying a movie that returns as a result 1 if executed successfully or 0 if not

Public Function editMovie(ByVal id As Integer, ByVal name As String, ByVal category As String, ByVal releaseDate As Date,

ByVal runtime As String, ByVal rating As Double, ByVal description As String,

ByVal stock As Integer, ByVal price As Double, ByVal img() As Byte)

objCommand.CommandText = "UPDATE Films SET film\_title = @title, film\_categories = @category, film\_release\_date = @releasedate," &

"film\_runtime = @runtime, film\_rating = @rating, film\_description = @description, film\_stock = @stock," &

"film\_price = @price, film\_image=@img WHERE film\_id = @id"

objCommand.Parameters.AddWithValue("@title", name)

objCommand.Parameters.AddWithValue("@category", category)

objCommand.Parameters.AddWithValue("@releasedate", releaseDate)

objCommand.Parameters.AddWithValue("@runtime", runtime)

objCommand.Parameters.AddWithValue("@rating", rating)

objCommand.Parameters.AddWithValue("@description", description)

objCommand.Parameters.AddWithValue("@stock", stock)

objCommand.Parameters.AddWithValue("@price", price)

objCommand.Parameters.AddWithValue("@id", id)

objCommand.Parameters.AddWithValue("@img", img)

'Executing the query

objConnection.Open()

Dim result As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Return result

End Function

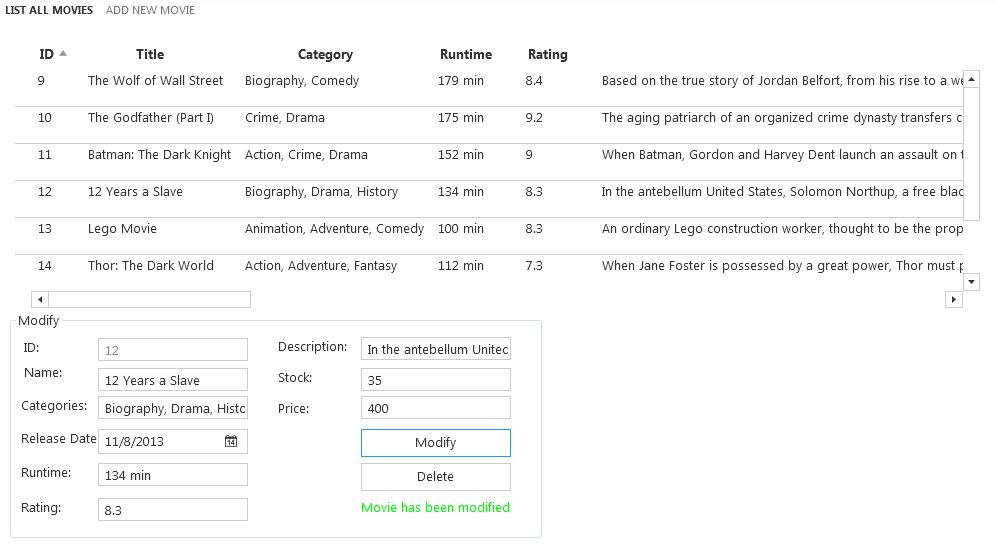


Fig. 8.3 A screenshot that indicates that the movie has been modified successfully using the code above.

**d) Deleting a movie**

To select a movie from the data grid is used the same code as above. What changes is the code for the ‘delete’ button.

'Deleting a movie

Private Sub btnDelete\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnDelete.Click

If txtID.Text.Length <> 0 Then

Dim id As Integer = CType(txtID.Text, Integer)

Dim result As Integer = MessageBox.Show("Do you really want do delete this movie?", "Delete movie", MessageBoxButton.YesNo)

'Checking if the movie deleted successfully

If result = vbYes Then

Dim rs As Integer = db.deleteMovie(id)

If rs = 1 Then

lblSuccess.Content = "Movie has beed deleted successfully."

lblError.Content = ""

dgMovies.ItemsSource = db.getAllMovies().DefaultView

Else

lblError.Content = "Something wrong has happened. Please try again later."

lblSuccess.Content = ""

End If

End If

Else

lblError.Content = "Please select a movie from the list."

lblSuccess.Content = ""

End If

End Sub

'Function that's used for deleting a movie and returning a result = 1 if deleted successfully or 0 if not

Public Function deleteMovie(ByVal id As Integer) As Integer

objCommand.CommandText = "DELETE FROM Films WHERE film\_id = @id"

objCommand.Parameters.AddWithValue("@id", id)

'Executing the query

objConnection.Open()

Dim rs As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Return rs

End Function

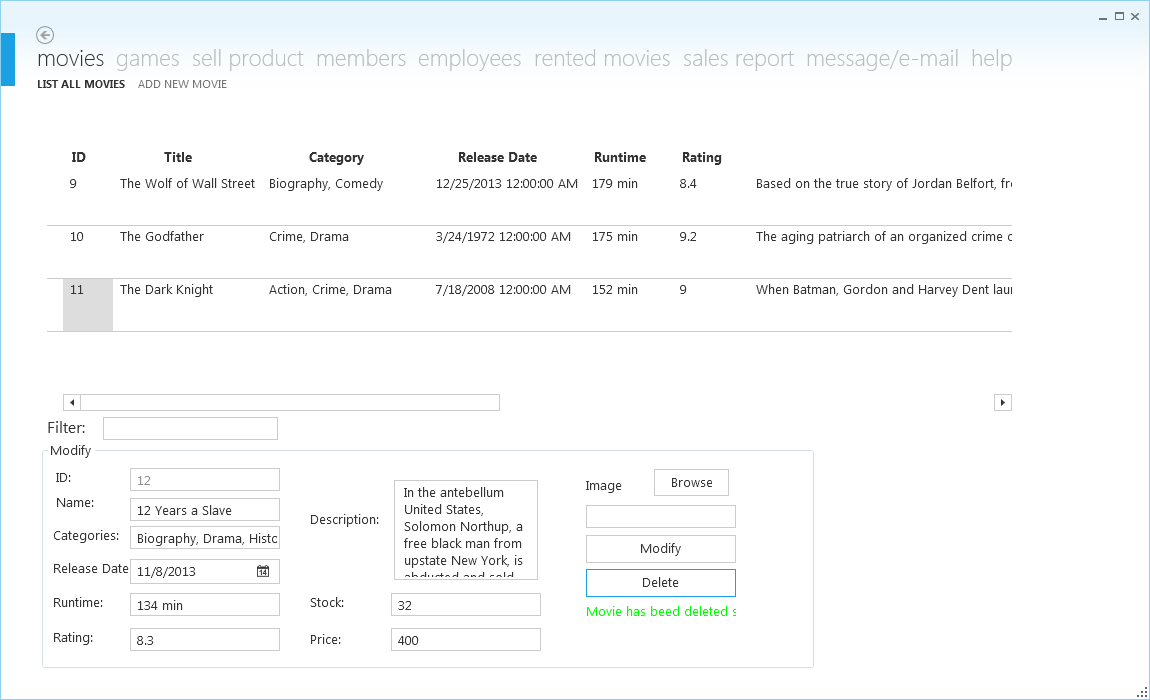


Fig. 8.4 A screenshot showing that the movie has been deleted successfully with the code above.

The logic of code for adding, modifying, listing and deleting a movie is the same for movies, games, members and employees.

**e) Authentication security**

1. Logging in as an administrator

Class AdministratorLogin

Dim objConnection As SqlConnection

Dim objCommand As SqlCommand

Dim objDataAdapter As SqlDataAdapter

Dim objDataSet As DataSet

'Logging in as an administrator

Private Sub btnLogin\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnLogin.Click

'Getting the username and password as input

Dim username As String = txtUsername.Text

Dim password As String = txtPassword.Password

'Checking if the fields are filled

If username.Length = 0 Or password.Length = 0 Then

lblError.Content = "Please fill in all the fields."

Else

'Connecting with the database

Dim connString As String = Utility.GetConnectionString()

objConnection = New SqlConnection("server=localhost;database=CambridgeProject;Integrated Security=SSPI")

objCommand.Connection = objConnection

'Checking if the username and password coincide with a row in the database table for administrator

objCommand.Parameters.Clear()

objCommand.CommandText = "SELECT \* FROM administrator WHERE admin\_username=@username AND admin\_password=@password"

objCommand.Parameters.AddWithValue("@username", username)

objCommand.Parameters.AddWithValue("@password", password)

'Executing the query

objConnection.Open()

objDataAdapter = New SqlDataAdapter()

objDataAdapter.SelectCommand = objCommand

objDataSet = New DataSet()

objDataAdapter.Fill(objDataSet, "admin")

objConnection.Close()

Dim result As Integer = objDataSet.Tables(0).Rows.Count

If result = 0 Then

lblError.Content = "Invalid credentials"

Else

'Opening a new window with all the functionalities of the administrator

Dim newWindow As New MainWindow(0)

newWindow.Width = 1150

newWindow.Height = 700

'Creating menus for the administrator in the Modern UI theme

Dim mainMenu(9) As FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(1) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(1).DisplayName = "Movies"

mainMenu(2) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(2).DisplayName = "Games"

mainMenu(3) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(3).DisplayName = "Sell Product"

mainMenu(4) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(4).DisplayName = "Members"

mainMenu(5) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(5).DisplayName = "Employees"

mainMenu(6) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(6).DisplayName = "Rented Movies"

mainMenu(7) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(7).DisplayName = "Sales Report"

mainMenu(8) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(8).DisplayName = "Message/E-mail"

mainMenu(9) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(9).DisplayName = "Help"

newWindow.MenuLinkGroups.Add(mainMenu(1))

newWindow.MenuLinkGroups.Add(mainMenu(2))

newWindow.MenuLinkGroups.Add(mainMenu(3))

newWindow.MenuLinkGroups.Add(mainMenu(4))

newWindow.MenuLinkGroups.Add(mainMenu(5))

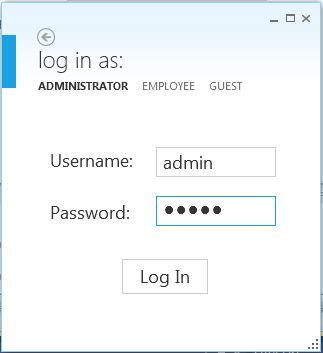
newWindow.MenuLinkGroups.Add(mainMenu(6))

newWindow.MenuLinkGroups.Add(mainMenu(7))

newWindow.MenuLinkGroups.Add(mainMenu(8))

newWindow.MenuLinkGroups.Add(mainMenu(9))

newWindow.Show()

Fig. 8.5 Logging in as an administrator

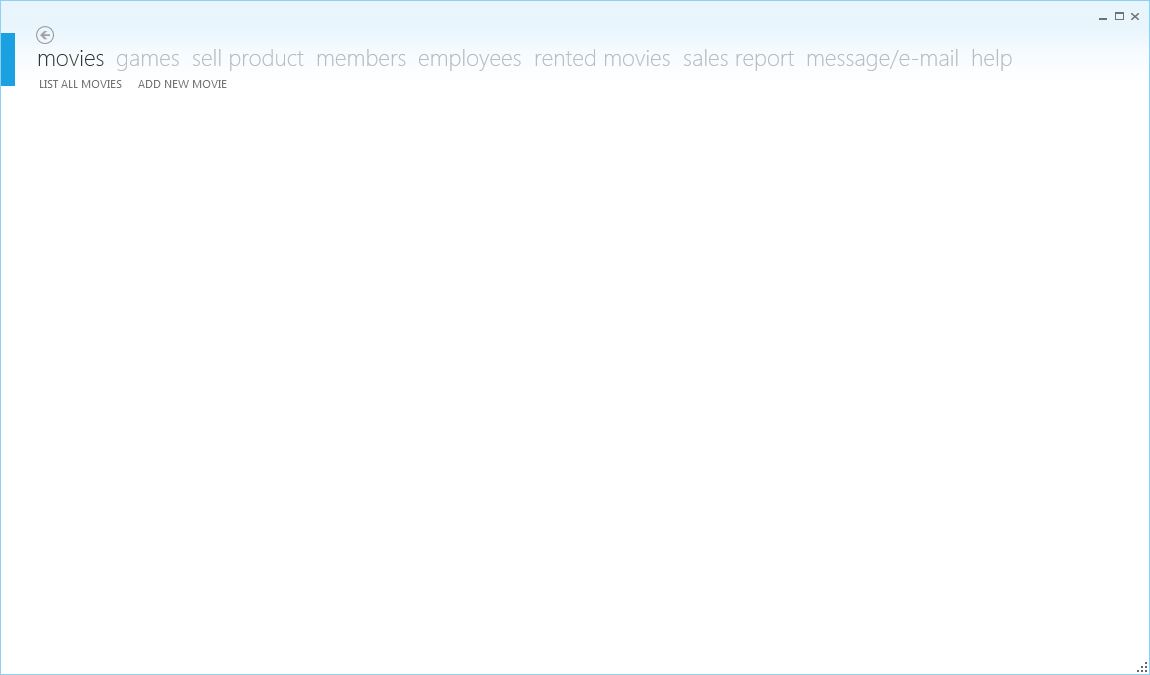


Fig. 8.6 The window opened for the administrator

2. Logging in as an employee

Class EmployeeLogin

Dim objConnection As SqlConnection

Dim objCommand As SqlCommand

Dim objDataAdapter As SqlDataAdapter

Dim objDataSet As DataSet

Dim db As DBTransaction

'Logging in as an employee

Private Sub btnLogin\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnLogin.Click

'Getting the username and the password from the employee as an input

Dim username As String = txtUsername.Text

Dim password As String = txtPassword.Password

'Checking if all the fields are filled

If username.Length = 0 Or password.Length = 0 Then

lblError.Content = "Please fill in all the fields."

Else

'Connecting with the database

Dim connString As String = Utility.GetConnectionString()

objConnection = New SqlConnection("server=localhost;database=CambridgeProject;Integrated Security=SSPI")

'Checking if the username and password coincide with a row in the database table for administrator

objCommand = New SqlCommand()

objCommand.Connection = objConnection

objCommand.CommandType = CommandType.Text

objCommand.CommandText = "SELECT \* FROM Employees WHERE employee\_username=@username AND employee\_password=@password"

objCommand.Parameters.AddWithValue("@username", username)

objCommand.Parameters.AddWithValue("@password", password)

'Executing the query

objConnection.Open()

objDataAdapter = New SqlDataAdapter()

objDataAdapter.SelectCommand = objCommand

objDataSet = New DataSet()

objDataAdapter.Fill(objDataSet, "employee")

objConnection.Close()

Dim result As Integer = objDataSet.Tables(0).Rows.Count

If result = 0 Then

lblError.Content = "Invalid credentials"

Else

'Getting the id of the employee logged in and opening a new window with all the functionialites of an employee

Dim dr As DataRow = objDataSet.Tables(0).Rows(0)

Dim id As Integer = dr.Item(0)

Dim newWindow As New MainWindow(id)

newWindow.Width = 1050

newWindow.Height = 700

'Creating menus fpr the employee using Moder UI theme

Dim mainMenu(8) As FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(1) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(1).DisplayName = "Movies"

mainMenu(2) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(2).DisplayName = "Games"

mainMenu(3) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(3).DisplayName = "Sell Product"

mainMenu(4) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(4).DisplayName = "Members"

mainMenu(5) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(5).DisplayName = "Rented Movies"

mainMenu(6) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(6).DisplayName = "My Sales"

mainMenu(7) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(7).DisplayName = "Message/E-Mail"

mainMenu(8) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(8).DisplayName = "Help"

newWindow.MenuLinkGroups.Add(mainMenu(1))

newWindow.MenuLinkGroups.Add(mainMenu(2))

newWindow.MenuLinkGroups.Add(mainMenu(3))

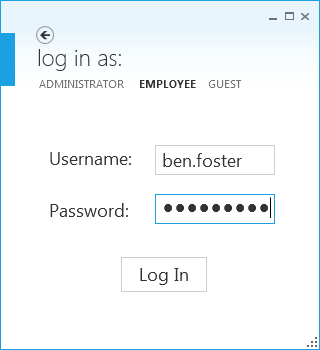
newWindow.MenuLinkGroups.Add(mainMenu(4))

newWindow.MenuLinkGroups.Add(mainMenu(5))

newWindow.MenuLinkGroups.Add(mainMenu(6))

newWindow.MenuLinkGroups.Add(mainMenu(7))

newWindow.Show()

Fig. 8.7 Logging in as an employee.

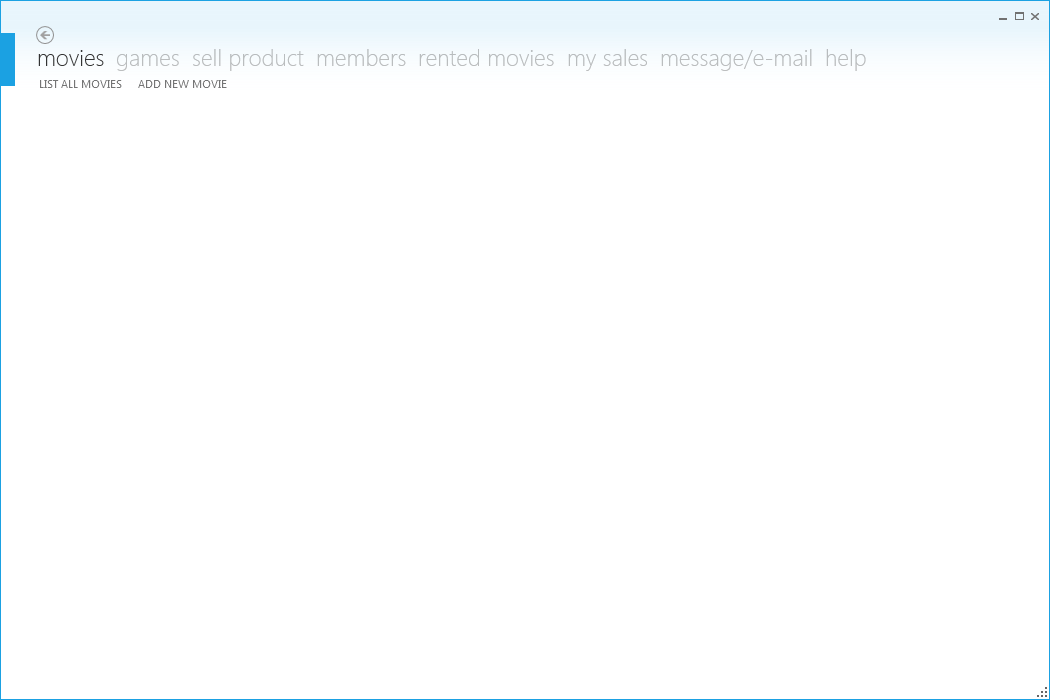


Fig. 8.8 Employee panel after logging in with the code above.

3. Logging in as a guest

Class GuestLogin

'Logging in as a guest

Private Sub btnGuest\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnGuest.Click

Dim newWindow As New MainWindow(-1)

newWindow.Height = 600

newWindow.Width = 750

'Getting all the menus for the guest panel through Modern UI

Dim mainMenu(4) As FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(1) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(1).DisplayName = "Movies"

mainMenu(2) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(2).DisplayName = "Games"

mainMenu(3) = New FirstFloor.ModernUI.Presentation.LinkGroup

mainMenu(3).DisplayName = "Log out"

newWindow.MenuLinkGroups.Add(mainMenu(1))

newWindow.MenuLinkGroups.Add(mainMenu(2))

newWindow.MenuLinkGroups.Add(mainMenu(3))

newWindow.Show()

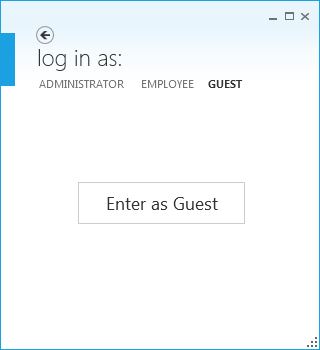


Fig. 8.9 The logging in panel as a guest (no obstacle in logging in)

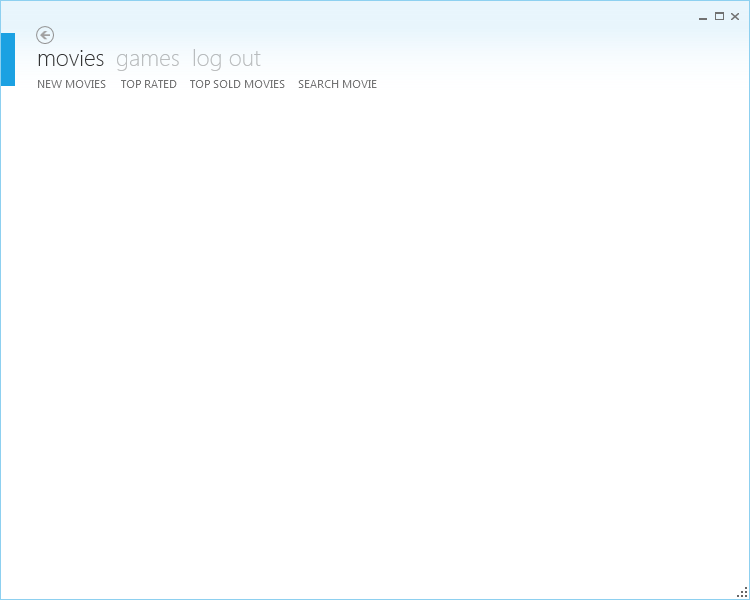


Fig. 8.10 Guest panel after logging in as a guest using the code above.

**f) Selling a product**

'Selling a movie

Private Sub btnSellMovie\_Click(sender As System.Object, e As System.Windows.RoutedEventArgs) Handles btnSellMovie.Click

'Getting the necessary data from movie for the selling process

If txtFilmId.Text.Length <> 0 And CType(txtStock.Text.ToString, Integer) <> 0 Then

Dim filmId As Integer = CType(txtFilmId.Text, Integer)

Dim stock As Integer = CType(txtStock.Text, Integer)

Dim price As Double = CType(txtPrice.Text, Integer)

Dim cost As Double = CType(txtCost.Text, Integer)

Dim memberId As Integer = 0

'Checking if movie is going to be sold to a member

If txtMemberId.Text.Length <> 0 Then

memberId = CType(txtMemberId.Text, Integer)

End If

'Checking if the movie sold completely

If db.sellMovie(MainWindow.getId(), filmId, Date.Now, price, cost, memberId) = 3 And db.decrementFilmStock(filmId) = 1 Then

lblSuccess.Content = "Sold"

lblError2.Content = ""

clearFields()

dgMovies.ItemsSource = db.getAllMovies.DefaultView

dgMembers.ItemsSource = db.getAllMembers.DefaultView

Else

lblSuccess.Content = ""

lblError2.Content = "Wrong"

End If

Else

lblError2.Content = "No stock"

End If

End Sub

'function used to sell a movie

Public Function sellMovie(ByVal employeeId As Integer, ByVal movieId As Integer, ByVal movieSellDate As Date, ByVal moviePrice As Double,

ByVal movieCost As Double, ByVal memberId As Integer) As Integer

objCommand.CommandText = "INSERT INTO Film\_sales(employee\_sold\_id, film\_sold\_id, film\_sale\_date, film\_price, film\_cost, member\_bought\_id)" &

"VALUES (@employeeId, @movieId, @movieSellDate, @moviePrice, @movieCost, @memberId)"

objCommand.Parameters.AddWithValue("@employeeId", employeeId)

objCommand.Parameters.AddWithValue("@movieId", movieId)

objCommand.Parameters.AddWithValue("@movieSellDate", movieSellDate)

objCommand.Parameters.AddWithValue("@moviePrice", moviePrice)

objCommand.Parameters.AddWithValue("@movieCost", movieCost)

objCommand.Parameters.AddWithValue("@memberId", memberId)

'Execute query

objConnection.Open()

Dim rs As Integer = objCommand.ExecuteNonQuery()

objCommand.Parameters.Clear()

Dim rs1 As Integer = 1

Dim rs2 As Integer = 1

'Check if it's sold by an employee

If employeeId <> 0 Then

objCommand.CommandText = "UPDATE Employees SET employee\_films\_sold = employee\_films\_sold + 1 WHERE employee\_id = @employeeId"

objCommand.Parameters.AddWithValue("@employeeId", employeeId)

rs2 = objCommand.ExecuteNonQuery()

objCommand.Parameters.Clear()

'Checking for bonus of the employee

Dim bonus As Double = checkBonus(employeeId)

objCommand.CommandText = "UPDATE employees SET employee\_bonus = @bonus WHERE employee\_id=@eid"

objCommand.Parameters.AddWithValue("@bonus", bonus)

objCommand.Parameters.AddWithValue("@eid", employeeId)

'Executing query

objCommand.ExecuteNonQuery()

objCommand.Parameters.Clear()

End If

'Add to the member one more movie bought

If memberId <> 0 Then

objCommand.CommandText = "UPDATE members SET member\_films\_bought = member\_films\_bought + 1 WHERE member\_id = @memberId"

objCommand.Parameters.AddWithValue("@memberId", memberId)

rs1 = objCommand.ExecuteNonQuery()

End If

objConnection.Close()

objCommand.Parameters.Clear()

'Add one to the number of times this film is sold

objCommand.CommandText = "UPDATE films SET film\_times\_sold = film\_times\_sold + 1 WHERE film\_id=@id"

objCommand.Parameters.AddWithValue("@id", movieId)

objConnection.Open()

objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Return rs + rs1 + rs2

End Function

'Function that decrements 1 from the stock of the movie when it's sold

Public Function decrementFilmStock(ByVal movieId As Integer) As Integer

objCommand.CommandText = "UPDATE Films SET film\_stock = film\_stock - 1 WHERE film\_id = @movieId"

objCommand.Parameters.AddWithValue("@movieId", movieId)

'Executing the query

objConnection.Open()

Dim rs As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Return rs

End Function

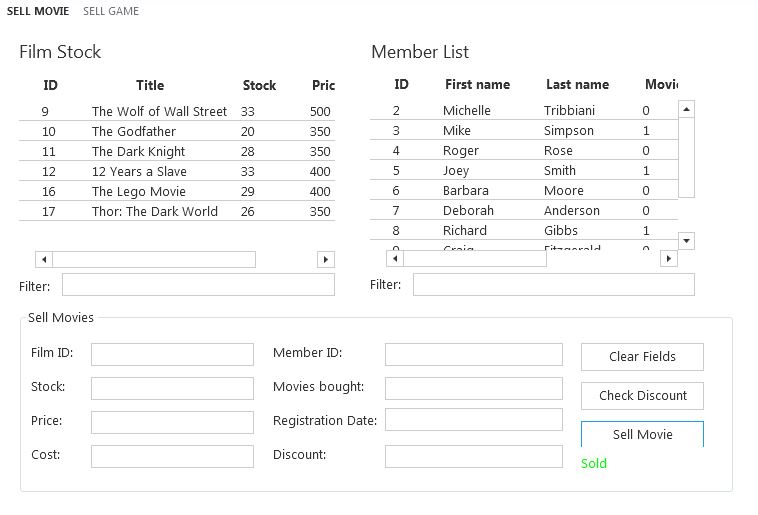


Fig. 8.11 A screenshot showing that the above code is executed successfully.

**g) Rental process**

'Renting a movie to a member

Private Sub btnRent\_Click(sender As Object, e As RoutedEventArgs) Handles btnRent.Click

'Checking if member and movie are selected

If txtMemberID.Text.Length <> 0 And txtFilmId.Text.Length <> 0 Then

If CType(txtRented.Text, Integer) < 2 Then

If CType(txtFilmStock.Text.ToString, Integer) <> 0 Then

Dim fID As Integer = CType(txtFilmId.Text, Integer)

Dim mID As Integer = CType(txtMemberID.Text, Integer)

Dim price As Double = CType(txtPrice.Text, Double)

Dim rs As Integer = db.rentMovie(fID, mID, price)

Dim rs1 As Integer = db.decrementFilmStock(fID)

'Checking if query is executed successfully

If rs = 1 And rs1 = 1 Then

dTableMovies = db.getAllMovies()

dgMovies.ItemsSource = dTableMovies.DefaultView

lblSuccess.Content = "OK"

clearFields()

Else

lblError.Content = "Try again later"

End If

Else

lblError.Content = "No Stock"

End If

Else

lblError.Content = "Limit rents"

End If

Else

lblError.Content = "Film & Member?"

End If

End Sub

'The function called for renting a movie

'The function inserts information to a table in database that keeps track of rents

Public Function rentMovie(ByVal fID As Integer, ByVal mID As Integer, ByVal price As Double) As Integer

objCommand.CommandText = "INSERT INTO film\_rent(film\_rent\_id, member\_rent\_id, rent\_date, return\_date, film\_rent\_price, rent\_status) " &

"VALUES(@fid, @mid, @rd, @red, @pr, @st)"

objCommand.Parameters.AddWithValue("@fid", fID)

objCommand.Parameters.AddWithValue("@mid", mID)

objCommand.Parameters.AddWithValue("@rd", Date.Now())

objCommand.Parameters.AddWithValue("@red", Date.Now.AddDays(3))

objCommand.Parameters.AddWithValue("@pr", price)

objCommand.Parameters.AddWithValue("@st", True)

objConnection.Open()

'Executing query

Dim rs As Integer = objCommand.ExecuteNonQuery()

objConnection.Close()

objCommand.Parameters.Clear()

Return rs

End Function

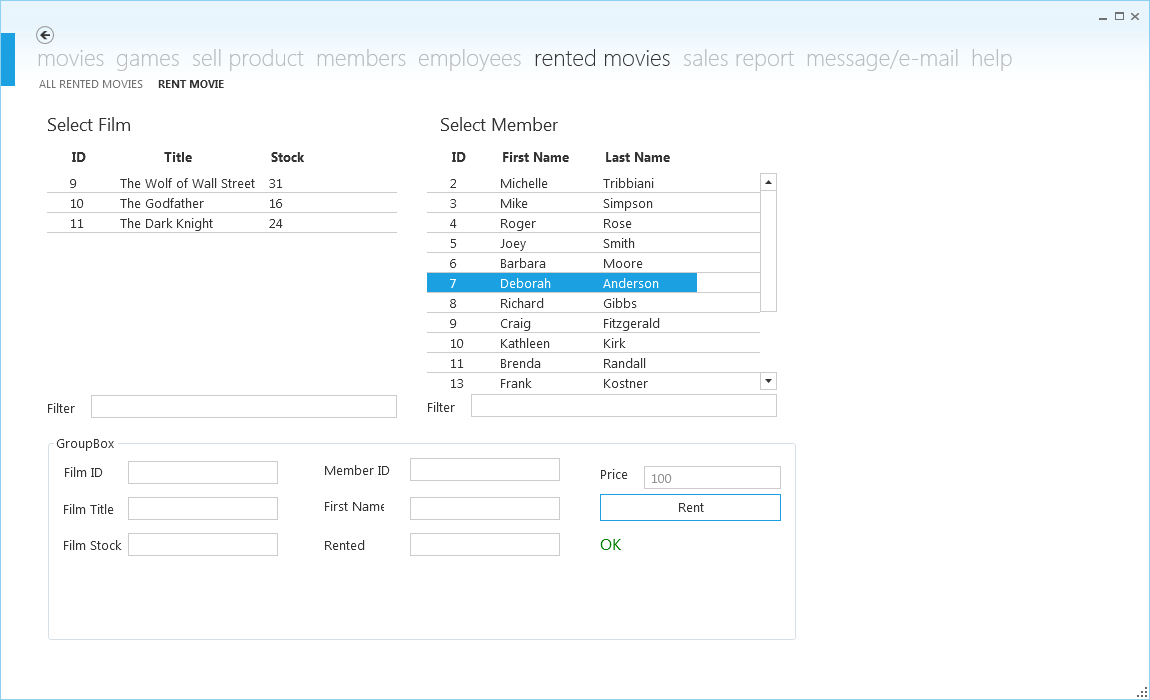


Fig. 8.12 A screenshot showing that using the above code the movie is rented to a member.

**ii) Beta testing**

After alpha testing has been completed successfully, the next step is beta testing.

The program created is given to the client for a tryout for one week. The reason for this is not to be put to work totally, but just for the client to give a try and to find any glitches.

After the program was tested for a week we received the following feedback from the client:

* Some of the labels were not shown completely
* Some text boxes didn’t show all the writing in it.
* The images in guest panel were not the size they should be.
* The program couldn’t send SMS to a member
* The need for some on – screen help when completing the form – based interface when adding a movie

The program before and after the change for the above feedback is shown in the figures below:

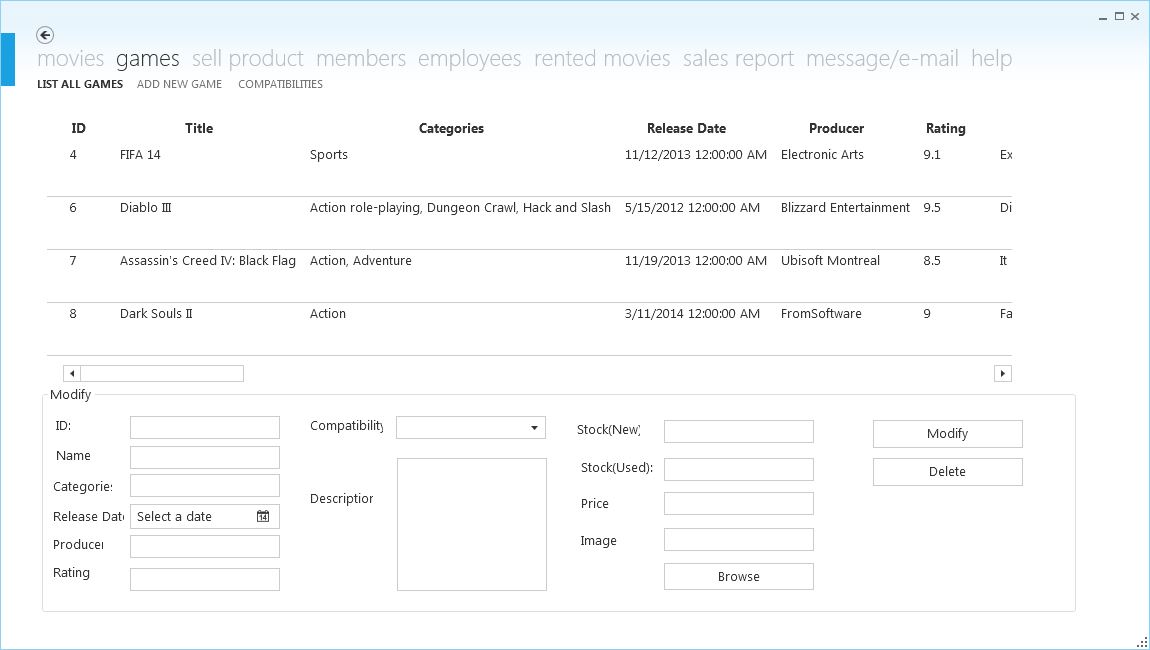


Fig. 8.13 Labels before beta testing

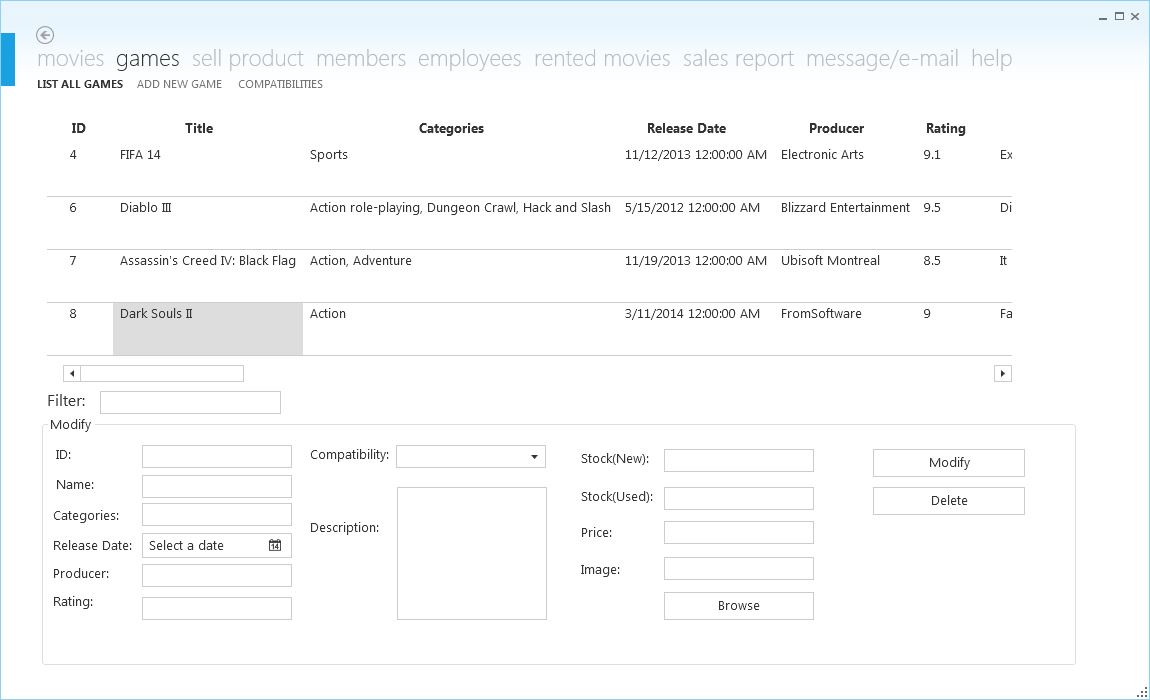


Fig. 8.14 Labels after beta testing



Fig. 8.14 Games panel before beta testing

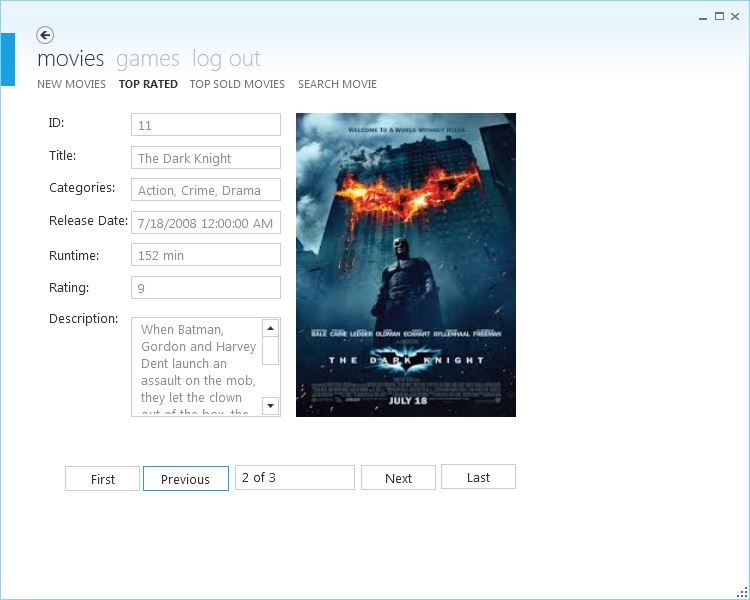


Fig. 8.15 Games panel after beta testing

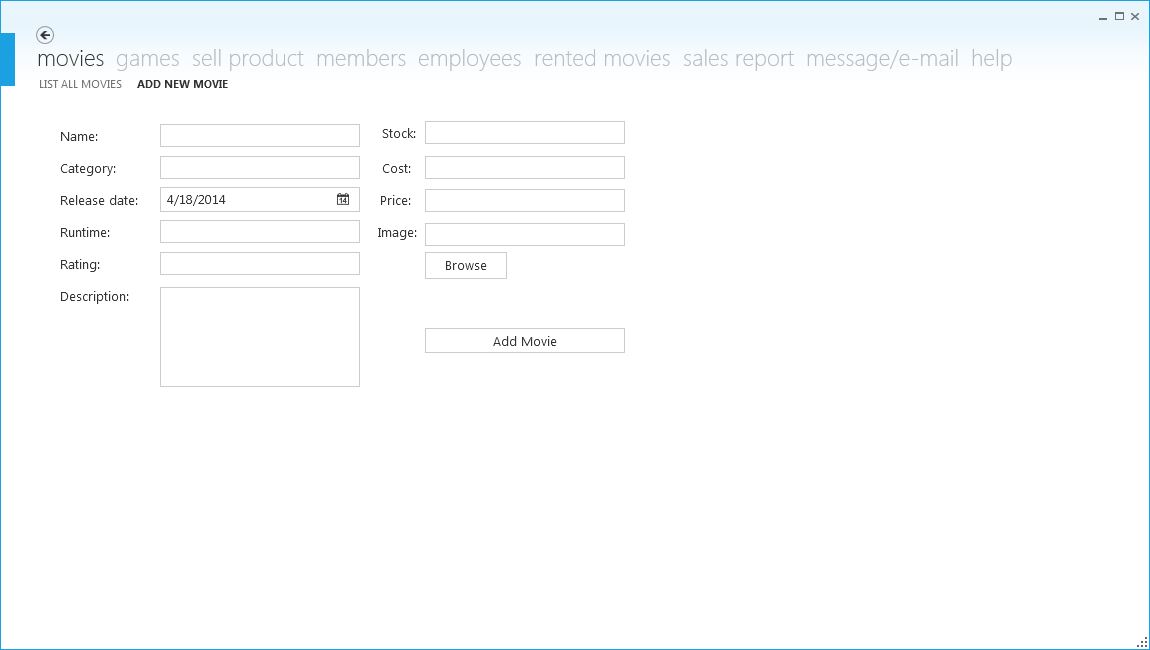


Fig. 8.16 Adding a movie before beta testing

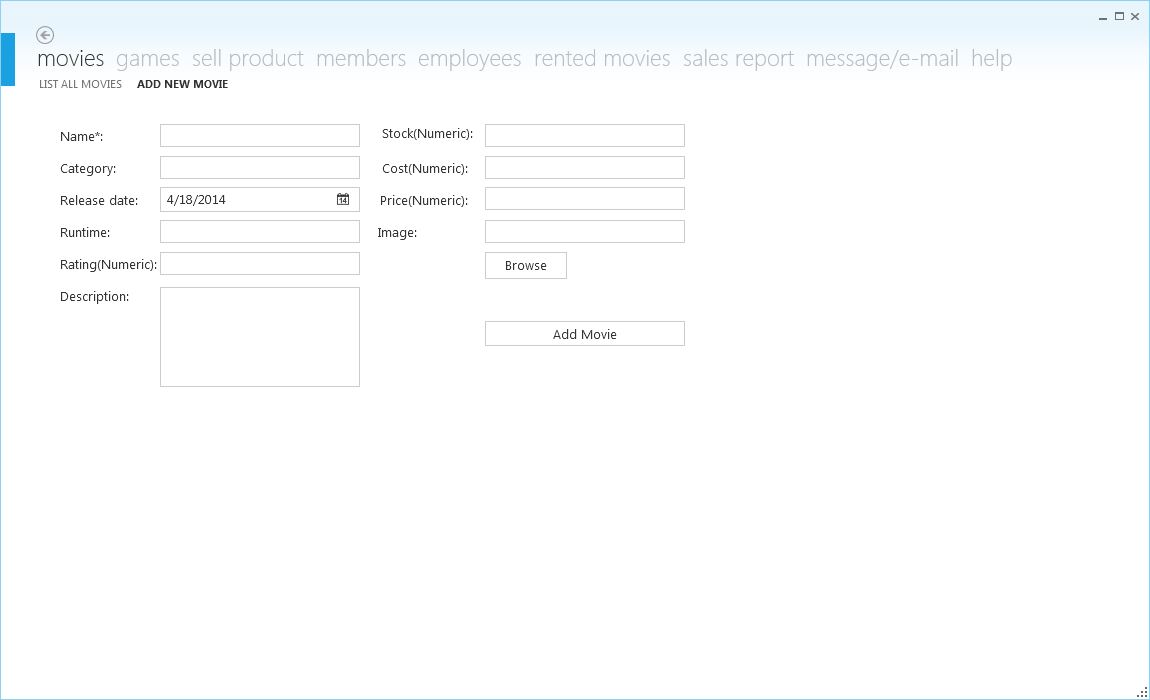


Fig. 8.17 Adding a movie after beta testing

**9. Installation**

**i) Planning the installation**

After the beta testing the program was improved even more. Now it’s ready to be installed in the store. The following steps will be followed for this process

* Configuring the terminal computer to have a fixed IP address in order to point to the SQL Server
* Installing database in the receptionist computer.
* Configuring the connection string in terminal in order to point to the above IP.
* Running the setup in both computers.
* The first time it runs it needs to be run from administrator panel in order to create employees to fill the data.
* After entering some data it is now fully operable even from the guest account
* From that moment the software will be ready for execution, filled with data and used normally.

**ii) Training the staff**

The staff is already trained to use this application software. The only things needed are some basic computer skills and knowledge, which the staff already posses. A user guide will be available for all the users of this program for a more detailed specification of how this program is used.

**iii) Strategy for implementation**

The application software has been completed and is now ready for implementation. After contacting my client, he has already purchased a new computer as a terminal for the client and the client relation employee. We’re thinking of using parallel running as a strategy for implementation.

Even though the program passed its tests, it may still have some bugs. The client will use the old system and the new system until the new system is considered fault free. Meanwhile he will also put all the movie, games, members and employees to the new system.

The program will be installed according to the installation plan presented above.

We’ve proposed this way of implementing the software to our client and he has agreed doing so. Now the software is ready for implementation.

**10. System maintenance documentation**

**i) Hardware and software specifications**

In order to work properly with the program as long as it concerns to its requirements the following software and hardware are required:

**a) Software requirements:**

1. .NET Framework 3 (or later version) – the platform which allows the execution of the application

2. Microsoft SQL Server 2008 or later – RDBMS (Relational Database Management System) stores the data of the application in table format which are related to each other

3. Microsoft Windows Vista/7/8 – Operating system that serves as a platform for the execution of the application

**b) Hardware requirement (*minimum*):**

1. 2 PCs with:  
 a) Pentium 4; 3.0 GHz (or equivalent AMD processor) – to have a good user experience  
 b) 2 GB RAM DDRII – to have a good user experience  
 c) 500 GB HDD – to store the data   
 d) Network interface card – in order to connect to internet  
 e) Intel HD Graphics 4000 (or any other graphic card for better performance) (it’s optional)

2. 19” Flat Monitor x2 (touch: optional) -

3. Keyboard x2 - to input data

4. Mouse x2 - to navigate

5. Printer - to print hard copies of sales

6. CD-ROM – to install the program

7. Router or switch (wireless optional) – to connect the computers with each other and the internet.

**ii) Technologies used**

For the creation of this program, the following technologies are used:

* Visual Basic .NET (VB.NET) – an object-oriented computer programming language that can be viewed as an evolution of the classic Visual Basic (VB), implemented on the .NET Framework
* Microsoft SQL Server – a relational database management system developed by Microsoft (uses SQL (Standard Query Language) to query the database to add/edit/list/delete data)
* iText – is an open source library for creating and manipulating PDF files in .NET Framework.
* Modern UI – a set of controls and styles converting a WPF application into a great looking Modern UI app.

**iii) Original programming used in the new system**

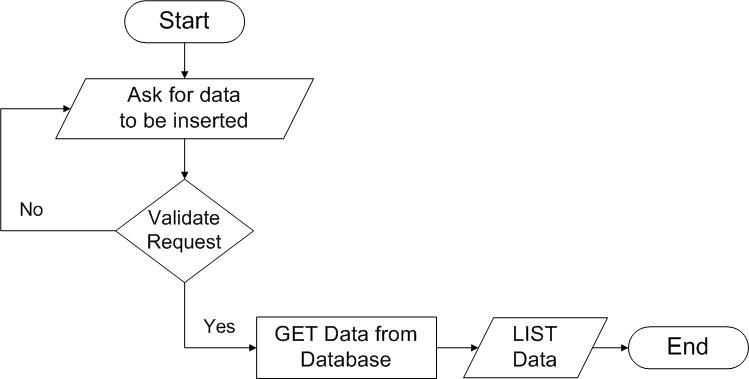
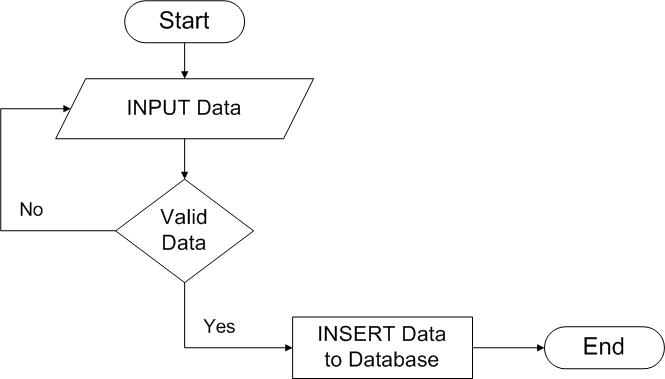


Fig. 10.1 Adding an item in database Fig. 10.2 Listing data as an output

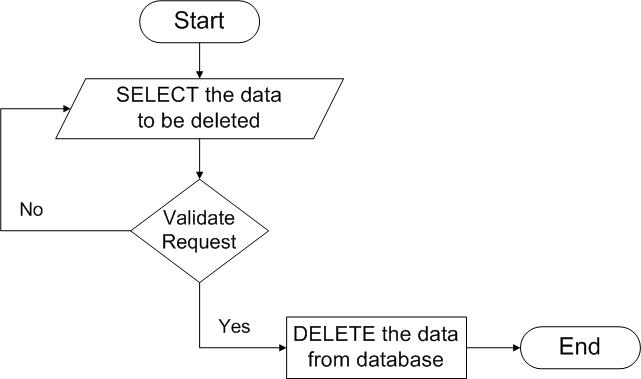
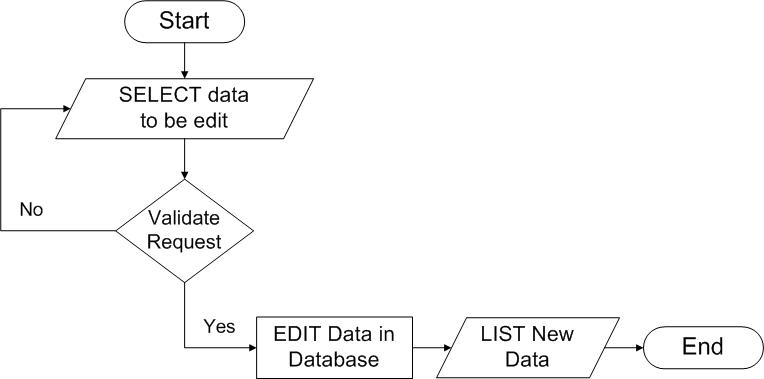


Fig. 10.3 Editing an item Fig. 10.4 Deleting an item

**11. User guide**

**i) Introduction**

This is a application software created for a specific purpose of the business XXYYY of XXXYYY. The main purpose of this software is to some of the procedures on the business.

**ii)Set-up guide**

**a) Logging in**

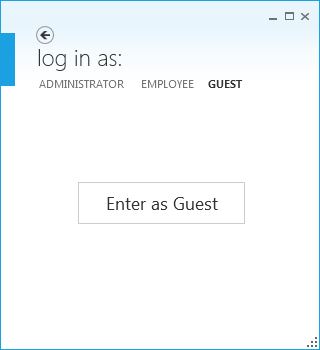
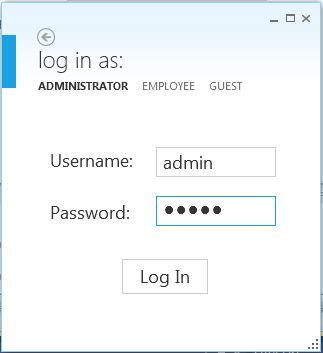


Fig. 11.1 Fig. 11.2 Fig. 11.3

When the program is opened there’s a log-in window showed and you can chose as what type of user you want to log-in. If you chose as an administrator or an employee, you’ll need to enter the correct username and password to proceed to the next window. Otherwise, the program won’t let you continue to the next step.

After logging in successfully this type of window will open depending on what type of user you chose.

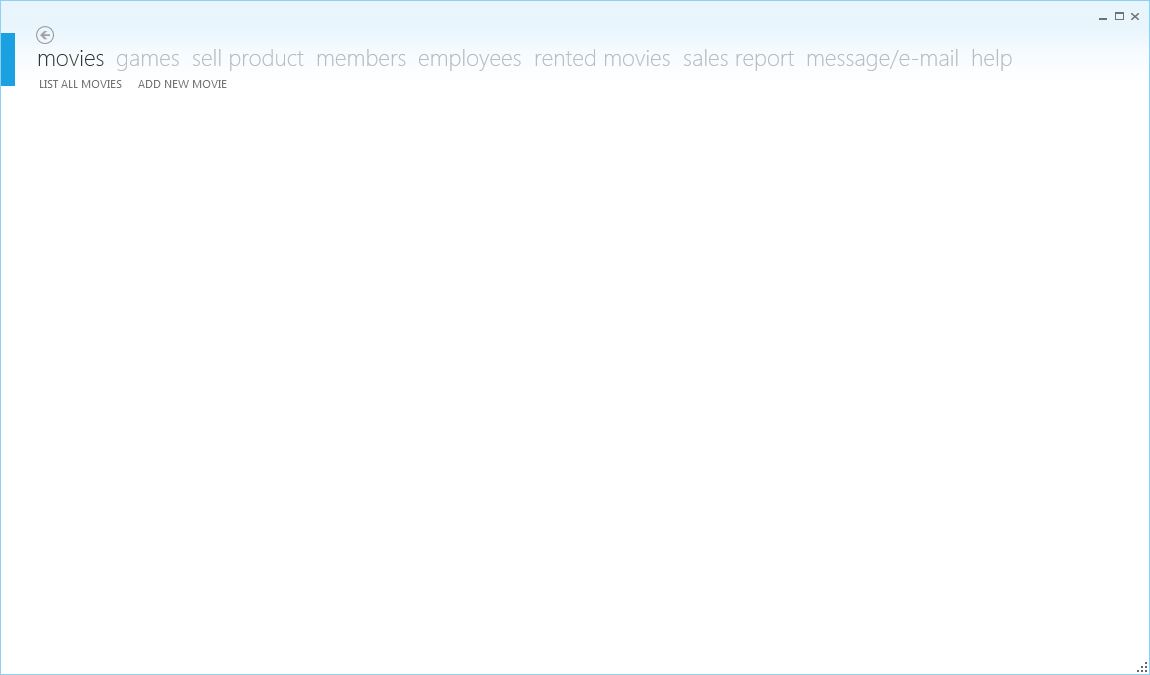


Fig. 11.4 Window shown if logged in as an administrator.

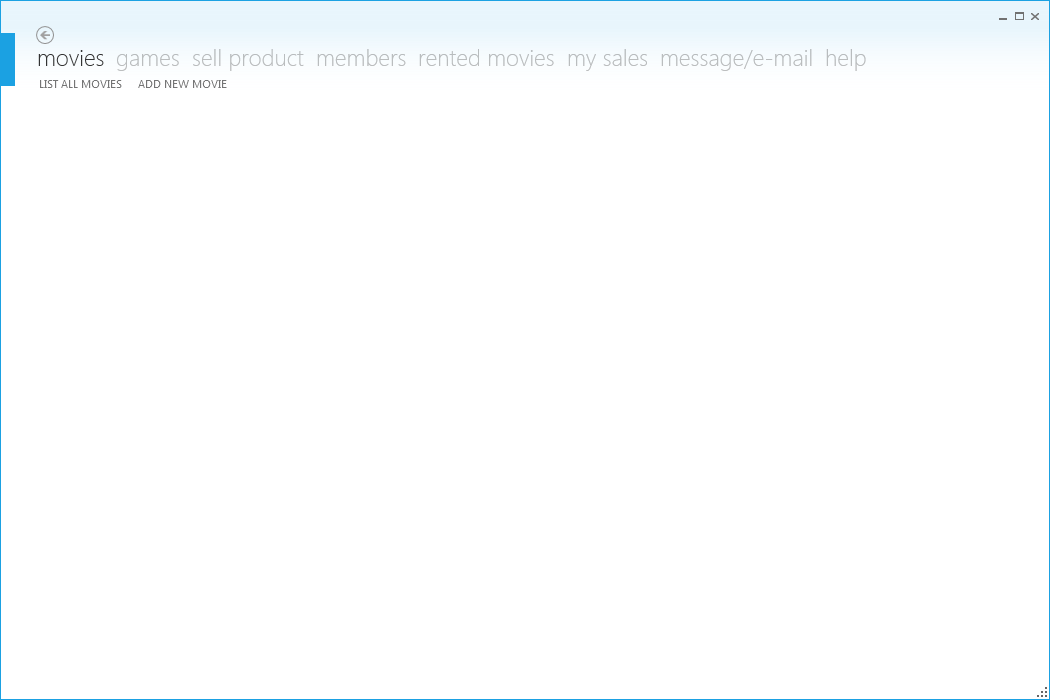


Fig. 11.5 Window shown if logged in as an employee.

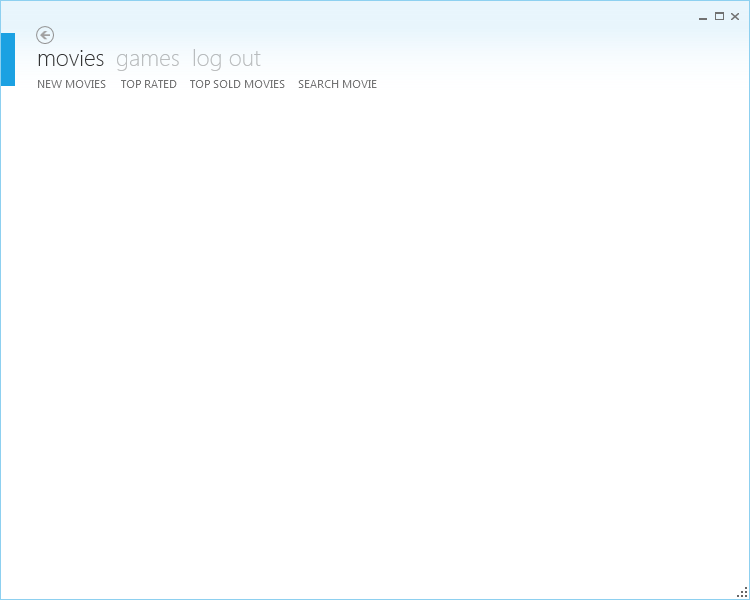


Fig. 11.6 Window shown if logged in as a guest.

To use any option that each of these windows offer you only have to click on one of the menus of the window.

**b) Option 1: Adding something new to the program.**

1. Adding a movie

When logged in as an administrator or an employee you have the right to add a new movie. This can be done my clicking ‘movies’ and then ‘add new movie’ (in the fig. 11.4 and 11.5). After doing so the page will change to this (fig. 11.7):

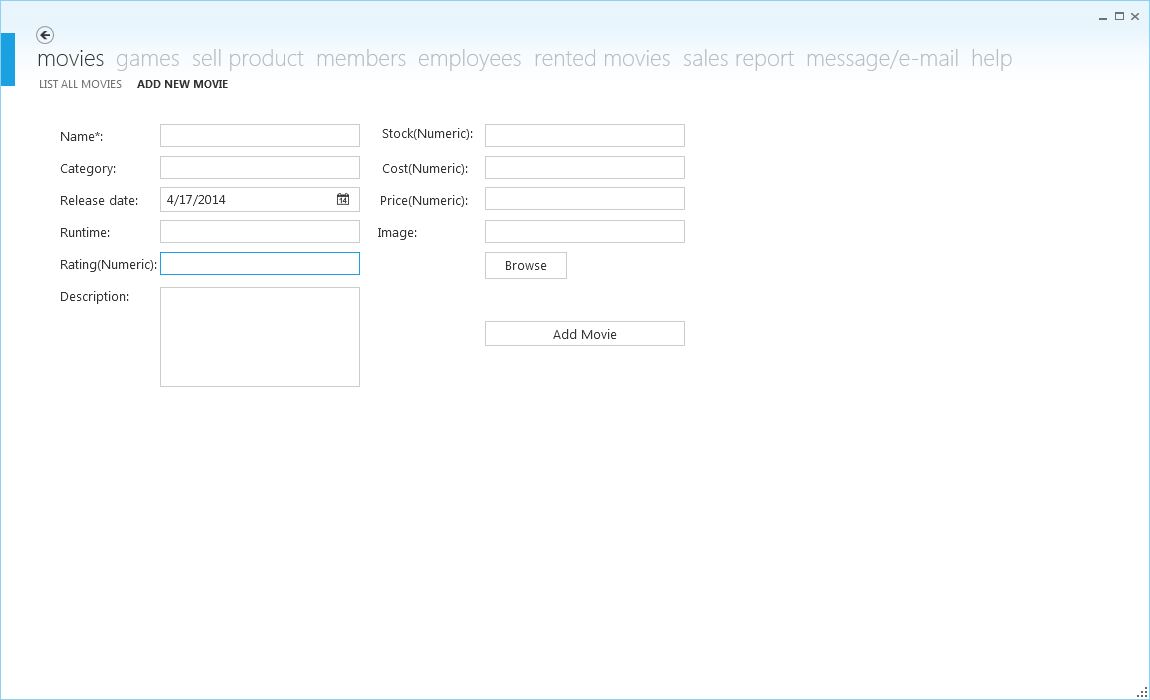


Fig. 11.7 Adding a new movie in a form-based interface for both the administrator and the employee.

To add a movie successfully all of the fields have to be completed. The fields (name, category, runtime and description) can accept all kinds of characters. Fields such as (rating, cost, and price) can accept any kind of number. Field ‘stock’ can accept only integers. For the ‘release date’ field you can chose any date from the calendar by clicking on the icon on the right of the field or you can write the date directly on the text box. To add an image to the movie you have to click the ‘browse’ button and find an image on any folder location on your computer. After completing all the fields, press the ‘add movie’ button and the movie will be inserted successfully. An example of completing this form is showed in the fig. 11.8 below.

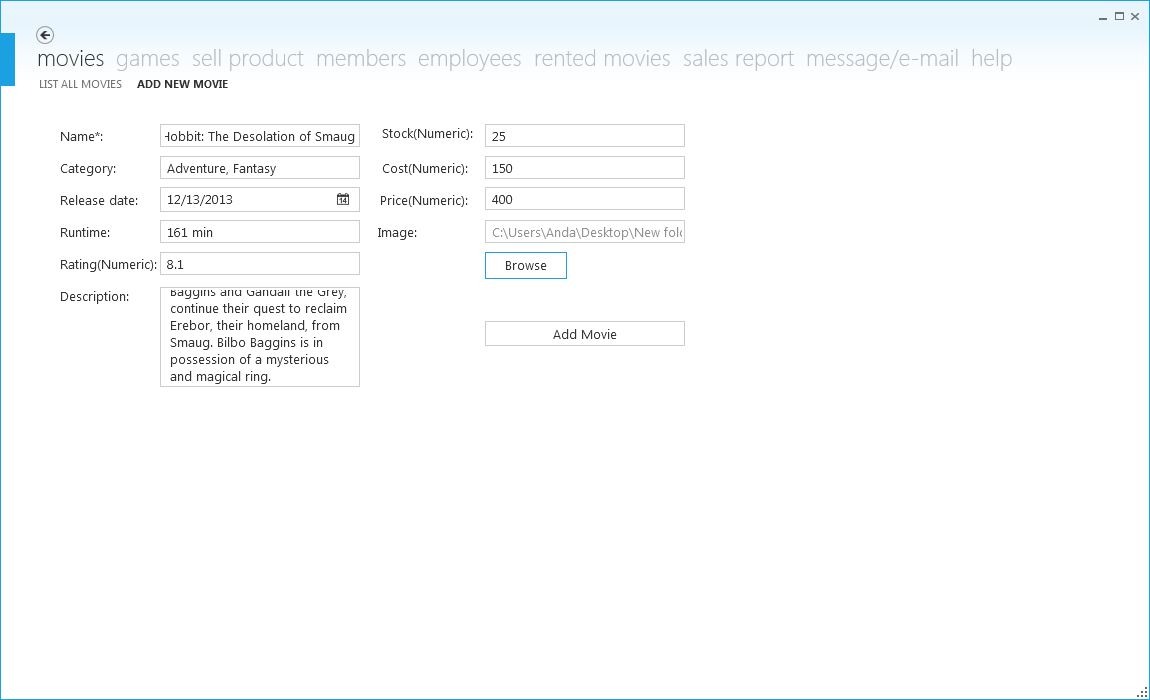


Fig. 11.8 An example of a completed form of adding a new movie

2. Adding a game

When logged in as an administrator or an employee you also have the right to add a new game. This can be done my clicking ‘games’ and then ‘add new game’ (in the fig. 11.4 and 11.5). After doing so the page will change like in the fig. 11.9.

Here at games this process is a little bit more complicated. Firstly the user should add any compatibility that his games are. This can be done by clicking the ‘compatibilities’ at the ‘games’ page (fig. 11.10).

After adding all the compatibilities we can proceed with the option of adding a new game. The fields here are completed the same as in the movies. The only change here is that the field ‘runtime’ is replaced with ‘producer’ which both accepts all kinds of characters. The new thing here at games is the compatibility and the stock. To select compatibility you have to click the arrow showing down on the drop-down list to give you all the compatibilities that you have added previously at fig. 11.10.

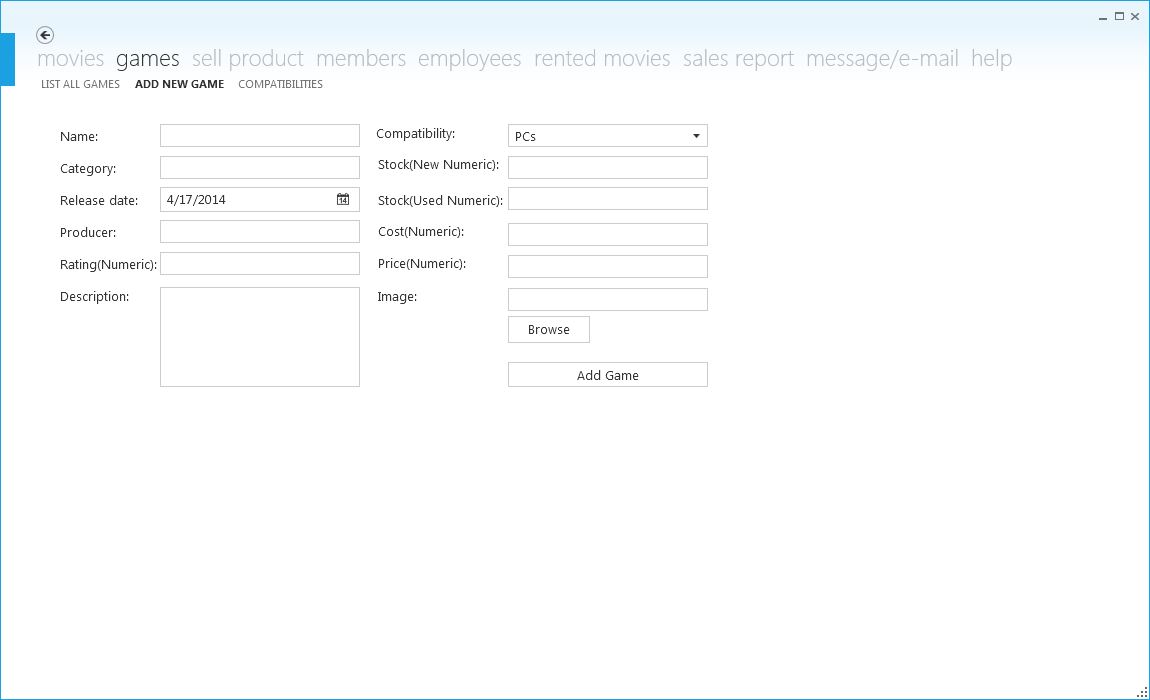


Fig. 11.9 A page showing the form-based interface of adding a new game process.

After compatibility is selected you’ll have to write in the text boxes of ‘stock (new)’ and ‘stock (used)’ the number of new and used games added to the store respectively.

After all of the fields are completed like the example in fig. 11.11 you can proceed the same way as you did in the option of adding a new movie.

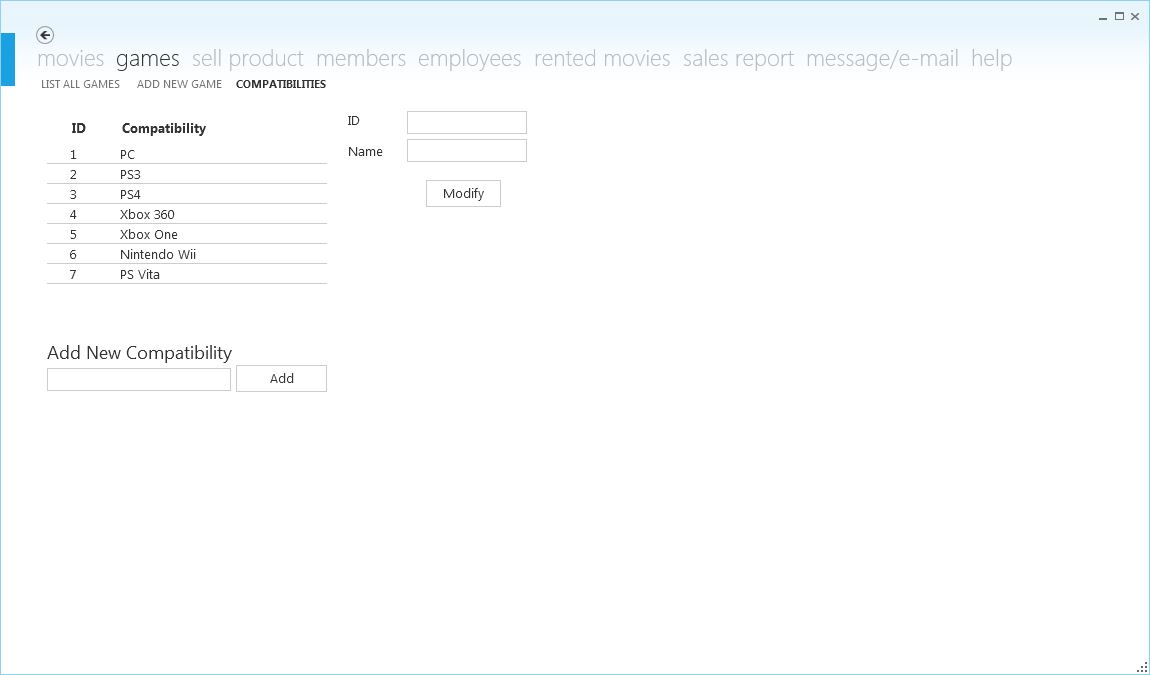


Fig. 11.10 This image shows the page were new compatibilities can be added (this is an example of added compatibilities).

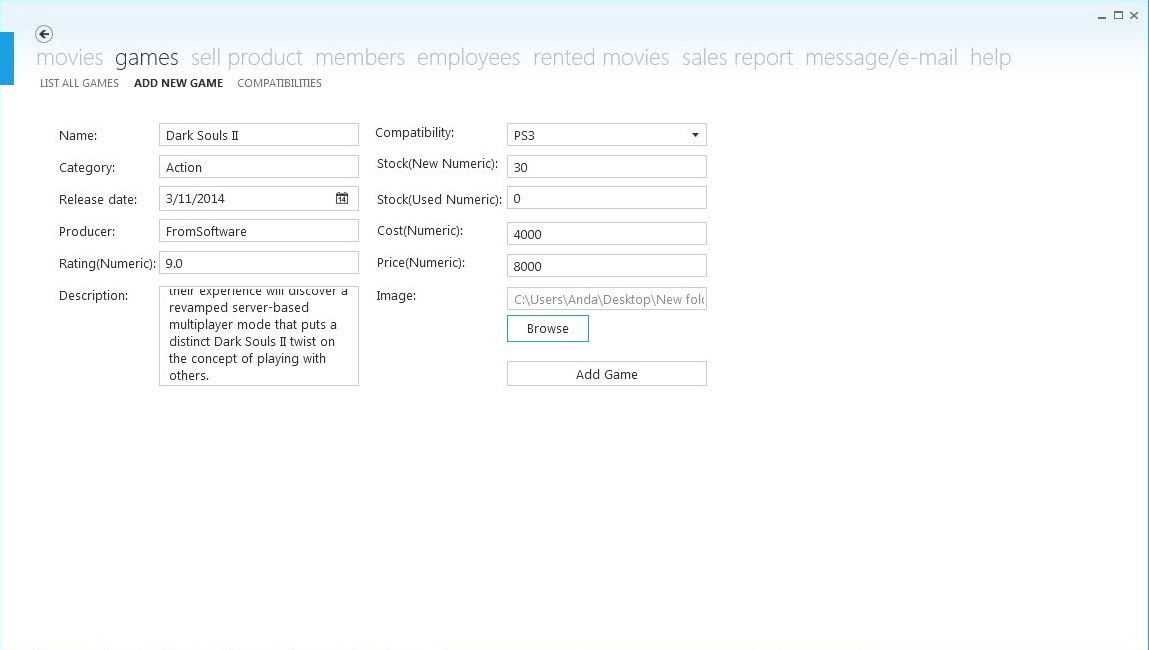


Fig. 11.11 An example of completed form-based interface for adding a new game

3. Adding a member

Another privilege that’s provided when logged in as an administrator or employee is to add new members. This can be done my clicking ‘member’ and then ‘add new member’ (in the fig. 11.4 and 11.5). After doing so the page will change like in the fig. 11.12.

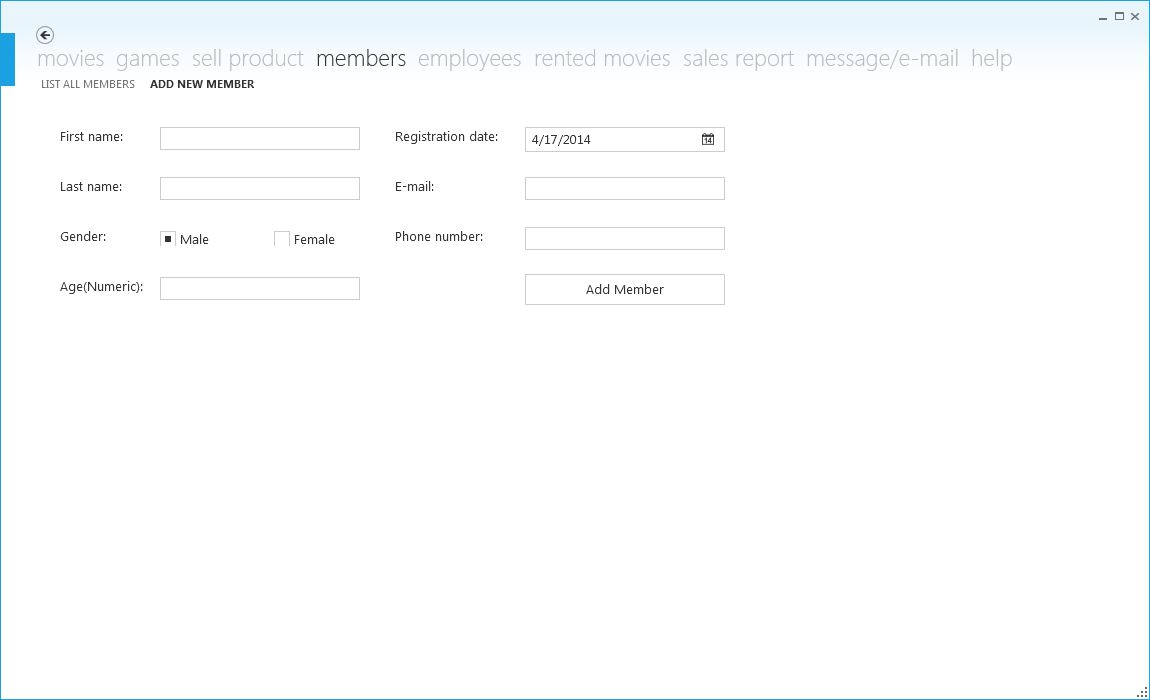
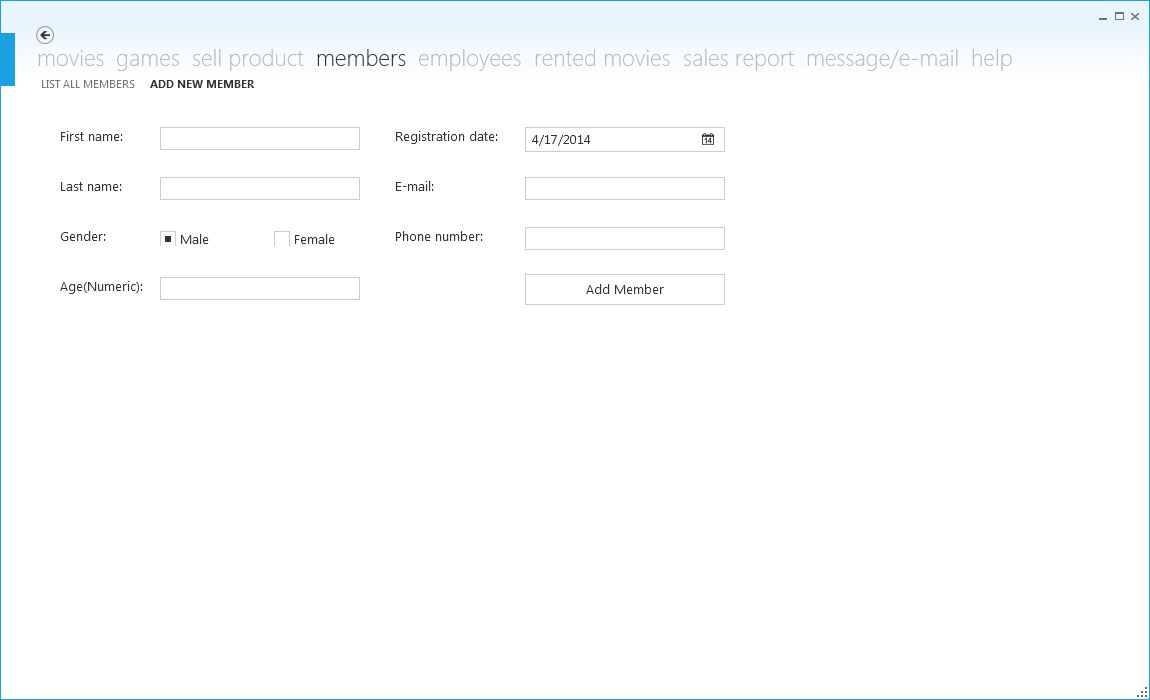


Fig. 11.12 The form-based interface for adding a new member

To add a new member is simpler than the two previous adding options explained above. All the fields accept every type of character. The new thing here is the radio button on which you can check if the member added is male or female. After completing this form by clicking ‘add member’ button you’ll register a new member to the store.

4. Adding an employee

Adding an employee is a function provided only to the administrator. After logged in and the page in the fig. 11.4 will be shown, select ‘employees’ and then ‘add new employee’ for a page like in fig. 11.13 to be shown.

Fig. 11.13 The form-based interface for adding a new employee.

The process of adding an employee is as simple as adding a new member. The two new fields here are ‘username’ and ‘password’. You have to be very careful what you write here because these two fields determine the username and password of the employee account. After you have completed all the fields, you can now click the button ‘add employee’, to successfully add a new employee to the store.

**c) Option 2: Showing a list of movies, games, members or employees**

1. Listing all the movies, games, members or employees

Once you’re logged in as an administrator or an employee you have access to the option number 2 (if you’re logged in as an employee you can’t have a list of all the employees). To do this you first have to click the tab of the item that you want to list (ex. ‘movies’) and then the page that says to list the items that you’re searching for (ex. ‘list all movies’). After doing so a page like in the fig. 11.14 will be shown. You can scroll through the grid that shown on the page in two directions, horizontally and vertically, to view all the elements of the grid.

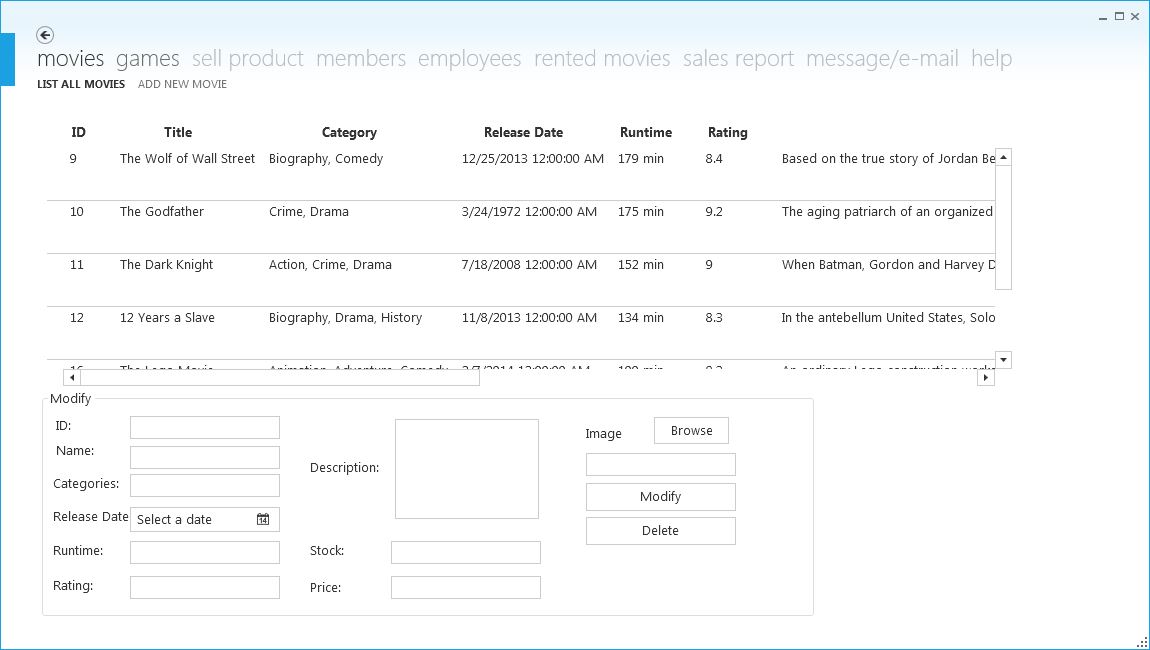


Fig. 11.14 An example of a page listed with all the movies added to the system.

**d) Option 3: Modifying or deleting one of the listed items (movies, games, members, employees)**

1. Modifying one of the listed items.

The process of modifying an item is the same for both the administrator and employee. The difference is that the administrator has the right to modify an employee, while the employee doesn’t have this right.

To modify one of the items in the grid you simply have to double click the item that you want to modify and all the fields in the ‘Modify’ box will be filled with the current details of the item, except the ‘Image’ field in the case of modifying a movie or a game; (example shown in fig 11.15).

When the item is double clicked and all the fields are filled except the ‘Image’ field, you have to browse an image for the item in the case of modifying a movie or a game. This image can be the same image as it was before or a different one. After completing this step (image is needed for movies and games), you can change the desired information in the corresponding text box and then click ‘Modify’ to successfully modified the selected item. The grid will be immediately refreshed with the updated item.



Fig. 11.15 An example of page used to modify a movie. All the fields are filled except the ‘Image’ field.

2. Deleting one of the listed items

To delete an item from one of the lists you’ll have to follow the same steps as in modifying an item (example: movie > list all movies > double click one of the movies; fig. 11.15). After reaching to this phase and the fields are all filled, except the ‘Image’ field in the case of movies and games, you have to simply press the ‘Delete’ button (there’s no need to enter an image in case of movies and games) and the item is deleted immediately from the list.

**e) Selling a product**

When you’re logged in as an administrator or an employee, you can sell a product to a client (ordinary client or member). In the fig. 11.4 (user: administrator) or 11.5 (user: employee) go to the tab sell product to sell a movie or a game to a client (fig 11.16 will appear). The procedure of selling a product to an ordinary client is different than the procedure of selling a product to a member. Each of these procedures will be explained in further detail below.

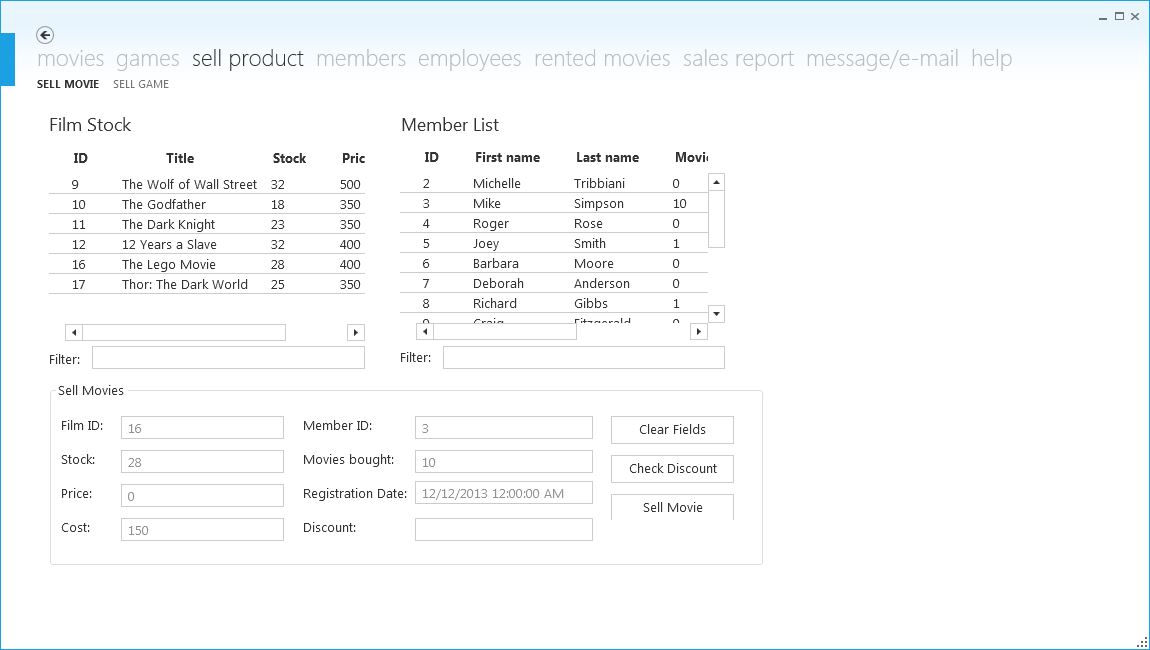


Fig. 11.16 A page which shows the interface for selling a movie (the interface for selling a game is very similar to this one just with a few differences).

1. Selling a product to an ordinary client

In this part of the guide we’re going to explain how a game or a movie is sold to an ordinary client. To do so follow this steps: sell product (tab) > sell movie or sell game (page) > double click a product from the movies or games list on that page > check used or new in the case of the games \*1 (fig. 11. 17) > sell product.

After following these procedures the stock of the product is decremented by 1 and the number of times sold of that product is added by 1. Also the cost and the price for that product are listed in the sales list.

\*1: if ‘new’ is checked then the price and cost remains the same, else if ‘used’ is checked then when the product is sold 30% of the price and cost is subtracted from the original price and cost (this because of the discussion during the interview with the business owner about the used games); this step is skipped for selling a movie.

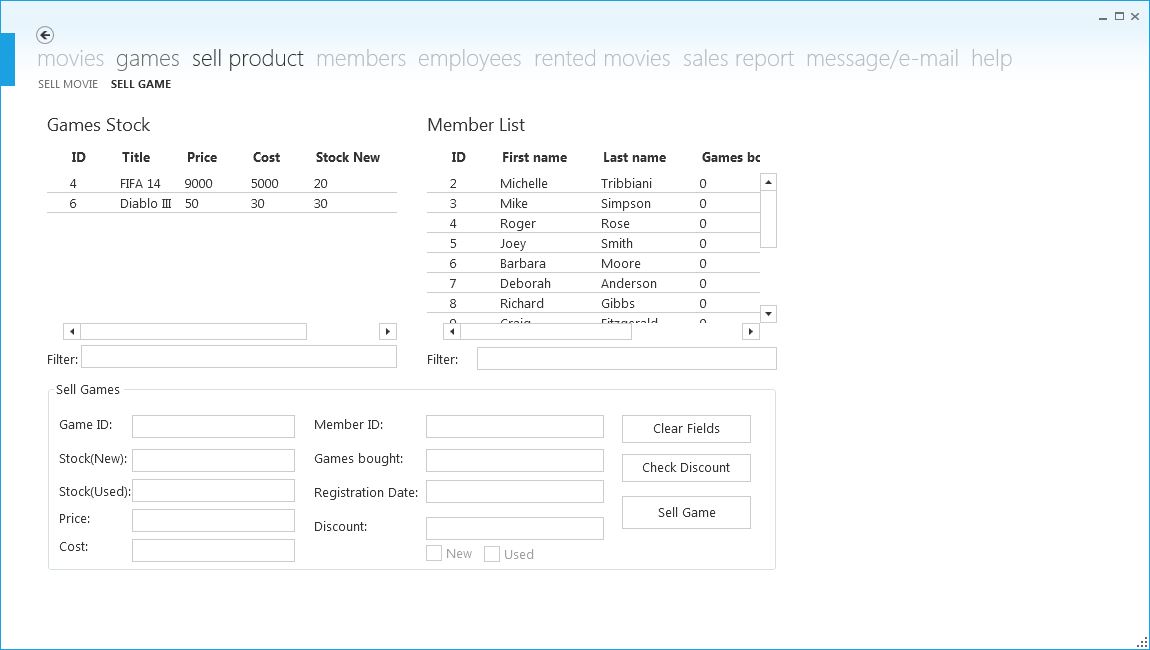


Fig. 11.17 The interface of selling a game.

2. Selling a product to a member

To sell a product to a member is very similar to selling a product to an ordinary client. These are the steps to follow for this procedure: sell product (tab) > sell movie or game (page) > double click a product from the movies or games list in that page > double click the member from the members list on that page > check new or used in case of the games \*1 > check discount \*2 > sell product

\*2: check discount button is based on the discussion during the interview with the business owner about the discount that is made to the client according to time that he has been registered or the amount of products bought; even if this button is not selected during the process the discount is made automatically when the sell movie or game button is pressed.

**f) Renting a movie**

1. Renting a movie

To rent a movie is quite simple. After logged in as an administrator or employee follow these steps: rented movies (tab) > rent movies (page) > double click a movie form the list on that page > select a member from the list on that page > click ‘rent’ (fig. 11.18). When this procedure is done stock of the movie is decremented by one until the movie is returned.

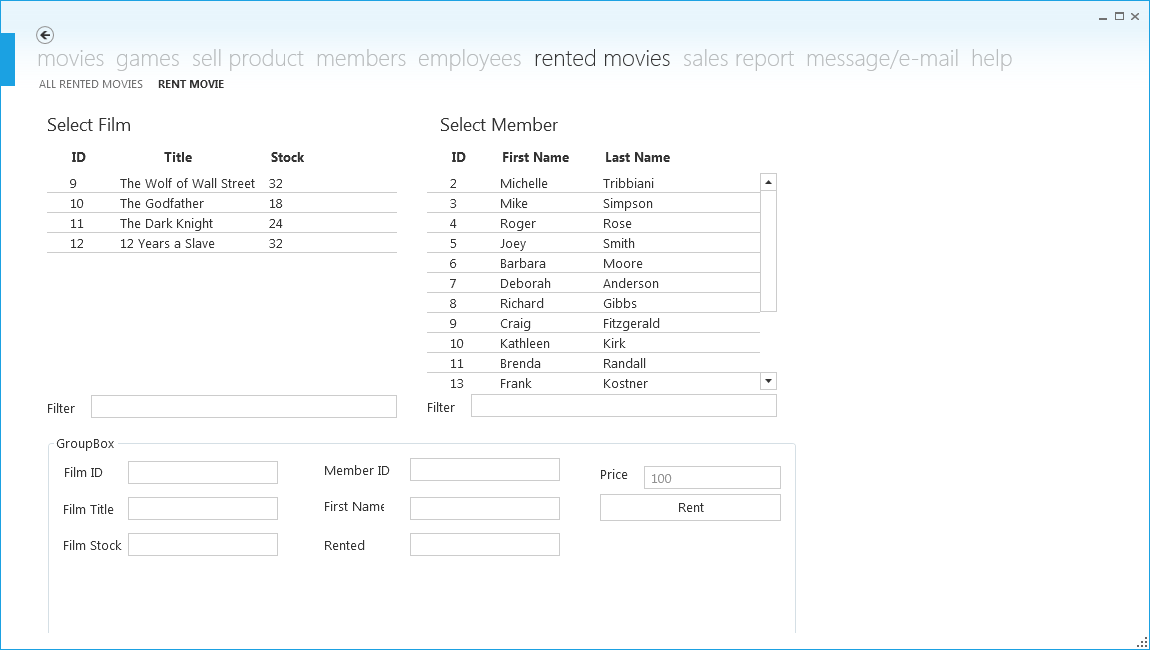


Fig 11.18 The interface of the page ‘Rent Movie’.

2. Listing all the rented movies, returning a movie and postponing the deadline

To view the list of all the rented movies click the page ‘all rented movies’ on the tab ‘rented movies’. The list shows all the rented movies with the deadlines (fig 11.19).

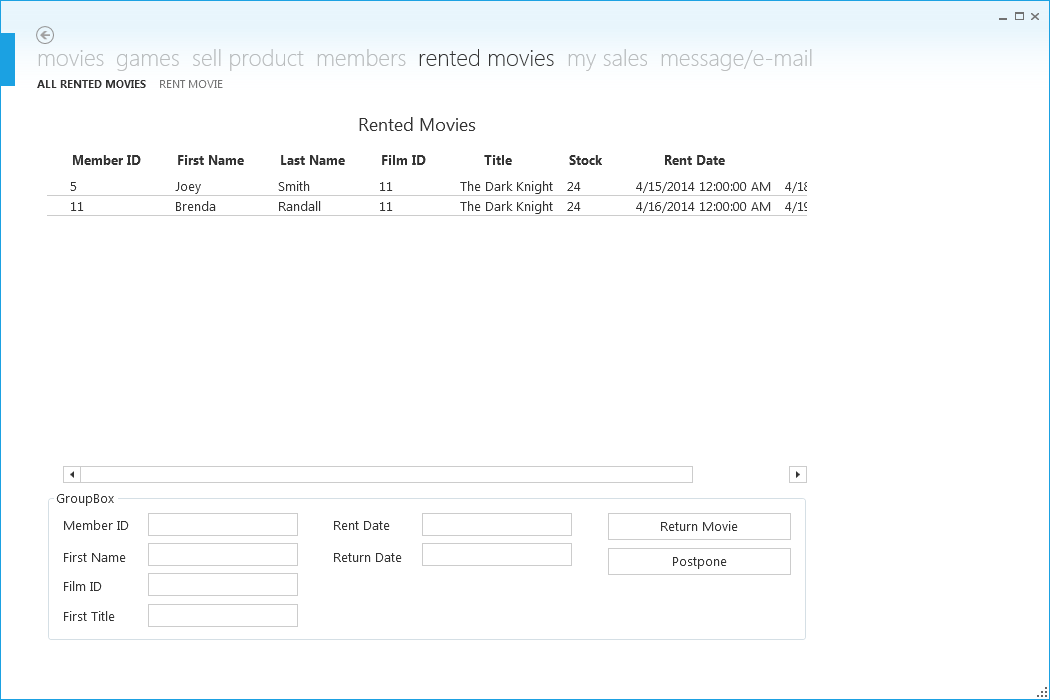


Fig 11.19 An example of the list of all the rented movies.

To return a movie, double click the movie from the list in the fig 11.19 and press ‘return movie’ button. In this case the stock of the movie is added by one again.

To postpone the deadline of a rent double click from the list in the fig 11.19 and press ‘postpone’. The deadline is then postponed by one day.

3. View the list of the rents that the date has passed

When the dateline has passed and the member hasn’t returned the movie yet you can view the list of the past dates rents and e-mail these members from the program at the page ‘passed dates’ at the message/email tab, like in the figure 11.20.

To selects one of the past dates, double click on the table and the fields will be filled. If you want to send an email to that member, you can do so by simply clicking ‘send email’ button.

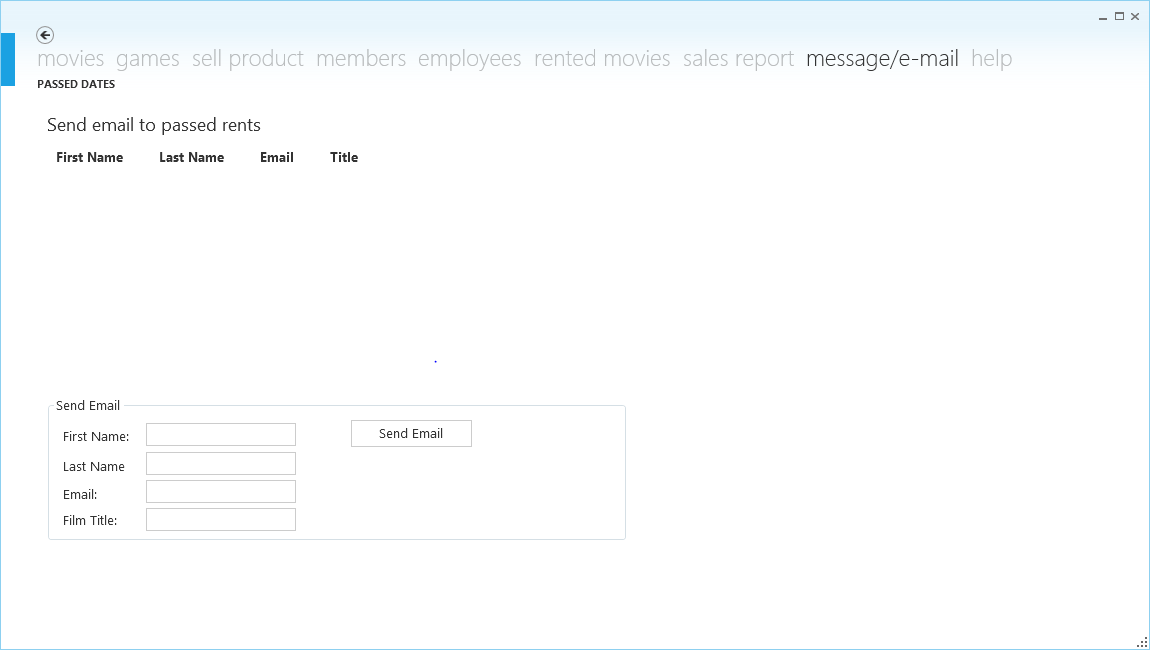


Fig. 11.20 The interface for the list of the past dates and to send e-mail to the members not correct.

**g) Viewing sales**

When you’re logged in as an administrator you have access to the sales. You can view a full report of the sales and the total income.

When you’re logged in as an employee you have access to your sales. You can view the total sales that you as an employee have done.

1. Sales as an administrator

To view the sales follow these steps: sales report (tab) > movie or game sales (page) > select date interval > press ‘show sales’ (fig. 11.21). Below the list of the sales is given the sum of the total sales.

The same way that you can view the movie sales you can also view game sales.

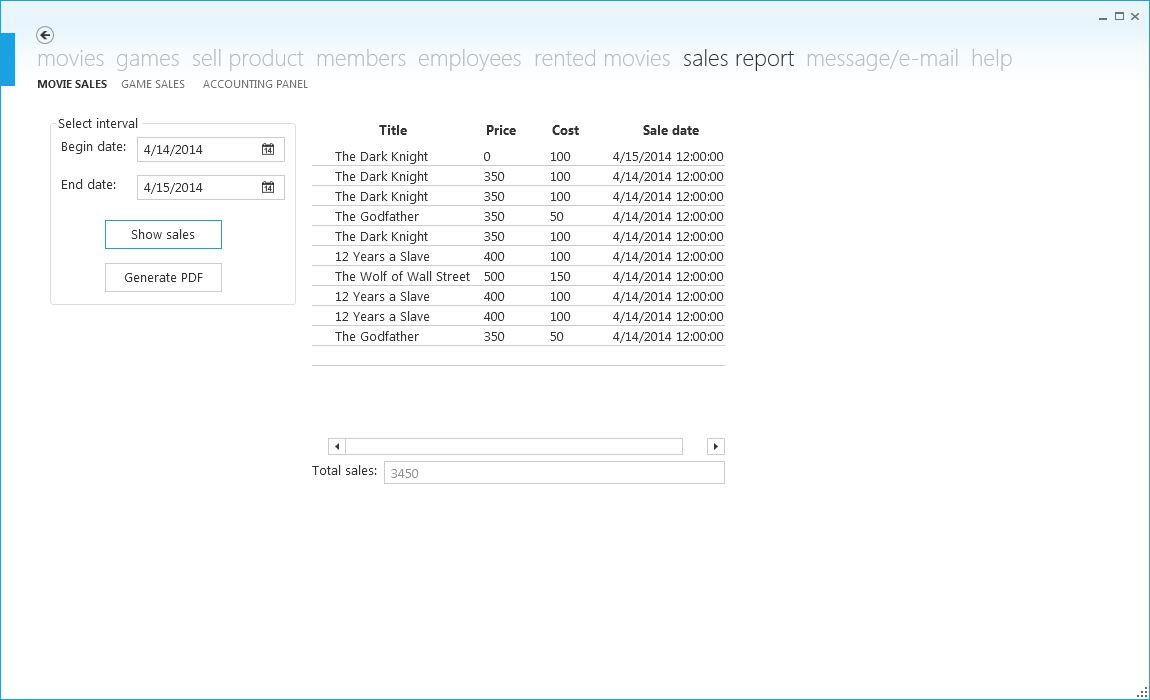


Fig. 11.20 An example of showing the total sales between the selected time interval.

Administrator also has an accounting panel (fig. 11.21) that gives the total turnover for the month and calculates the brute income after you have put the bill for the internet and energy. After you generate a PDF report here (explained later how) you close the report of the month and now it starts calculating for another one.

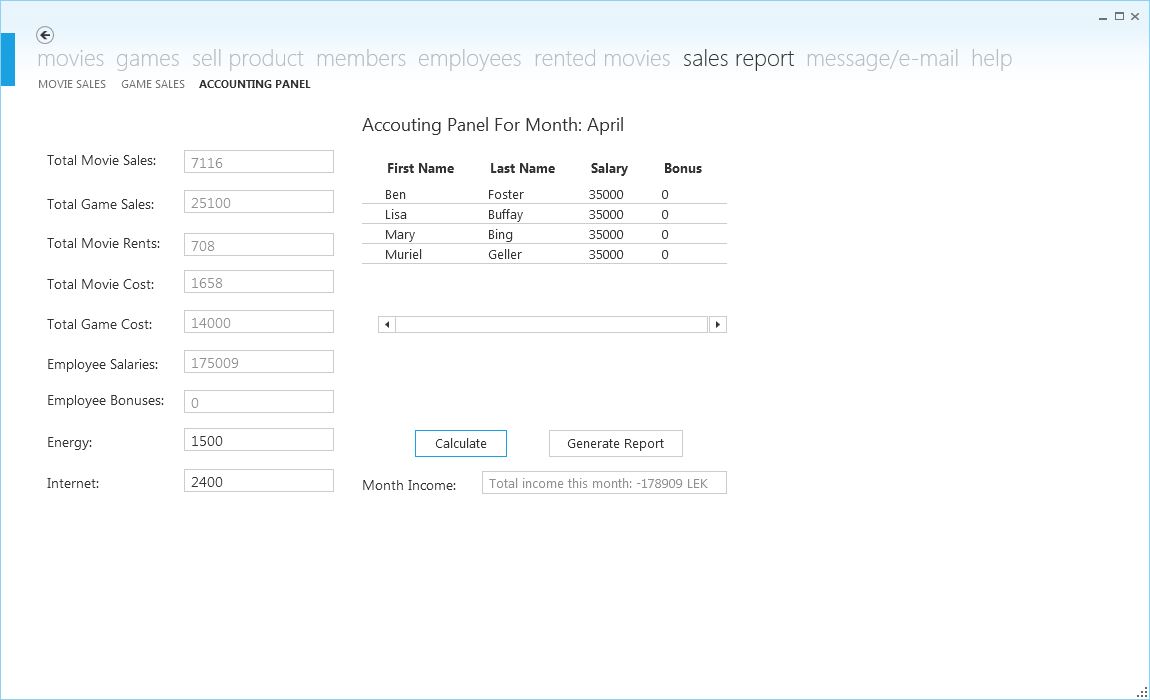


Fig. 11.21 An example of an accounting panel with the data used for this guide.

2. Sales as an employee

To view the sales as an employee follow these steps: my sales (tab) > movie or game sales (page) (fig. 11.22).

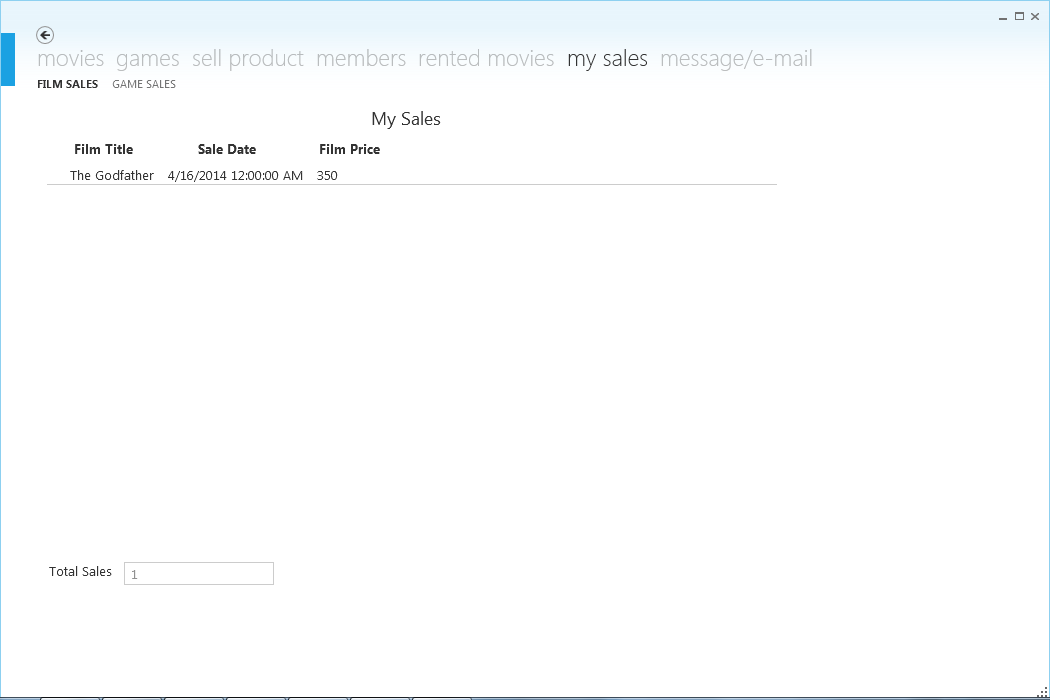


Fig. 11.22 An example of ‘my sales’ list of an employee.

Here at ‘my sales’ you can view a list of the sales that you have done as an employee and the sum of total sales is given below the list.

You can view the game sales the same way that you can view the movie sales.

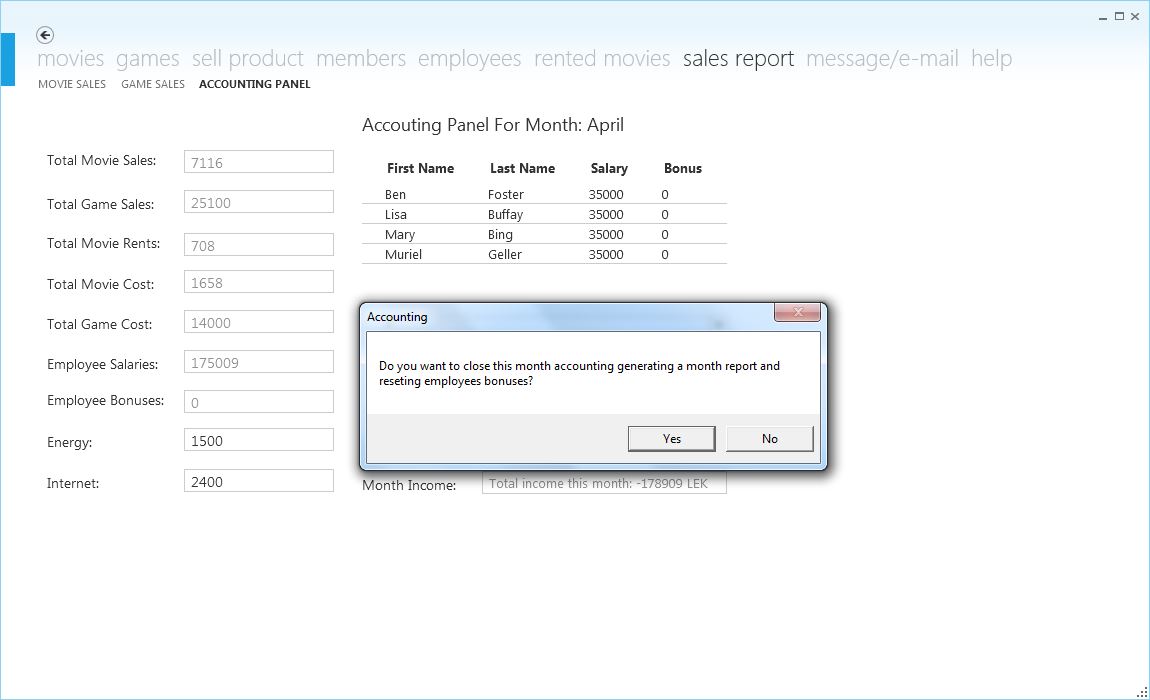


Fig. 11.21 The interface of the page where all the employee sales are shown.

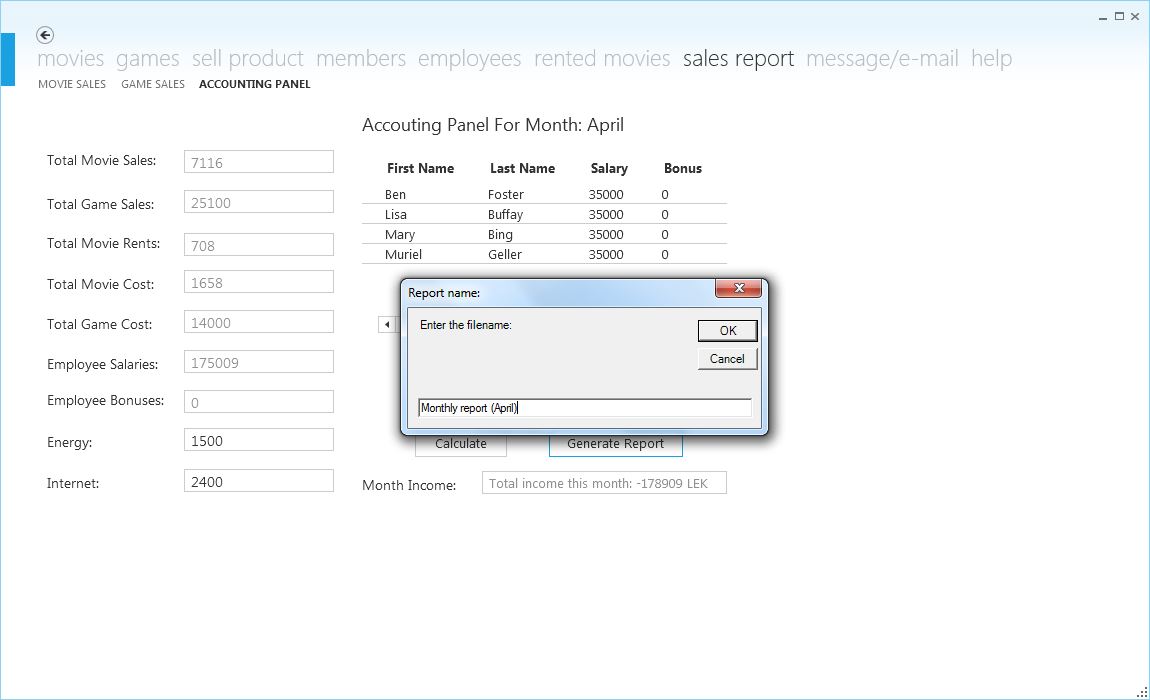
**h) Generating a PDF file of the sales**

To generate a PDF file format of the sales just click ‘generate PDF’ button in any of the pages of ‘sales report’ tab (for example fig. 11.20).

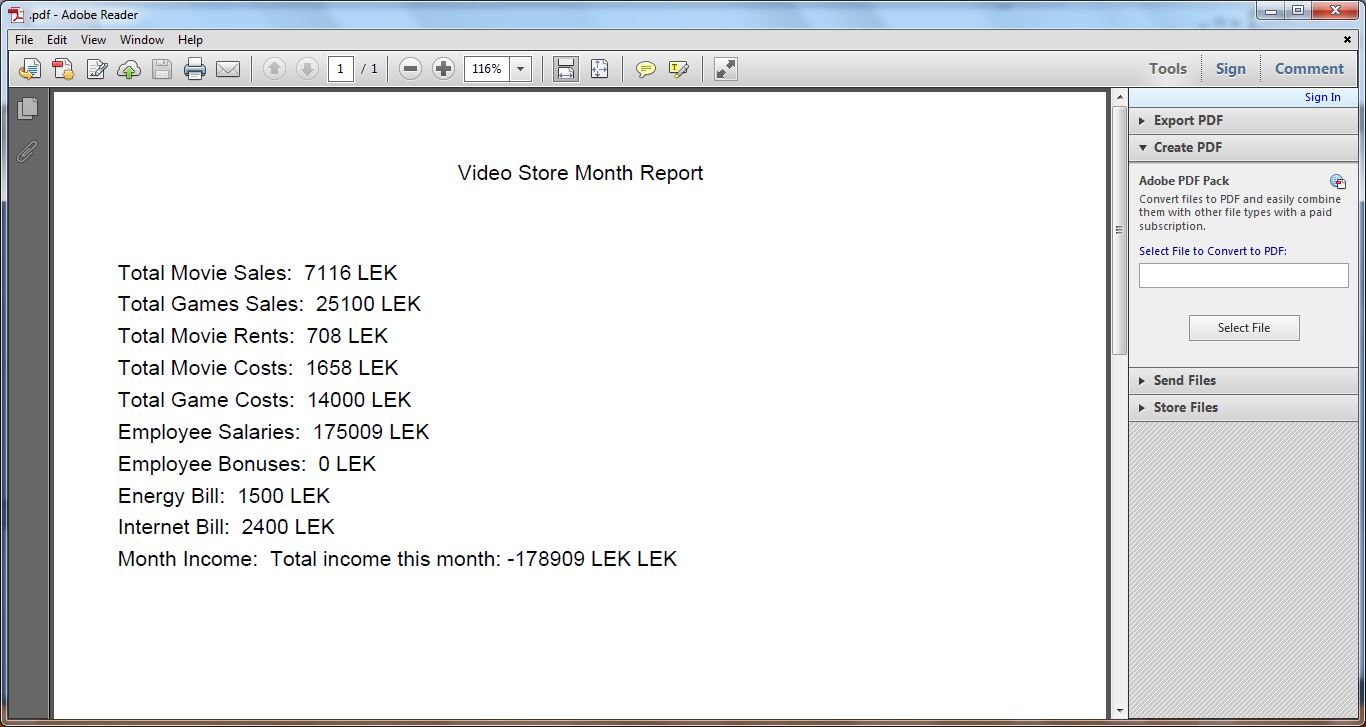
How to generate e report is shown here step by step with photos:



1. After you want to close the report you click generate report and then click ‘yes’ to close the report for the month and also resetting employee bonuses. (This step is skipped for generating a report of movie sales or game sales.



2. Then it’ll require you to name the report. If it exists it will require a new name.



3. After clicking ‘ok’ at the figure above it will give you a PDF file of the report that you generated (this is an example of the report of the accounting panel).

This PDF files are stored in a folder call ‘Report’ at Local Disk (C:). Do not move this file anywhere else because then the program won’t be able to generate a PDF format of the sales.

**i) Guest**

The guest panel can make a list of the movies or games. There are three types of lists that this panel can create: top rated, newest entering and top sold (fig. 11.23, 11.24 and 11.25). Another thing that can be done at guest panel is to search a type of movie or game (fig. 11.26).

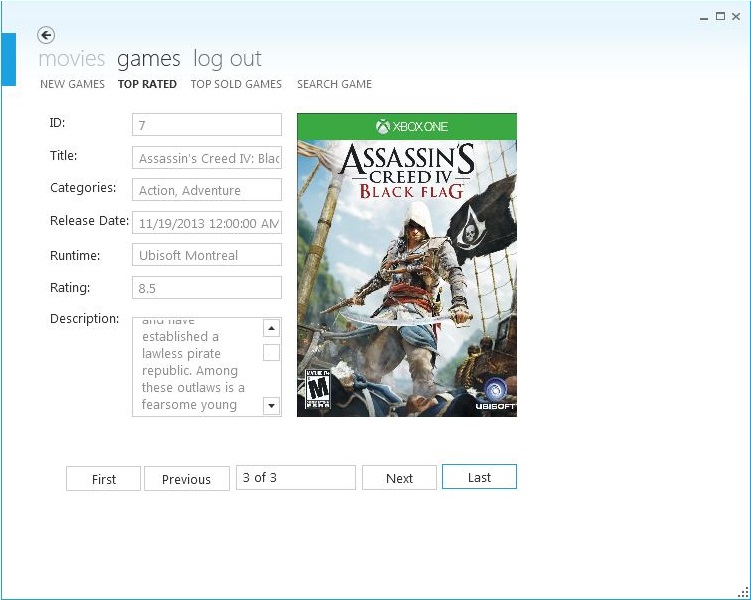


Fig. 11.23 Top Rated Category

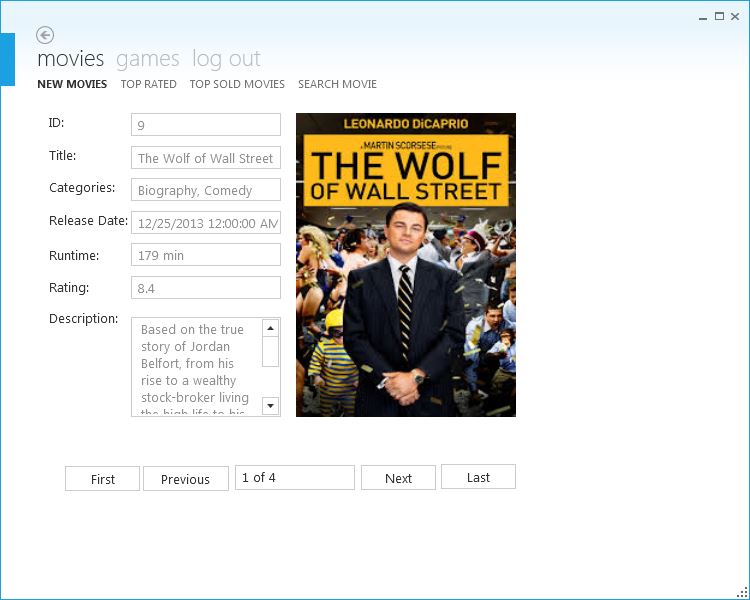


Fig. 11.24 Newest Entering Movies

If you want a list or search a product just click the tab of movies if you want movies or games if you want games. Then click the type of list that you want (fig. 11.23, 11.24 and 11.25). If you want to search click the page like in the fig. 11.26.

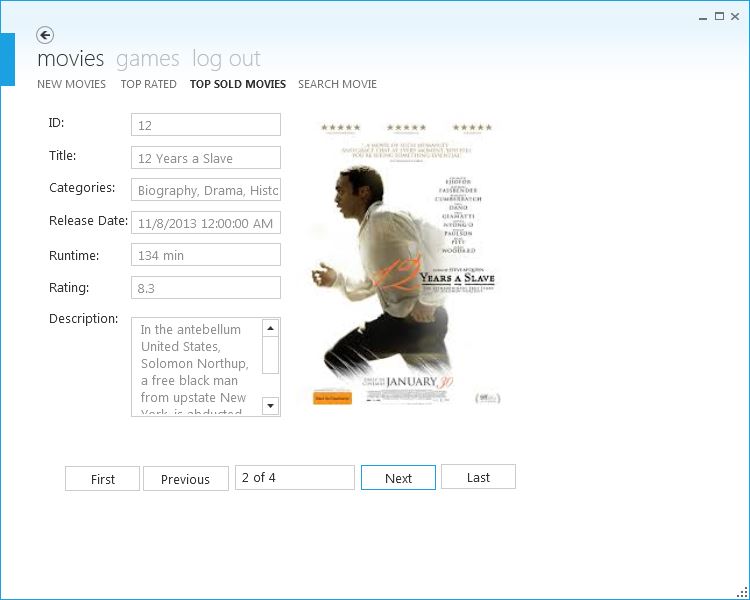


Fig. 11.25 Top Sold Movies

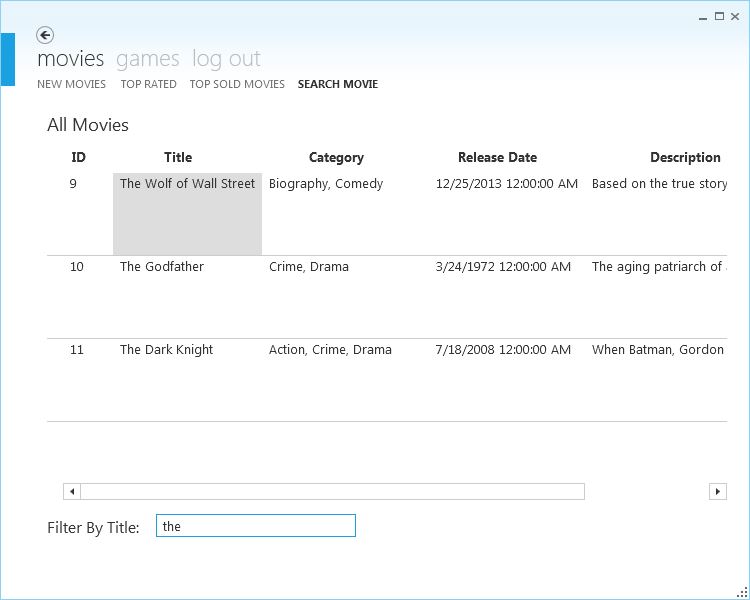


Fig. 11.26 Searching a Movie

For the lists you can view the movies with the buttons as in each one of these pages. ‘Next’ and ‘Previous’ buttons move to the next or the previous item on list respectively depending on what you’re ranking the product. The buttons ‘First’ and ‘Last’ take you to the first and last items on the list respectively.

To search for a product go the search page of movies and games and the list of all movies is presented (fig. 11.26). To type for a specific product, type the full name of the product or a key word of that product at ‘the filter by’ text box (fig. 11.27).

**j) Help**

At the last tab of the program as an administrator or employee there are three last pages. The first page is a troubleshooting guide (fig. 11.28), the other one is ‘my account’ on which the password is changed (fig. 11.29) and the final one is the log out page from which you can log out with a simple button (fig. 11.30). After logged out, the log in panel appears again (f­ig. 11.31).

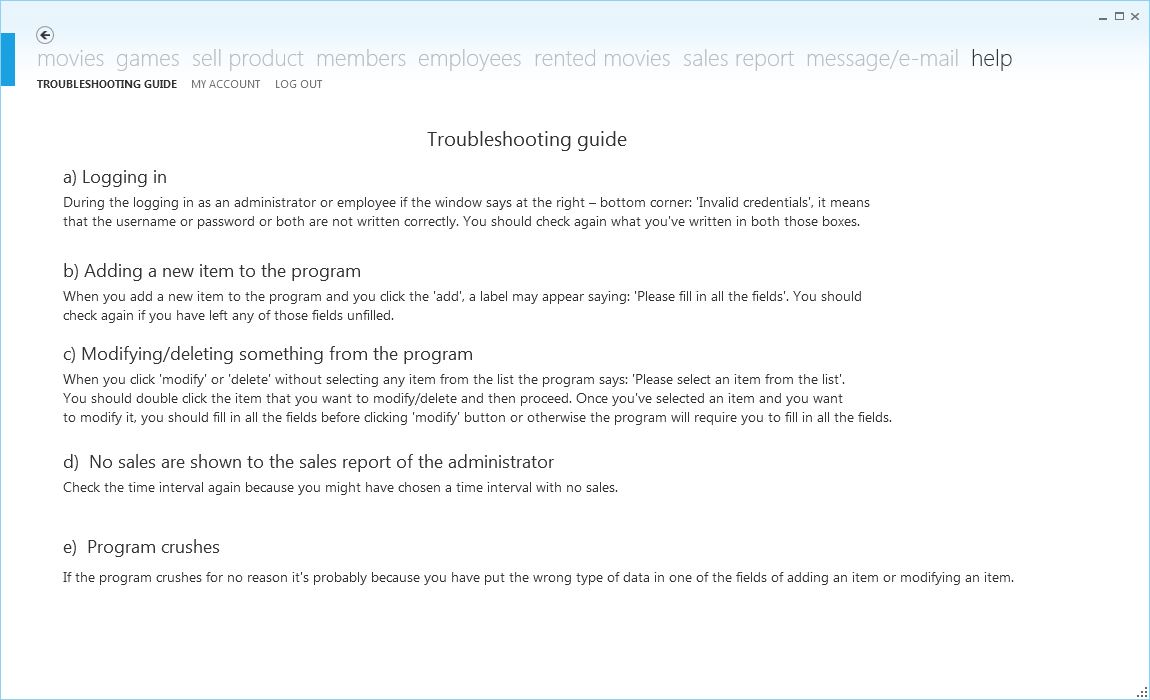


Fig. 11.28 The troubleshooting guide for the program.

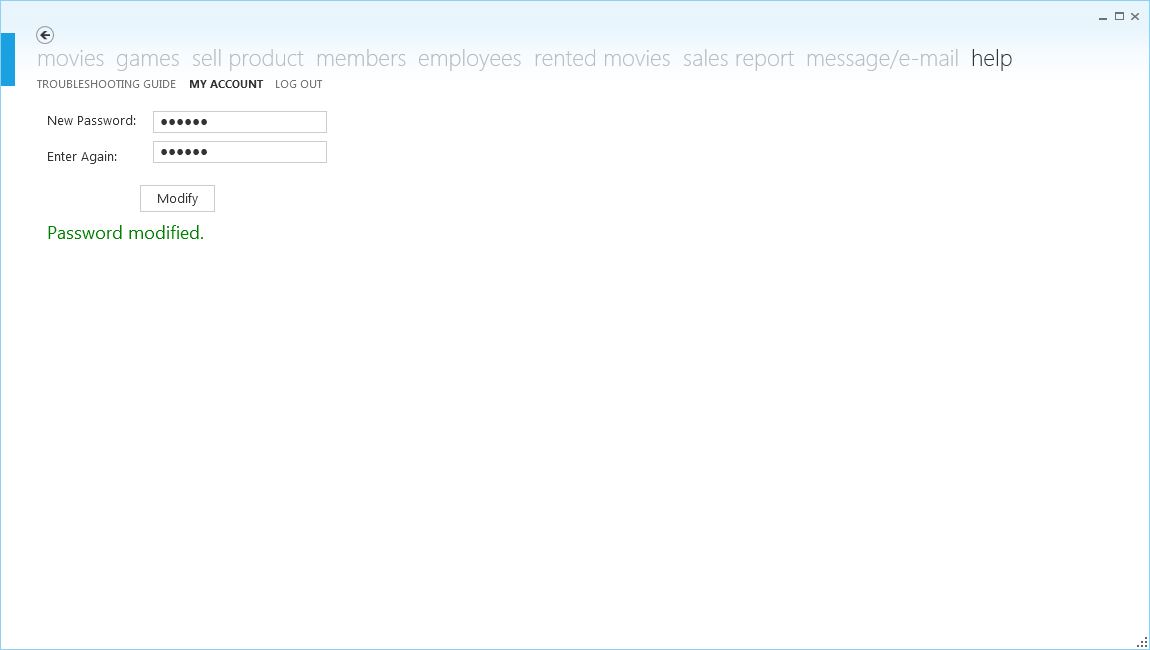


Fig 11.29 The interface for changing the password for administrator (employee is the same).

To change the password at the ‘my account’ page, just type the new password two times and then click ‘modify’ button. This way of changing password is for both the administrator and the employee.

At the help page of the guest panel there’s only the logging out page which will be explained later.

**k) Logging out**

To log out as an employee or administrator follow these steps (fig 11.30): help (tab) > log out (page) > log out (button).

To log out as a guest follow these steps: log out (tab) > log out (page) > log out (button)

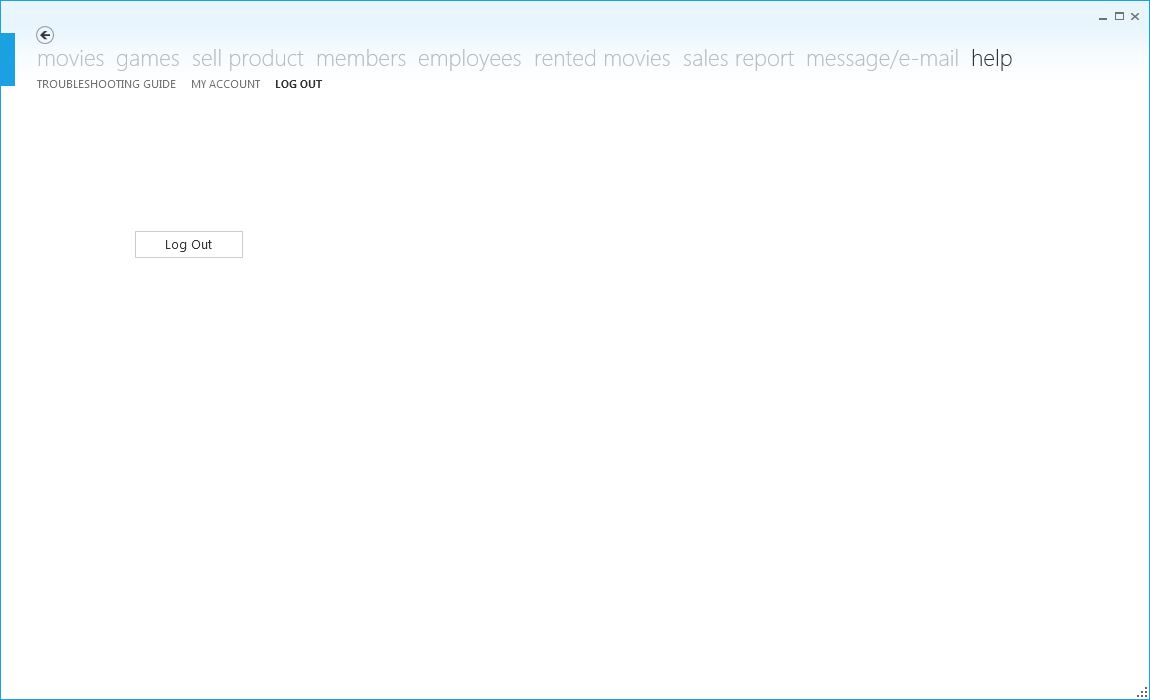


Fig. 11.30 The log out page for an administrator. Employee’s log out page is similar to this one.

After logging out, the log in window appears (fig. 11.31).

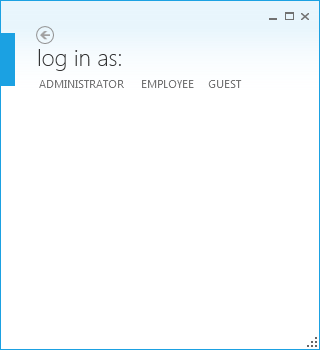


Fig 11.31 Logging in panel.

**iii) Troubleshooting guide**

**a) Logging in**

During the logging in as an administrator or employee if the windows says at the right – bottom corner: “Invalid credentials”, it means that the username or password or both are not written correctly. You should check again what you’ve written in both those boxes.

**b) Adding a new item to the program**

When you add a new item to the program and you click the ‘add’, a label may appear saying: “Please fill in all the fields”. You should check again if you have left any of those fields unfilled

**c) Modifying/deleting something from the program**

When you click ‘modify’ or ‘delete’ without selecting any item from the list the program says: “Please select an item from the list’. You should double click the item that you want to modify/delete and then proceed.

Once you’ve selected an item and you want to modify it, you should fill in all the fields before clicking ‘modify’ button or otherwise the program will require you to fill in all the fields.

**d) No sales are shown to the sales report of the administrator**

Check the time interval again because you might have chosen a time interval with no sales.

**e) Program crushes**

If the program crushes for no reason it’s probably because you have put the wrong type of data in one of the fields of adding an item or modifying an item.

**12. Evaluation**

**i) Success in meeting the original objectives**

**ii) The user’s response to the system**

**13. Glossary**

**A**

**Alpha testing** – the software is tested at the development company by a number of testers. At this stage it may well still have faults.

**Annotation** – text, usually in the form of comment statements, added to a source of program.

**Application software** – all the computer software that causes a computer to perform useful tasks beyond the running of the computer itself.

**Authentication –** checking that a user is who they say they are; digital signatures attached to an email are designed to check that the sender is who they appear to be.

**B**

**Beta testing** – the software under development is released to a number of privileged costumers for their feedback, before it is commercially released.

**Boolean** – data type that allows two values, TRUE or FALSE.

**Byte** – number of bits to store one character. Generally considered as eight bits.

**C**

**CD-ROM** – secondary storage media with data read and written with a laser; data contents cannot be changed; used for archiving backups.

**Character** – single character from a character set. Includes; upper and lower letters, digits, punctuation and ‘control’ characters (e.g. <Carriage Return>).

**Class** – in object-oriented programming, the blueprint design of an object that describes the properties and methods of the object.

**Class diagram** – in object-oriented programming, a diagram that shows how actual instances of one or more objects are related to an instance of another class.

**D**

**Database** – a structured set of data held in a computer, especially one that is accessible in various ways.

**Dataflow diagram (DFD)** – charting technique used to describe data flowing through a system: inputs, processes and outputs.

**Data grid** – a graphical user interface element that presents a tabular view of data.

**Design specification** – formal document that shows how the features of the requirements specification (inputs, processing and outputs) will be implemented.

**E**

**Entity** – in database design, something about which we store data, e.g. a customer.

**F**

**Feasibility study** – report on whether or not a computerized solution is feasible.

**Flowchart** – flowcharting techniques are used in many areas of computing; a program flowchart is used to describe the sequential steps in an algorithm.

**Folder** – a virtual area of storage referred to by name. Folder creation and management is done by the file manager module of the operating system. Also called “Directory”.

**Form-based interface** – interface through which the user communicates with the computer by entering data into controls (possibly arranged similar to a paper-based form), e.g. text boxes, radio buttons, check boxes and drop-down lists.

**Function (as procedure) header** – the first line of a function definition that contains the identifier name and parameters; also called the “function interface”.

**G**

**Graphic card** – an expansion card which generates a feed of output images to a display.

**H**

**Hard copy** – printed output

**Hard disk** – secondary storage device consisting of disk platters that rotate at high speed.

**Hardware** – the physical – mostly electrical and electronic – components that make up a computer system, e.g. a device controller or motherboard.

**I**

**Indentation** – programming style designed to help the “readability” of the source code, e.g. by indenting all the statements inside a loop or a selection structure.

**Integer** – whole number; a data type in most programming languages that stores values in two bytes (i.e. from -2768 to +2768)

**Interface** – a device or program enabling a user to communicate with a computer.

**Internet** – infrastructure of computers, networks and routers that uses various communications methods to link devices globally.

**M**

**Maintenance** – changes that may need to take place to a system after it has become operational.

**Modular approach** – the design methodology of taking a problem and breaking it down in smaller related tasks.

**Module** – self-contained part of a large computer program that provides features that are accessed by or integrated with other sections of program code; also called “program module”.

**N**

**Network interface card** – hardware that provides the connection from the network cable to the computer (motherboard), also called “network interface card (NIC)”.

**O**

**Object** – instance of a class

**Object-oriented programming (OOP)** – programming paradigm that uses a bottom-up approach by first defining the objects used by the problem.

**Operating system (OS) software** – software that makes computer hardware useable and provides an interface between the hardware and the user.

**P**

**Parallel running** – implementation in which the new system runs alongside with the old system for a limited period of time.

**Procedure** – block of program code referred to with an identifier name; it can be “called” many times throughout the program; it may or may not have parameters.

**Processor (CPU)** – the hardware device in a computer that executes all of the instructions from the software.

**Q**

**Query** – structured question used to retrieve information from a database or knowledge base.

**R**

**Random access memory (RAM)** – the most common computer memory which can be used by programs to perform necessary tasks while the computer is on; an integrated circuit memory chip allows information to be stored or accessed in any order and all storage locations are equally accessible

**Relationship** – link between two database tables made using a primary key and foreign key.

**Router** – hardware device used on a packet switching network to direct packets from the sending device to the receiving device.

**Run-time error** – error that only becomes apparent when the program is executed, e.g. caused by an attempted division by zero.

**S**

**Server** – computer that provides to “client” computers. The most basic server would be a file server.

**Software** – sequence of instructions designed to make a computer system perform some task.

**Spreadsheet** – software that provides a grid for the presentation of data and does calculations.

**Structured Query Language (SQL)** – industry-standard data description language and data manipulation language used by database and database management software.

**System analyst** – an individual in charge of designing, modifying, or analyzing various systems to ensure compatibility and user effectiveness.

**System flowchart** – diagram that focuses on the programs and files by a system.

**System software** – operating systems, programming language translators, utility programs and library programs.

**T**

**Table** – in relational database software, a table in the implementation of an entity in the problem.

**Touch screen** – monitor screen that acts as both an input and output device. Finger contact with a particular position on the screen is mapped by software to an action.

**V**

**Validation check** – checking the corrections (or validity) of a data value: presence, format, length, “from a list”, unique, range and check digit.

**Variable** – data value with an identifier that may change during the execution of a program.

**W**

**White box testing** – testing designed to analyze the structure of a program; test data are devised that test all possible routes through the code.

**Wizard** – a help feature of a software package that automates complex tasks by asking user a series of easy-to-answer questions.

**WPF Application** – a Microsoft’s development tool for Web applications and rich client applications.