<rentIT>

Analysis and Design Document

Student: Rus Rares

**Group: 30432**

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 04.04.2021 | 1.0 | Analysis and Design | Rus Rares |
| 24.04.2021 | 1.1 | Analysis and Design | Rus Rares |
| <28/04/2021> | 2.0 | Finals | <Rus Rares> |
| <26/05/2021> | 2.1 | Finals | <Rus Rares> |

Table of Contents

I. Project Specification 4

II. Elaboration – Iteration 1.1 4

1. Domain Model 4

2. Architectural Design 4

2.1 Conceptual Architecture 4

2.2 Package Design 4

2.3 Component and Deployment Diagrams 4

III. Elaboration – Iteration 1.2 4

1. Design Model 4

1.1 Dynamic Behavior 4

1.2 Class Design 4

2. Data Model 4

3. Unit Testing 4

IV. Elaboration – Iteration 2 4

1. Architectural Design Refinement 4

2. Design Model Refinement 4

V. Construction and Transition 5

1. System Testing 5

2. Future improvements 5

VI. Bibliography 5

# Project Specification

*I’m using Java to design to implement a desktop application for renting appartments. The application will have two types of users (a regular user represented and an administrator user) which have to provide a username and a password in order to use the application.*

*The regular user can perform the following operations:*

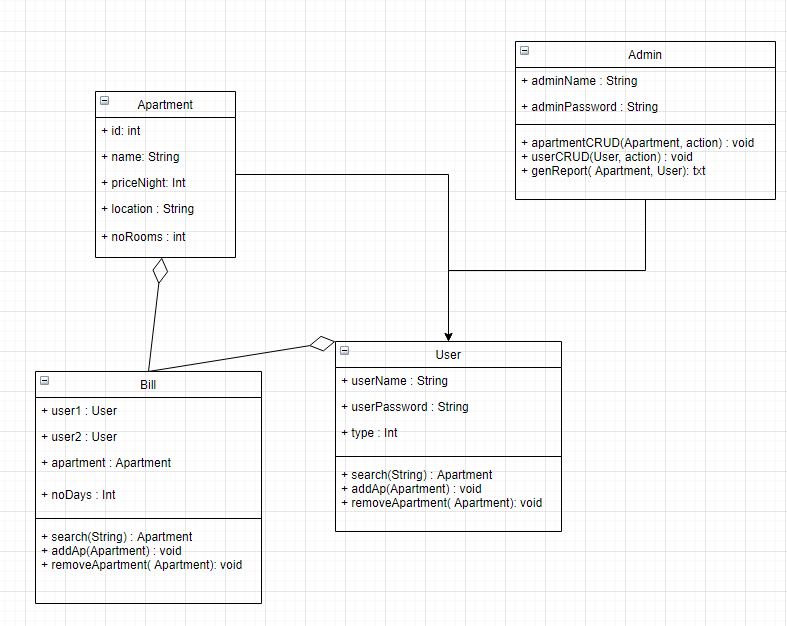
* *Search apartments to rent.*
* *Put apartments for renting.*

*The administrator can perform the following operations:*

* *CRUD on apartments.*
* *CRUD on regular users' information.*
* *Generate two types of report files, one in pdf format and one in txt or html format, with the history of a specific user. The reports need to be saved in a user-selected location (not predefined by the application), similar on how one would save a file from Notepad.*

# Elaboration – Iteration 1.1

# Domain Model



# Architectural Design

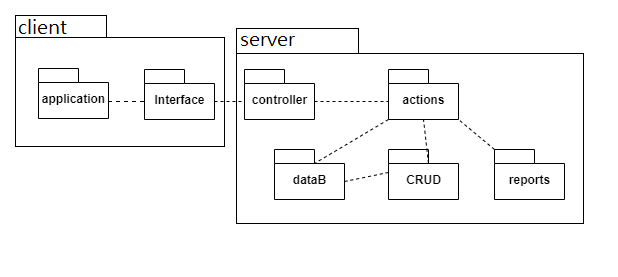
## Conceptual Architecture

The data used in the application will be stored in a database, using an ORM (Object Relational Mapping) framework.

Use the Model-View-Controller (or Model-View-ViewModel if using WPF) in designing the application. It's mandatory that the Controllers/ViewModels don't have direct reference to the View class (e.g., the one that inherits from JFrame or JPanel), but through an interface if really needed.

Use the Factory Method design pattern for generating the reports (i.e., selecting between the type of report to generate).

## Package Design

**

## Component and Deployment Diagrams

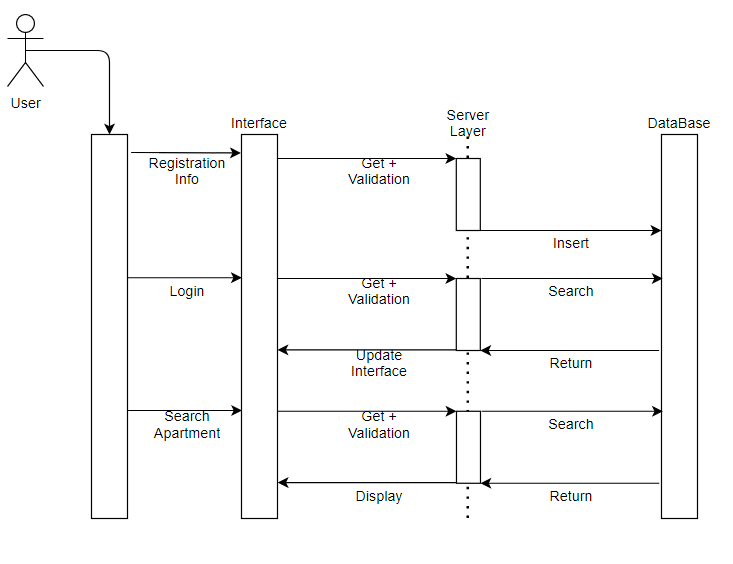
# 

# Elaboration – Iteration 1.2

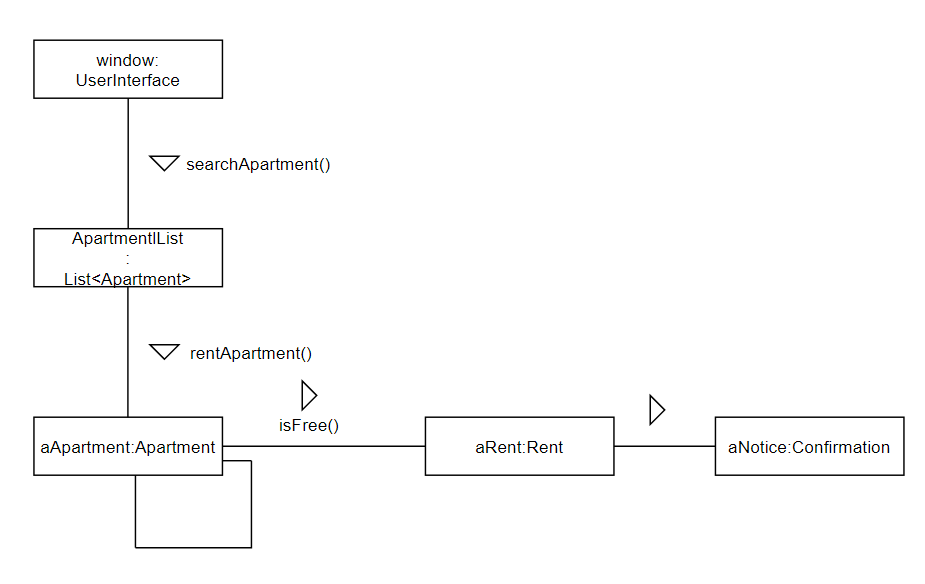
# Design Model

## Dynamic Behavior

Sequence Diagram

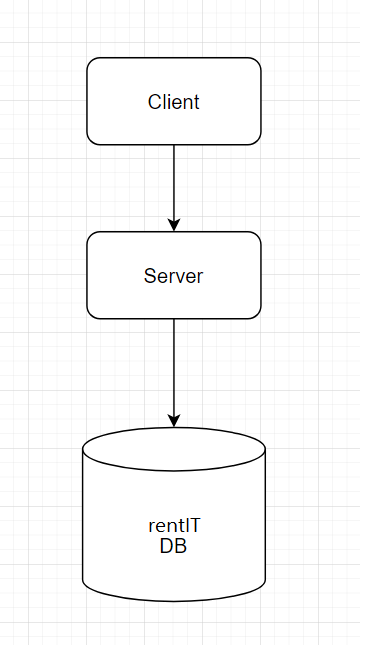


Communication Diagram



## Class Design

# Data Model

**

# Unit Testing

*[Present the used testing methods and the associated test case scenarios.]*

# Elaboration – Iteration 2

# Architectural Design Refinement

*[Refine the architectural design: conceptual architecture, package design (consider package design principles), component and deployment diagrams. Motivate the changes that have been made.]*

# Design Model Refinement

## *[Refine the UML class diagram by applying class design principles and GRASP; motivate your choices. Deliver the updated class diagrams.]*

# Construction and Transition

# System Testing

The next step will be to generate the test cases, based on the use-cases but only after integrating the units.

Test cases examples:

* 1. Login - the user enters the login details; if he gets something wrong, he/she will be notified, else he will continue to the next page.
  2. Create regular user - only admin can create one and only if he is logged in. If he can’t create it, he will be notified.
  3. Search apartment - if the apartment doesn’t exist, he will be notified.
  4. Create apartment listing - if the chosen song doesn’t exist an error will be displayed. A apartment cannot be inserted twice.
  5. Generate a report – can be done by an admin; the desired consultation is selected, the “Generate Report” (as .pdf or .txt) is pressed, and a popup menu for selecting the path for the new file will become available; the file will be created with the corresponding consultation details
  6. Delete apartment listing.

# Future improvements

There is room for many improvements such as sorting apartments based on location or type or price. The app could recommend apartments made by an algorithm. The admin could also create recommended playlists.

# Bibliography

<https://www.youtube.com/watch?v=rk2zcyzeK3U>

<https://docs.jboss.org/hibernate/orm/current/quickstart/html_single/#_the_hibernate_modules_artifacts>

<https://docs.oracle.com/en/java/javase/11/docs/api/java.base/java/util/List.html>