Student: Vescan Catalin

**Group: 30235**

Table of Contents

1. Requirements Analysis 3

1.1 Assignment Specification 3

1.2 Functional Requirements 3

1.3 Non-functional Requirements 3

2. Use-Case Model 3

3. System Architectural Design 3

4. UML Sequence Diagrams 7

5. Class Design 8

6. Data Model 9

7. System Testing 9

8. Bibliography 9

1. Requirements Analysis

# Assignment Specification

Aplicatia a fost implementata pentru o agentie de turism. Aceasta aplicatie suporta 2 tipuri de utilizatori, si anume 1 agent sau 1 administrator (la un momentdat), care au nevoie de un nume de utilizator si o parola pentru a se autentifica si pentru a putea folosi aplicatia.

Agentul poate:

* sa adauge, sa vada sau sa modifice informatiile corespunzatoare unui client;
* sa adauge, sa vada, sa modifice sau sa stearga informatiile corespunzatoare rezervarii unei vacante asociate unui client.

Administratorul poate:

* sa adauge, sa vada, sa modifice sau sa stearga informatiile corespunzatoare unui agent;
* sa genereze rapoarte.

# Functional Requirements

Aplicatia imi ofera posibilitatea de a stoca informatiile legate de administrator, agenti, clienti si vacantele clientilor. In cazul in care agentul sau administratorul introduc date gresite acestia vor fi avertizati printr-un mesaj de eroare, cum de asemenea atunci cand efectueaza o operatie care are success vor fi instiintati printr-un mesaj informative.

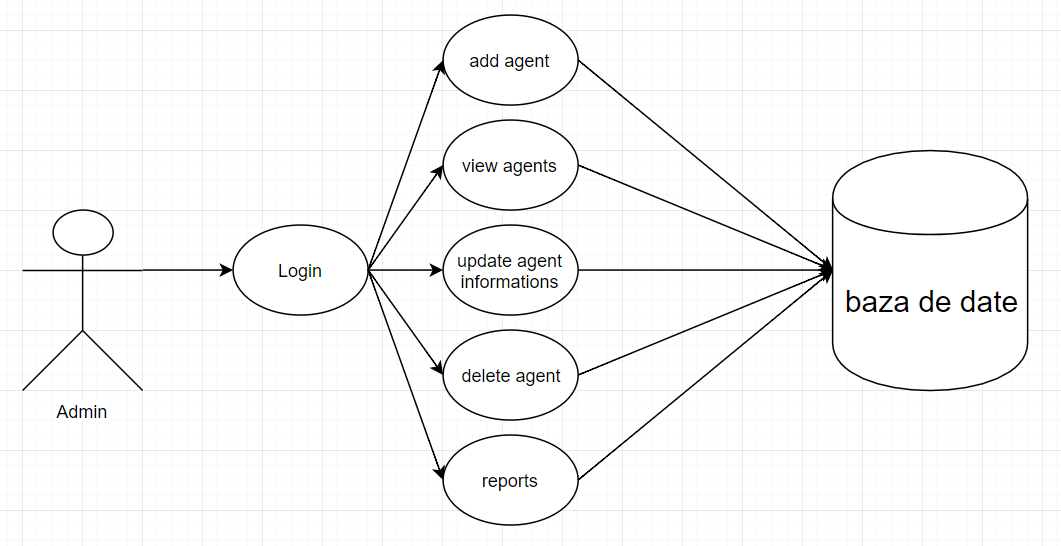
In situatia prezentata in continuare, este logat in aplicatie un agent. Acesta apasa butonul ”add client”, care ii va deschide o interfata unde agentul trebuie sa introduca datele, care la apasarea butonului ”OK” acestea vor fi introduse cu success in baza de date.

# Non-functional Requirements

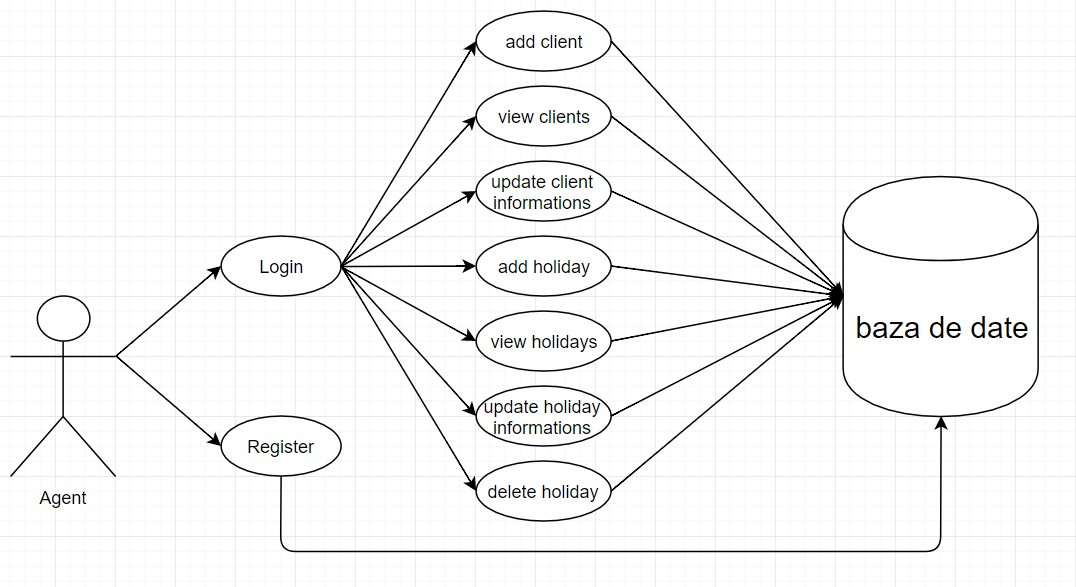
Aplicatia ruleaza doar pe desktop, indiferent de sistemul de operare, raspunde la orice actiune in mai putin de 5 secunde, si suporta 1 singur utilizator la un momentdat, iar datele sunt securizate prin back-up odata la 1 ora, timp care se poate modifica.

2. Use-Case Model

Usecase Admin



Usecase Agent



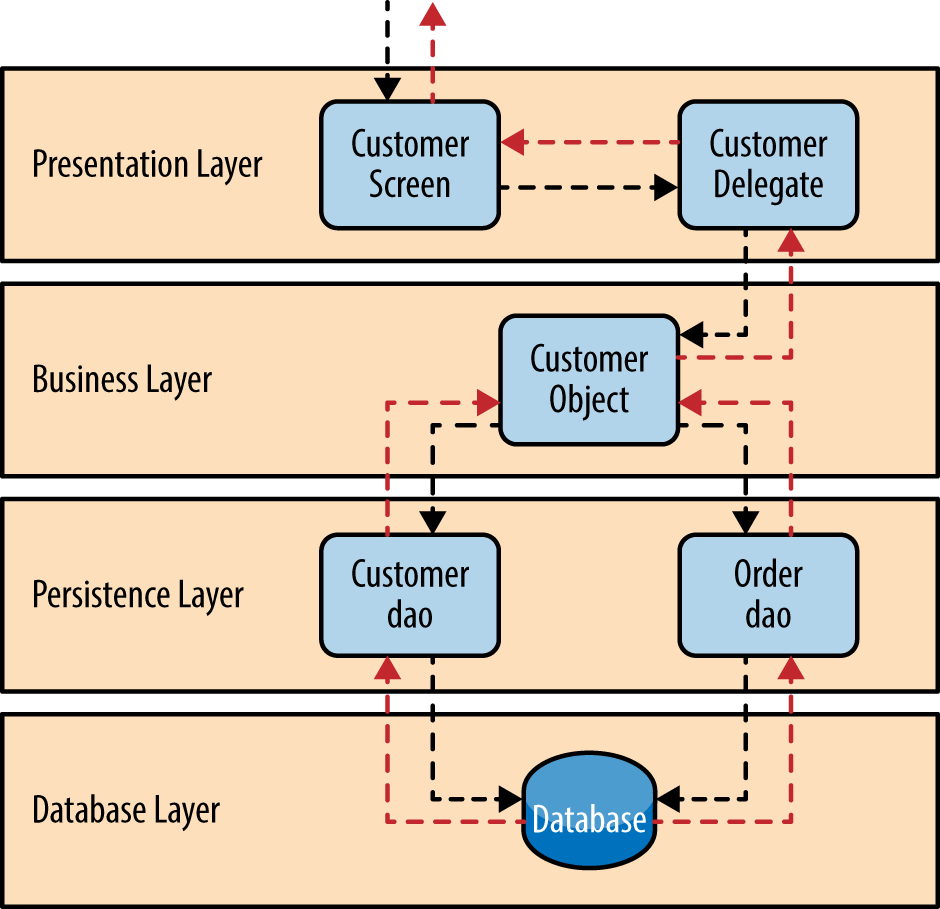
Actorul principal in urmatorul exemplu este un agent dintr-o firma de turism. Un posibil scenariu cu pasii necesari este urmatorul:

In fereastra de login, agentul isi introduce datele pentru autentificare (username si password), selecteaza optiunea ”Agent”, dupa care apasa butonul ”Login”. Odata autentificat, acesta alege sa faca update la un client. Clientii se identifica dupa un cod numeric personal care este unic, iar agentul nestiind codurile clientilor, acesta va apasa butonul ”view clients” care ii va afisa o fereastra cu toti clientii si datele acestora, de unde el va selecta codul numeric personal specific clientului a carui date doreste sa le modifice, iar apoi va apasa butonul ”update client”, iar datele se vor modifica in baza de date.

3. System Architectural Design

**3.1 Architectural Pattern Description**

In aceasta aplicatie am folosit 3 Layers Arhitectural Pattern, care este impartit in 4 mari categorii si anume nivelul de interfata grafica cu utilizatorul, nivelul de business, nivelul de persistenta a datelor, si nivelul bazei de date.



**3.2 Diagrams**

Diagrama de pachete

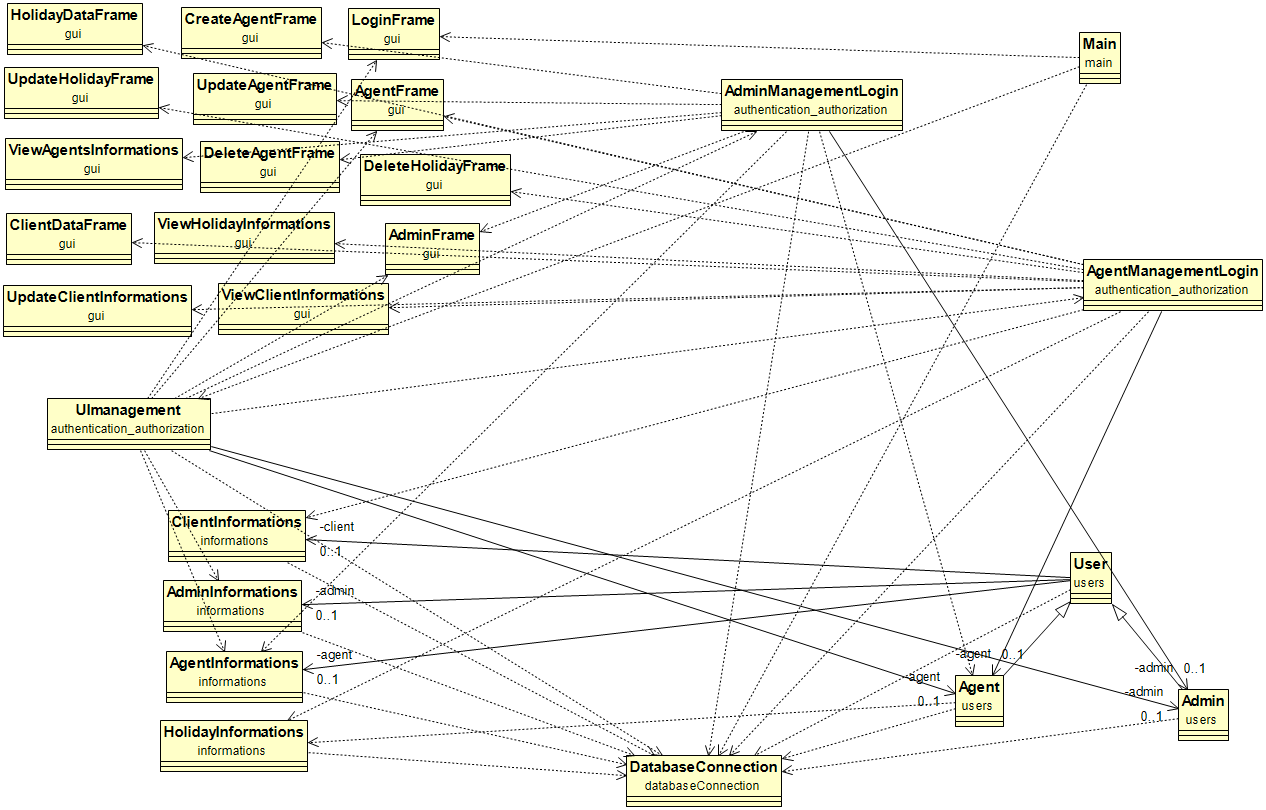
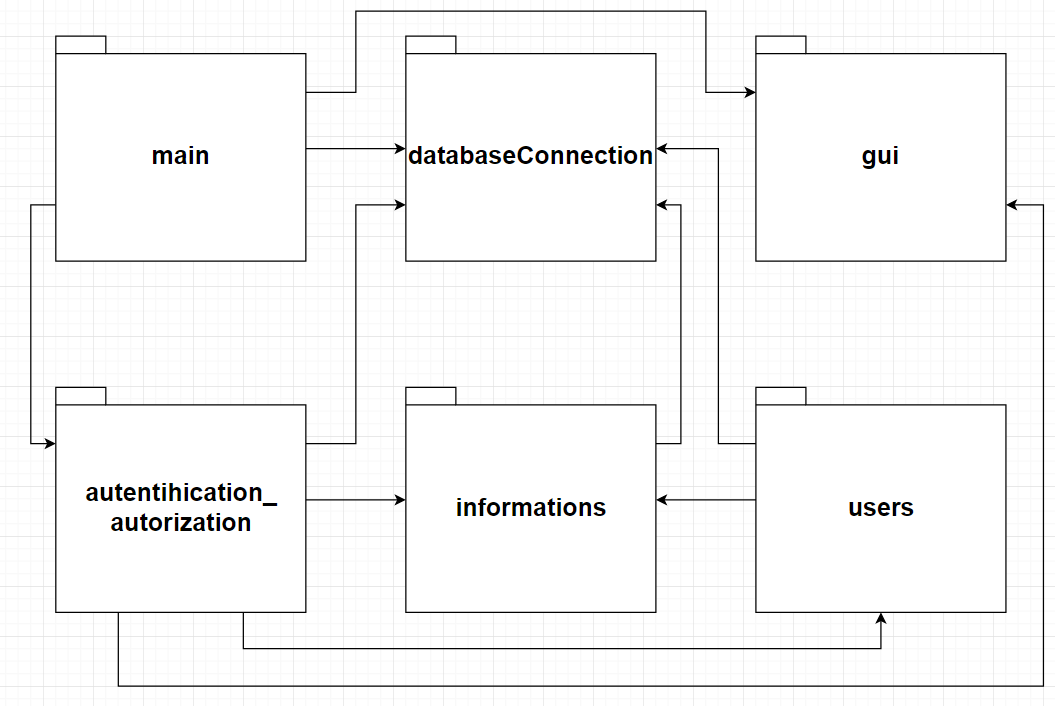
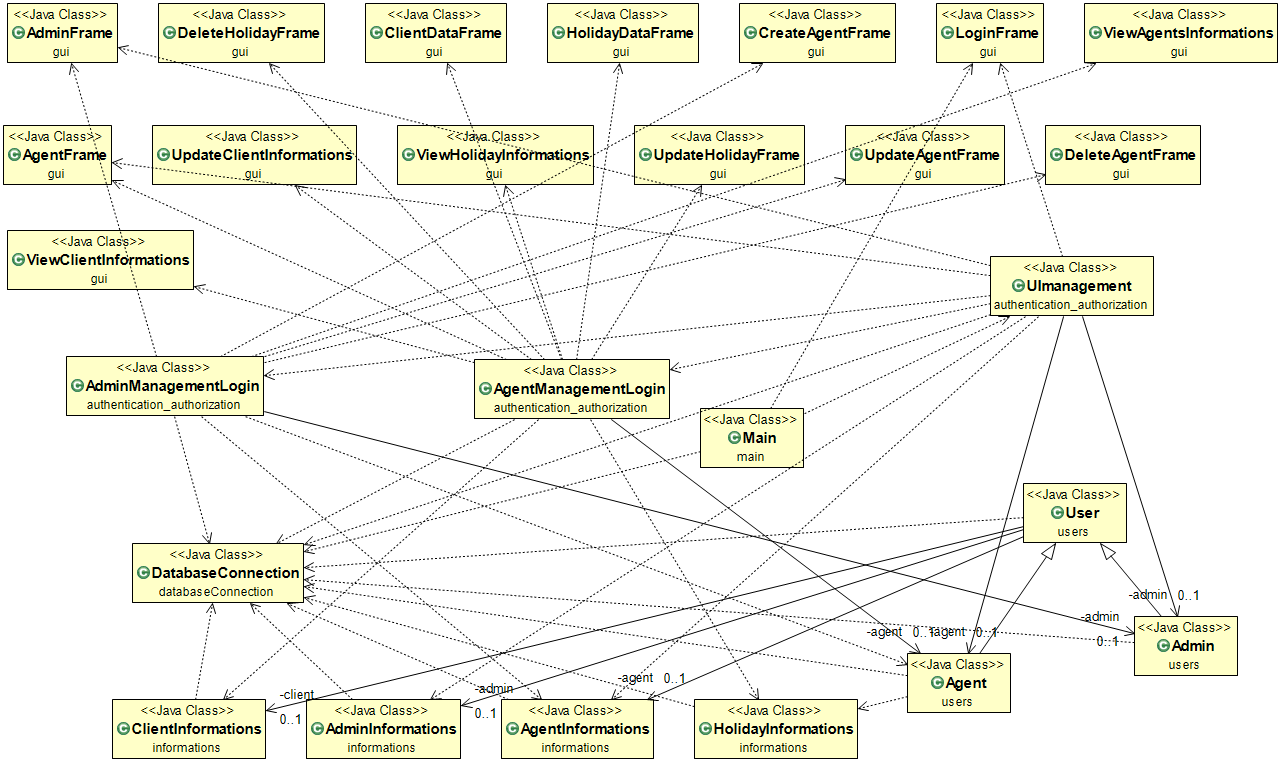
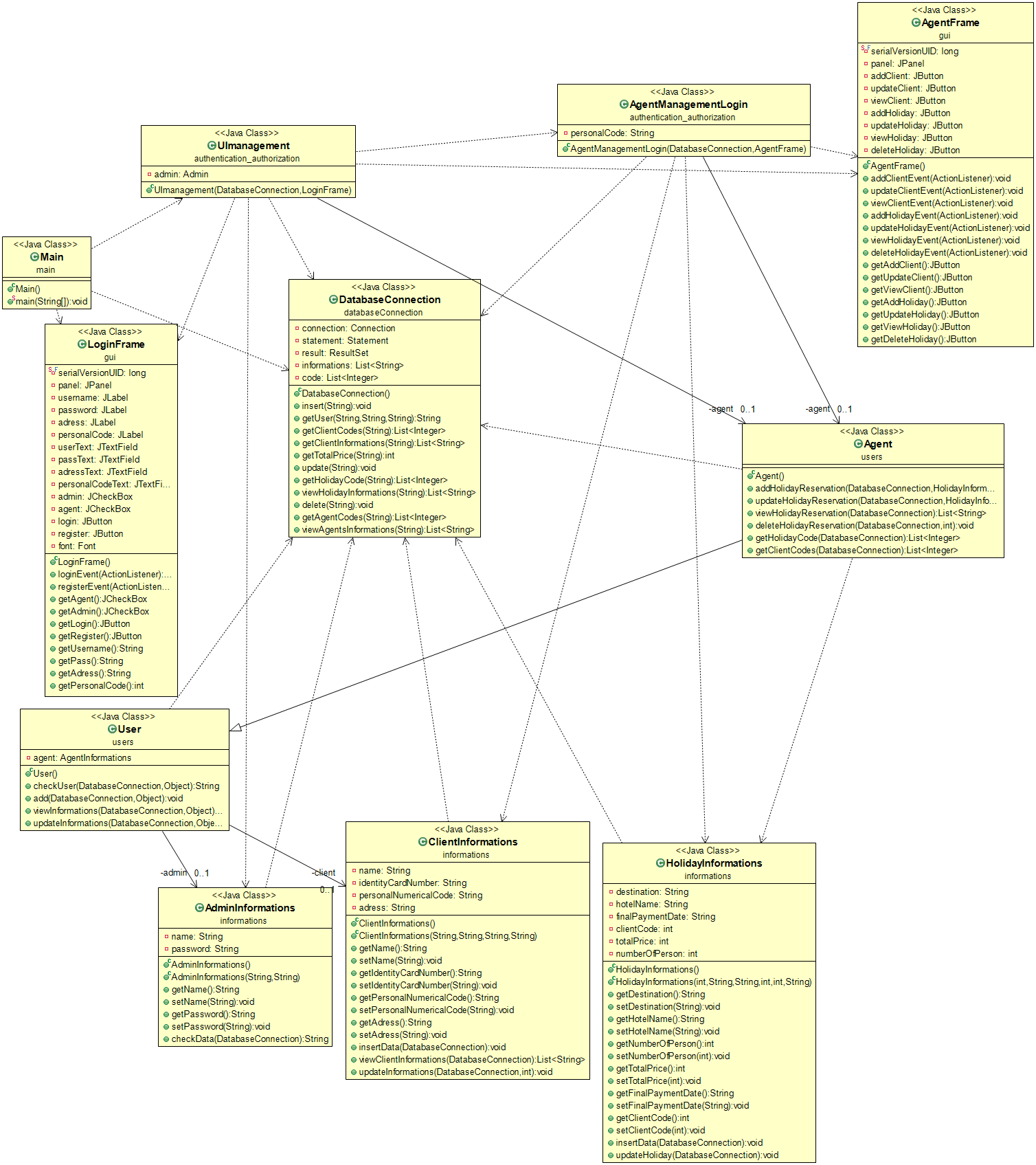
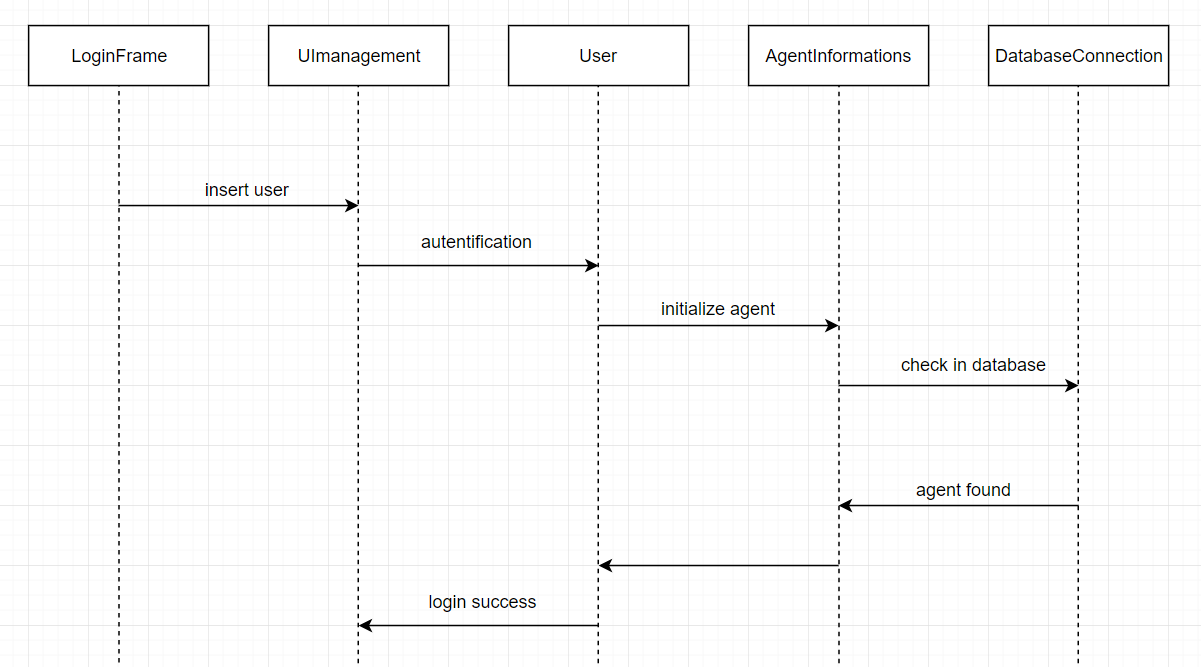


Diagrama de clase



4. UML Sequence Diagrams

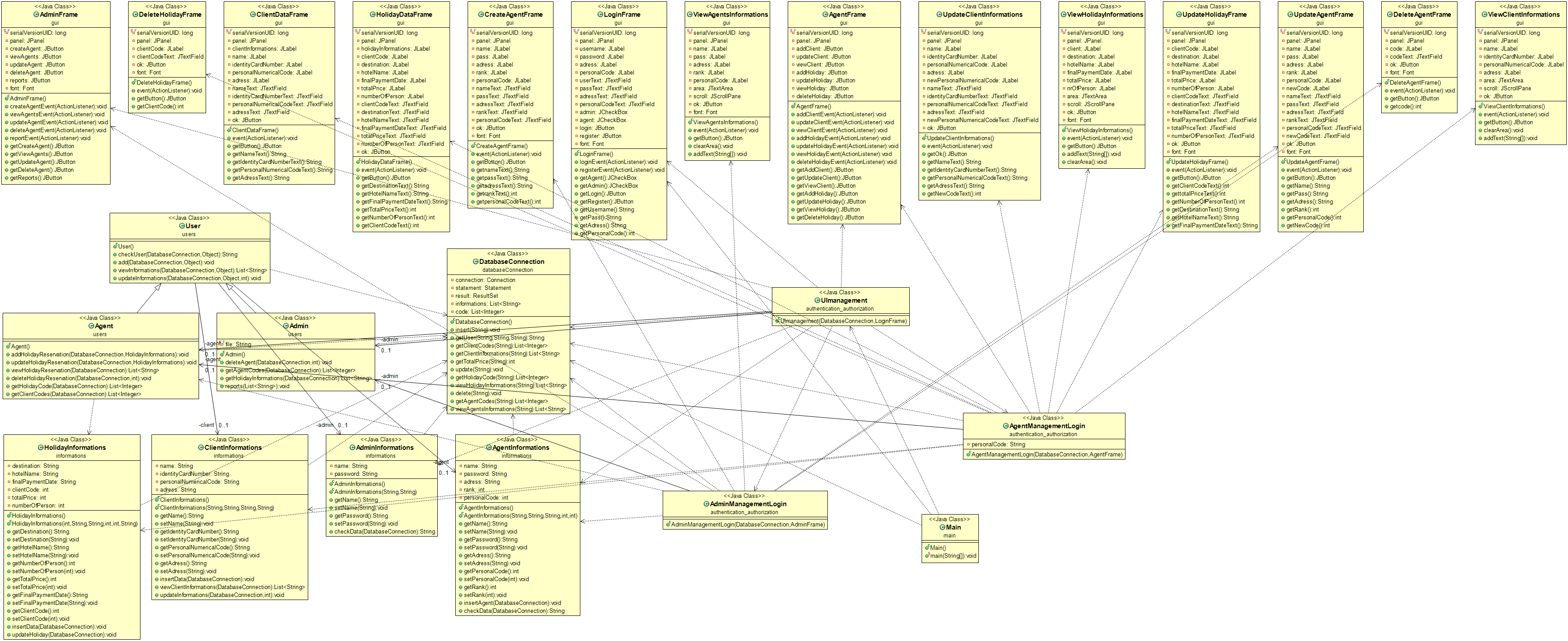


5. Class Design

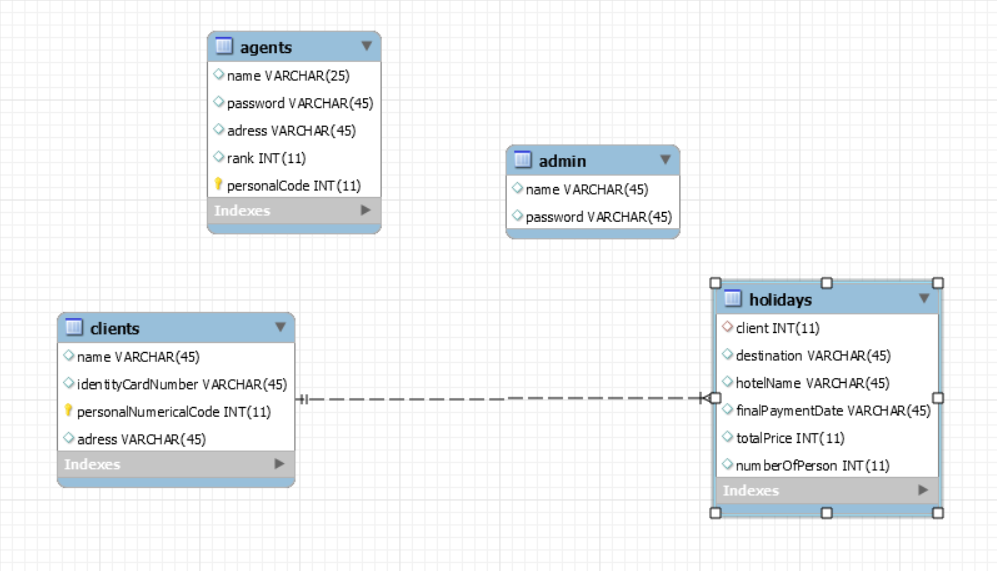
**5.1 Design Patterns Description**

In aceasta aplicatie am folosit ca arhitectura 3 layers arhtiecture, iar ca design pattern domain model deoarece clasele care stocheaza informatiile pot executa si alte operatii, si row data gateway pure pattern deoarece mapez variabilele din obiect in functie de coloanele tabelelor din baza de date.

**5.2 UML Class Diagram**



6. Data Model



7. System Testing

Pentru testarea metodelor folosite in aplicatie am folosit JUnitTest.

8. Bibliography

<https://www.google.ro/search?q=3+layer+architecture+pattern&rlz=1C1AVFC_enRO781RO781&source=lnms&tbm=isch&sa=X&ved=0ahUKEwirxLW915jaAhXIsKQKHfxtCwcQ_AUICigB&biw=1536&bih=759#imgrc=4QFudFrewxcXvM>:

<https://en.wikipedia.org/wiki/Multitier_architecture#Three-tier_architecture>

<https://en.wikipedia.org/wiki/Domain_model>

<https://en.wikipedia.org/wiki/Row_data_gateway>