

TECHNICAL UNIVERSITY OF CLUJ-NAPOCA

Laboratory Work – Assignment 4
Bank Application

***Name:** Popa Florin*

***Group:** 30425*

Table of Contents

1. Introduction	3
1.1 Problem specification	3
2. Design	4
2.1 UML diagrams	4
3. Packages and Interfaces	8
4. User Interface	10
5. Conclusions	11
6. Bibliography	11

1. Introduction

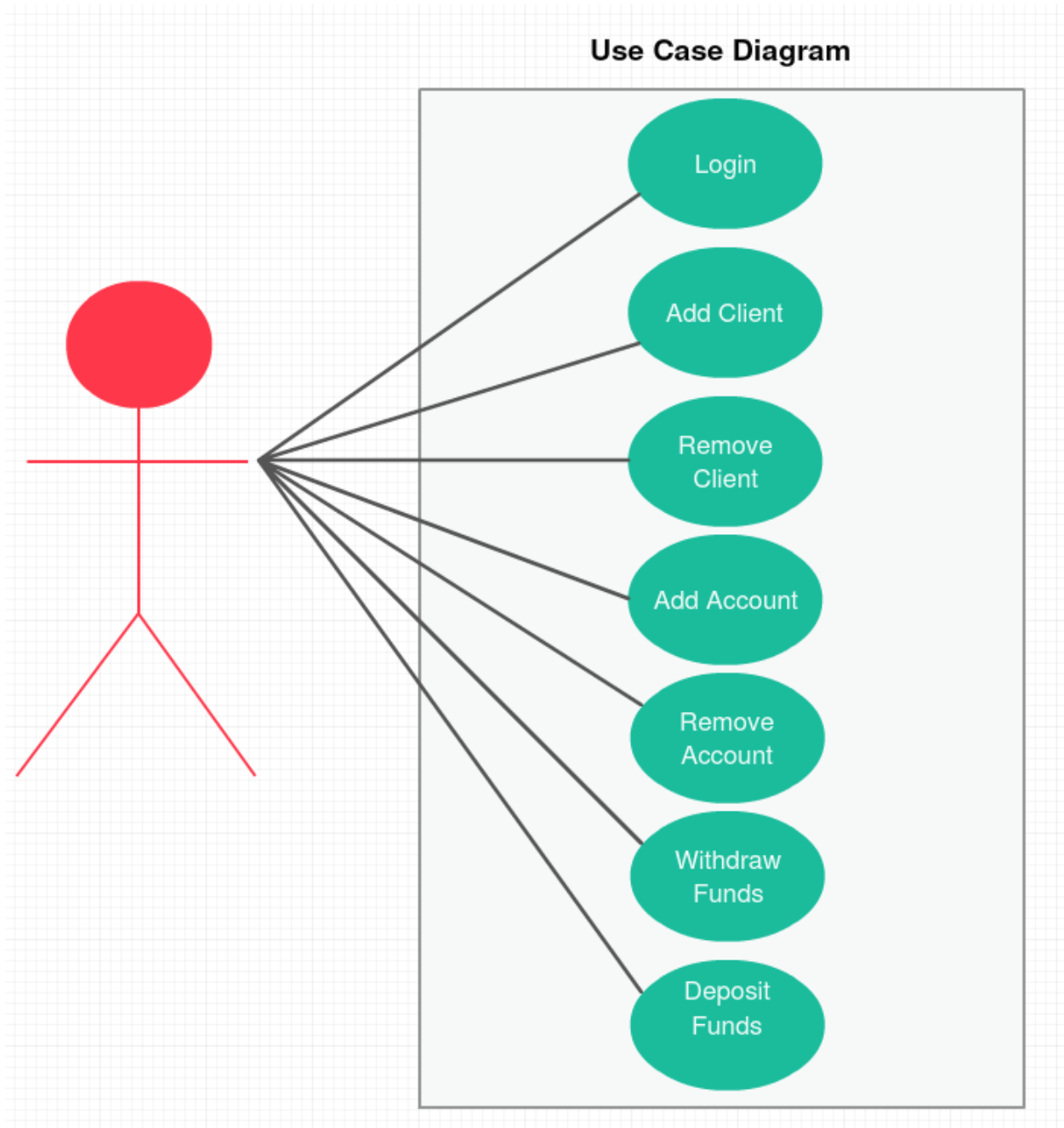
1.1 Problem specification

- 1.2 Define the interface BankProc (add/remove persons, add/remove holder associated accounts, read/write accounts data, report generators, etc). Specify the pre and post conditions for the interface methods.
- 1.3 2. Define and implement the classes Person, Account, SavingAccount and SpendingAccount. Other classes may be added as needed (give reasons for the new added classes).
- 1.4 3. An Observer DP will be defined and implemented. It will notify the account main holder about any account related operation.
- 1.5 4. Implement the class Bank using a predefined collection which uses a hashtable. The hashtable key will be generated based on the account main holder (ro. titularul contului). A person may act as main holder for many accounts. Use JTable to display Bank related information.
- 1.6 4.1 Define a method of type “well formed” for the class Bank.
- 1.7 4.2 Implement the class using Design by Contract method (involving pre, post conditions, invariants, and assertions).
- 1.8 5. Implement a test driver for the system.
- 1.9 6. The account data for populating the Bank object will be loaded/saved from/to a file.

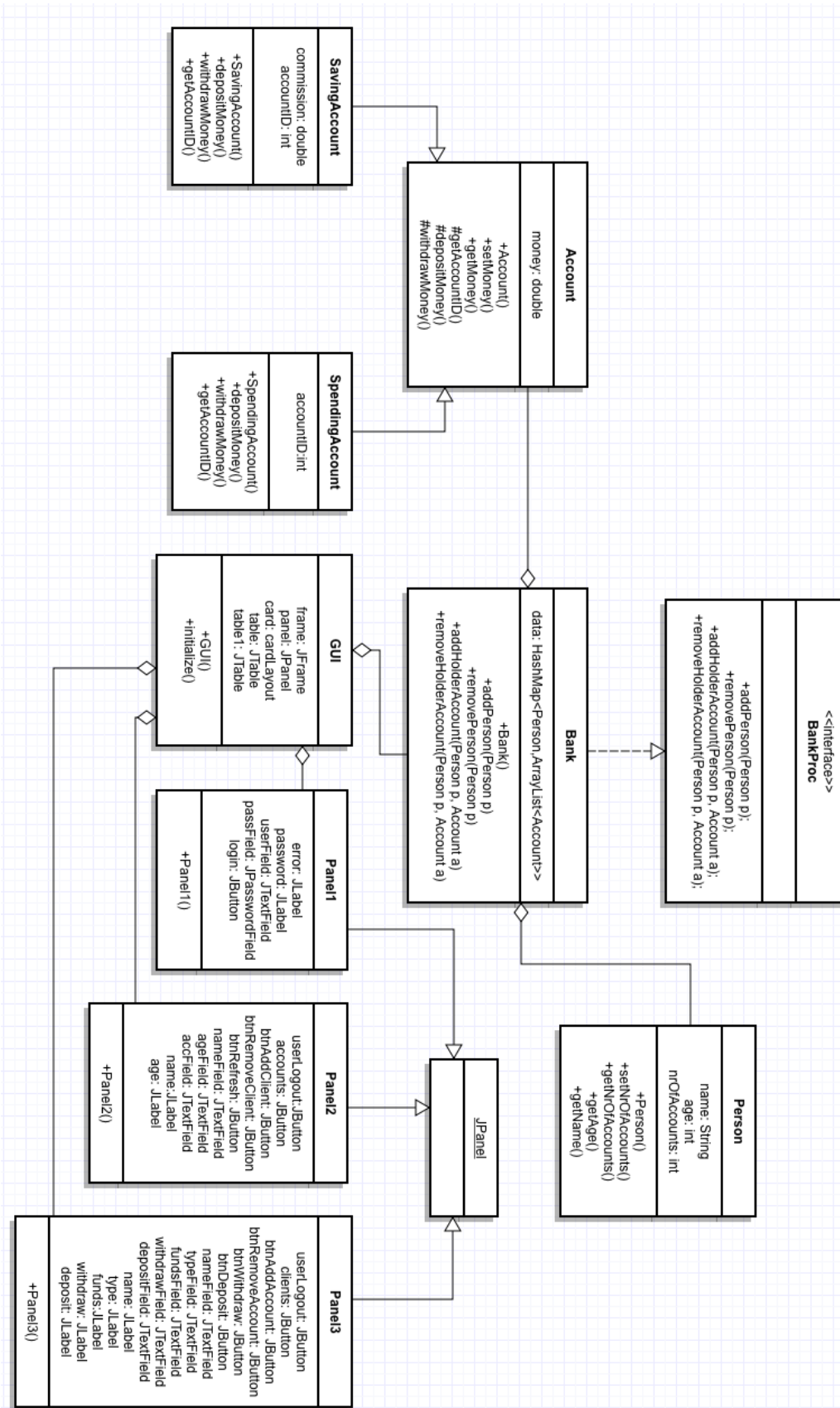
2. Design

2.1 UML diagrams

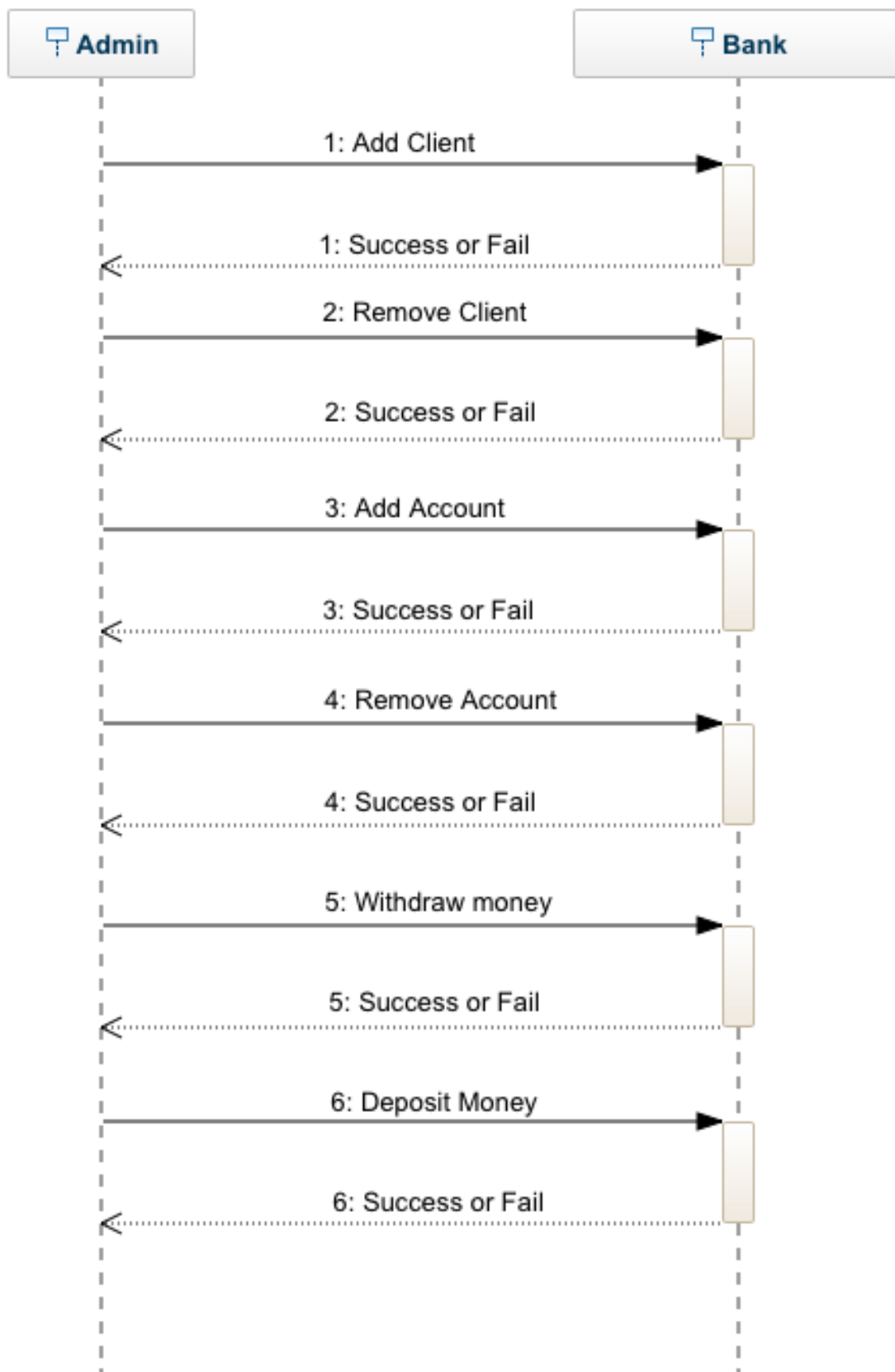
a) Use Case Diagram



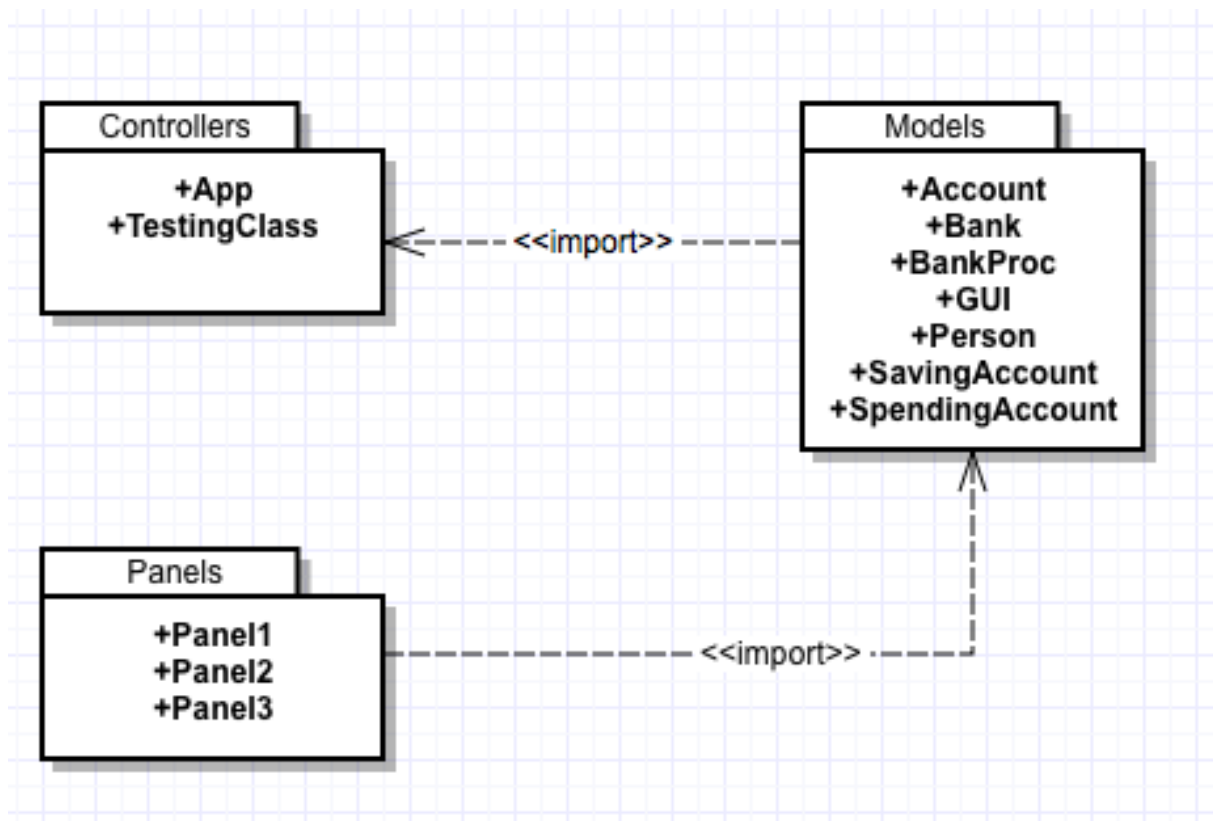
b) Class Diagram



c) Sequence diagram



d) Package diagram



3. Packages and Interfaces

A Java package is a mechanism for organizing Java classes into namespaces. Java packages can be stored in compressed files called JAR files, allowing classes to download faster as a group rather than one at a time. Programmers also typically use packages to organize classes belonging to the same category or providing similar functionality. A package provides a unique namespace for the types it contains. Classes in the same package can access each other's package-access members.

A package allows a developer to group classes (and interfaces) together. These classes will all be related in some way – they might all have to do with a specific application or perform a specific set of tasks.

For this application the following packages are imported, each of them having a certain role for the proper working of the application. We import them in the GUI Class (most of them relate to the user interface properties):

- 1) `import java.awt`: Contains all of the classes for creating user interfaces and for painting graphics and images. A user interface object such as a button or a scrollbar is called, in AWT terminology, a component. The `Component` class is the root of all AWT components.
 - a. `java.awt.Font`: The `Font` class represents fonts, which are used to render text in a visible way. A font provides the information needed to map sequences of characters to sequences of glyphs and to render sequences of glyphs on `Graphics` and `Component` objects.
- 2) `import java.awt.event` : Used in interacting with input devices such as the mouse and keyboard
 - a) `java.awt.event.ActionEvent`: A semantic event which indicates that a component-defined action occurred. This high-level event is generated by a component (such as a `Button`) when the component-specific action occurs (such as being pressed). The event is passed to every `ActionListener` object that registered to receive such events using the component's `addActionListener` method.
 - b) `java.awt.event.ActionListener`: The listener interface for receiving action events. The class that is interested in processing an action event implements this interface, and the object created with that class is registered with a component, using the component's `addActionListener` method. When the action event occurs, that object's `actionPerformed` method is invoked.
- 3) `import javax.swing`: Swing is a GUI widget toolkit for Java. It is part of Oracle's Java Foundation Classes (JFC) – an API for providing a graphical user interface (GUI) for Java programs.

Swing was developed to provide a more sophisticated set of GUI components than the earlier Abstract Window Toolkit (AWT). Swing provides a native look and feel that emulates the look and feel of several platforms, and also supports a pluggable look and feel that allows applications to have a look and feel unrelated to the underlying platform. It has more powerful and flexible components than AWT. In addition to familiar components such as buttons, check boxes and labels, Swing provides several advanced components such as tabbed panel, scroll panes, trees, tables, and lists.

- a) `javax.swing.JButton`: Buttons can be configured, and to some degree controlled, by Actions. Using an Action with a button has many benefits beyond directly configuring a button.
- b) `javax.swing.JFrame`: An extended version of `java.awt.Frame` that adds support for the JFC/Swing component architecture.
- c) `javax.swing.JLabel`: A display area for a short text string or an image, or both. A label does not react to input events. As a result, it cannot get the keyboard focus. A label can, however, display a keyboard alternative as a convenience for a nearby component that has a keyboard alternative but can't display it.

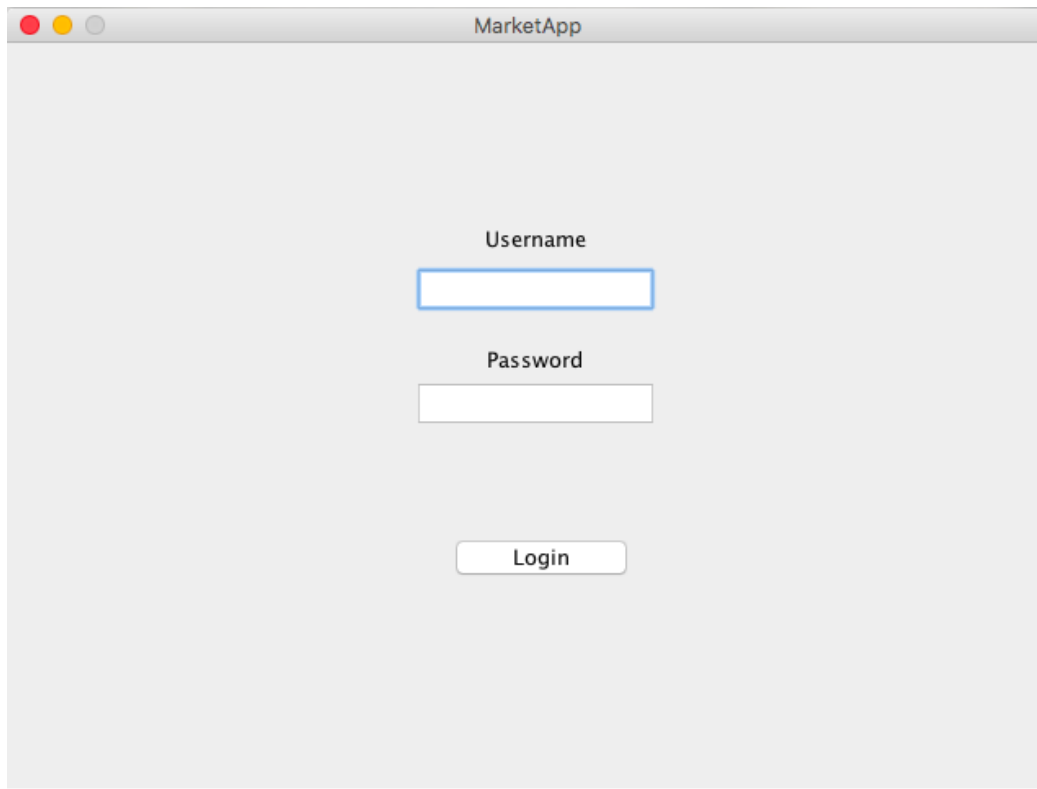
A `JLabel` object can display either text, an image, or both. You can specify where in the label's display area the label's contents are aligned by setting the vertical and horizontal alignment. By default, labels are vertically centered in their display area. Text-only labels are leading edge aligned, by default; image-only labels are horizontally centered, by default.

- d) `javax.swing.JPanel`: The `JPanel` class provides general-purpose containers for lightweight components. By default, panels do not add colors to anything except their own background; however, you can easily add borders to them and otherwise customize their painting.

In many types of look and feel, panels are opaque by default. Opaque panels work well as content panes and can help with painting efficiently, as described in Using Top-Level Containers. You can change a panel's transparency by invoking the `setOpaque` method. A transparent panel draws no background, so that any components underneath show through.

- e) `javax.swing.JTable`: The `JTable` is used to display and edit regular two-dimensional tables of cells. The `JTable` has many facilities that make it possible to customize its rendering and editing but provides defaults for these features so that simple tables can be set up easily.

4. User Interface



Login page (Panel 1)

Application Window

When running the application, the “BankApp” window will open and it will provide to the user the possibility of giving inputs and choosing the operations to be executed. This window is constructed in the GUI class using some predefined classes and instructions.

The user interface is based on the properties of the above mentioned packages. All the objects we need are declared as attributes of the GUI class and they are initialized in the constructor of this class.

The listener interface is for receiving action events. The class that is interested in processing an action event implements this interface, and the object created with that class is registered with a component, using the component's `addActionListener` method. When the action event occurs, that object's `actionPerformed` method is invoked. In this case the only events that occur are when the user clicks on one of the operation buttons from the graphical interface.

5. Conclusions

Developing this application was good challenge. It was a very good practice as a laboratory assignment. Of course, many features could be implemented in future versions, such as providing the user with the possibility of creating a new account, changing the data structure with an SQL database, add graphic images to improve esthetics, etc.

6. Bibliography

- <https://docs.oracle.com/>
- <http://stackoverflow.com>
- <https://wiki.eclipse.org>