Student:Tomoiaga Anamaria

**Group:30235**

Table of Contents

[1. Requirements Analysis 3](#_Toc510785012)

[1.1 Assignment Specification 3](#_Toc510785013)

[1.2 Functional Requirements 3](#_Toc510785014)

[1.3 Non-functional Requirements 3](#_Toc510785015)

[2. Use-Case Model 4](#_Toc510785016)

[3. System Architectural Design 4](#_Toc510785017)

[3.1 Architectural Pattern Description 4](#_Toc510785018)

[3.2 Diagrams 5](#_Toc510785019)

[4. UML Sequence Diagrams 6](#_Toc510785020)

[5. Class Design 7](#_Toc510785021)

[6. Data Model 7](#_Toc510785022)

[7. System Testing 8](#_Toc510785023)

[8. Bibliography 8](#_Toc510785024)

# 1. Requirements Analysis

## Assignment Specification

Utilizati JAVA/C# API pentru a implementa o aplicatie pentru agentii unei agentii de turism. Aplicatia are doua tipuri de useri (un user reprezentat de un agent de turism si un administrator) care vor folosi un username si o parola pentru a putea folosi aplicatia.

Userul poate realiza urmatoarele operatii:

* Adaugare, actualizare, vizualizare informatii client
* Adaugare, actualizare, vizualizare, stergere rezervare pentru un client
* Accepta plata partiala de la un client inainte de data finala de plata
* Vizualizeaza toti clientii care au ratat termenul final de plata si are posibilitatea de a anula rezervarea.

Administratorul poate realize urmatoarele operatii:

* CRUD pe informatiile agentului
* Generare de rapoarte pentru o perioada particulara care contine activitatile realizate de un agent.

## Functional Requirements

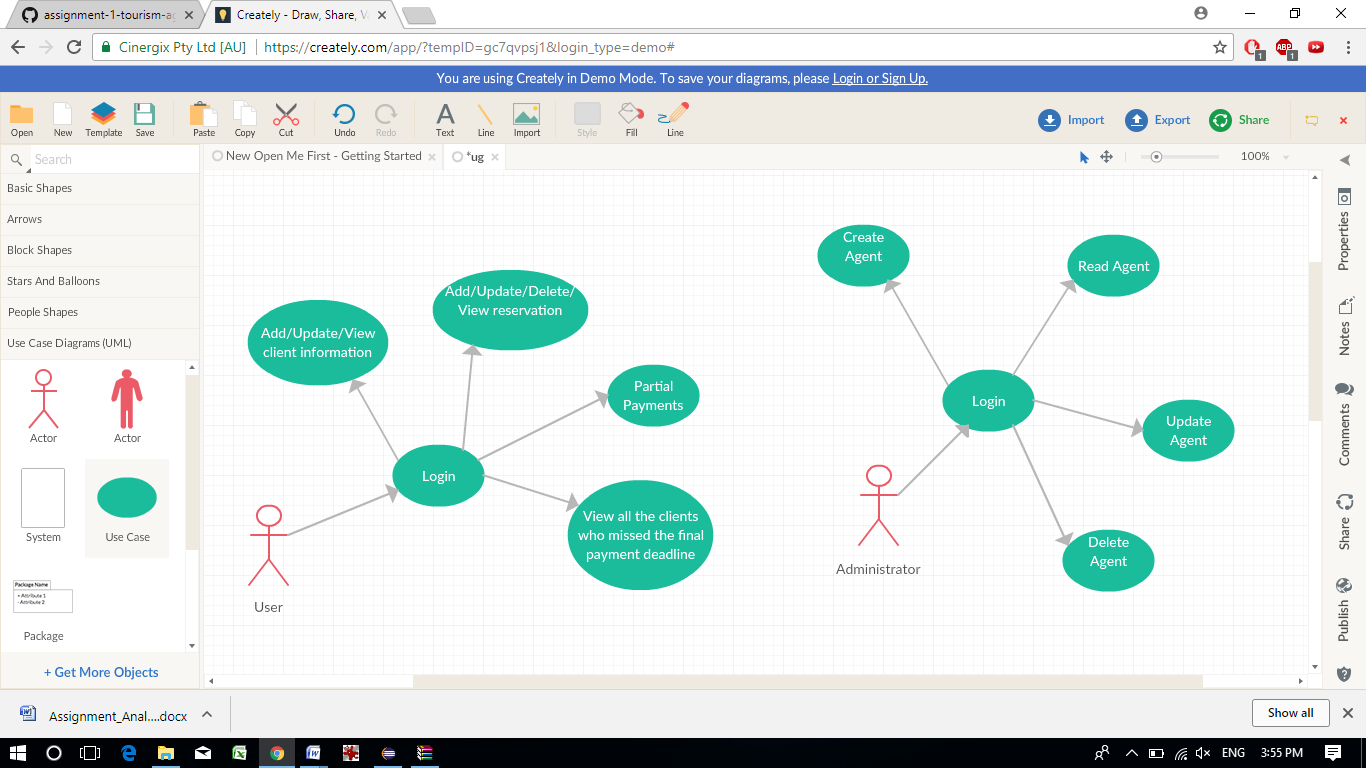
Administratorul nu se poate loga ca un user obisnuit.

## Non-functional Requirements

Securitate: sistemul este securizat prin cererea unui username si a unei parole pentru a putea utilize aplicatia.

Performanta: sistemul trebuie sa raspunda repede la cererile userului.

# Use-Case Model



Use case: <adaugare rezervare>

Level: <user goal level>

Primary actor: <utilizator regular>

Main success scenario: <Introducem toate informatiile despre rezervare, care vor fi salvate in baza de date>

Extensions: <Daca nu introducem toate informatiile despre rezervare vom primi un mesaj de eroare>

# System Architectural Design

## Architectural Pattern Description

Sistemul utilizeaza 3-tier architecture pattern, care imparte aplicatia in diferite layere. Fiecare layer are un scop anume si apeleaza functii din layerele de dupa el.

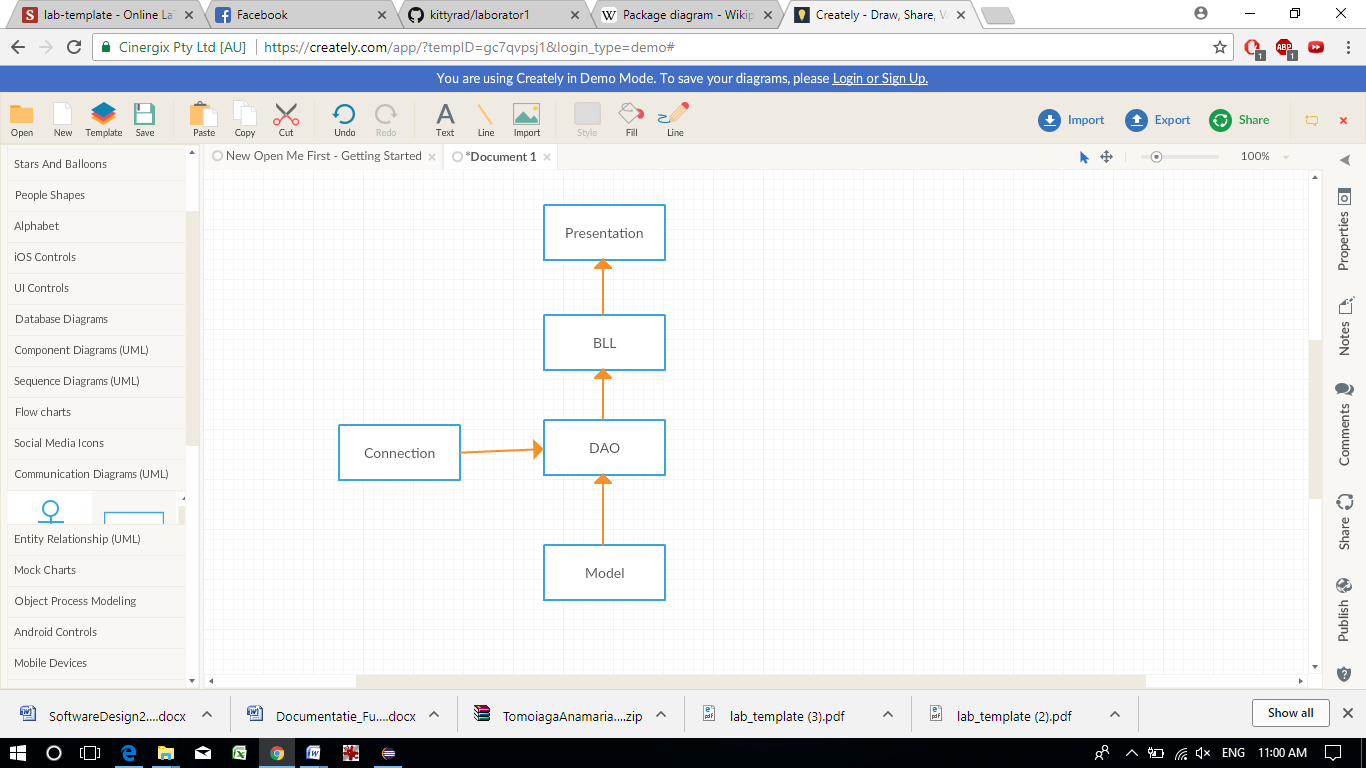
Pimul layer (Data Access Layer) cuprinde clasele care contin interogari si conectarea cu baza de date.

Al doilea layer (Business Logic) contine clasele care incapsuleaza logica aplicatiei.

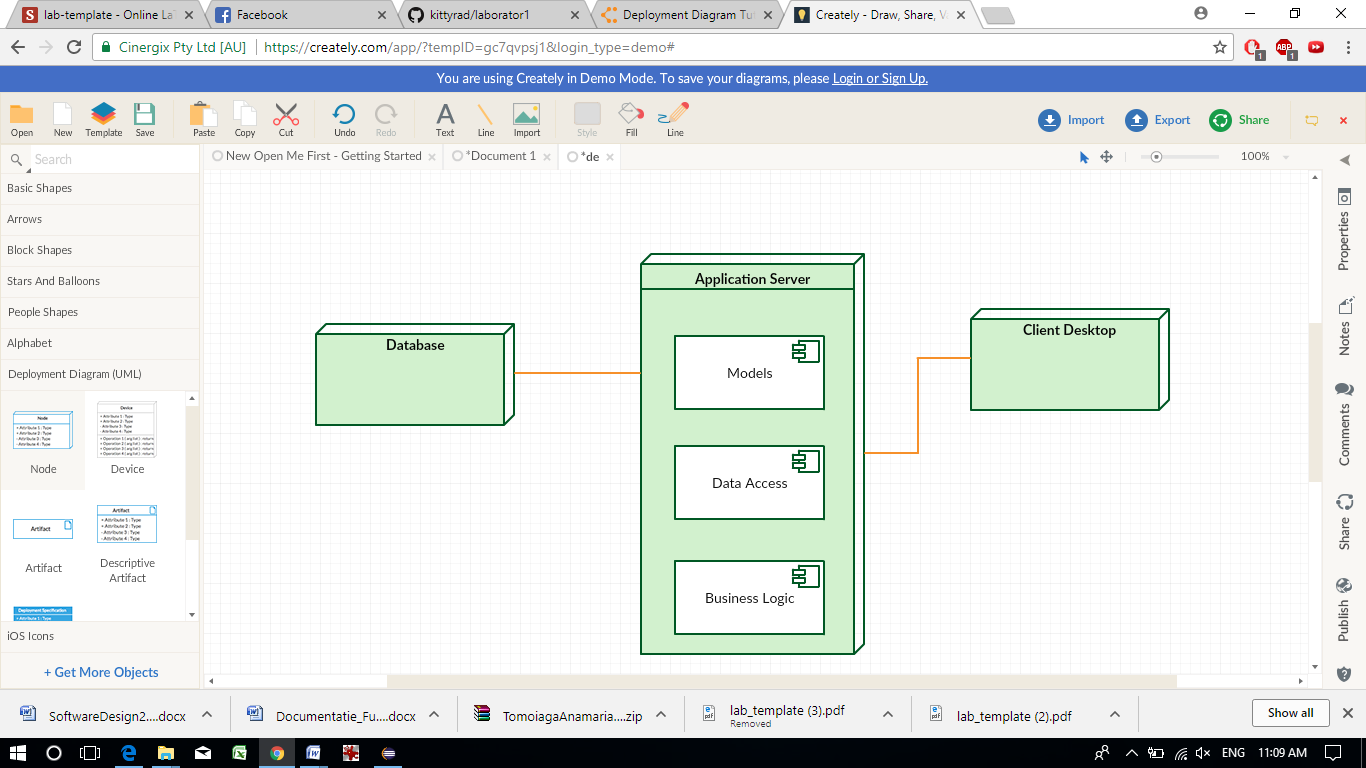
Al treilea layer (Presentation Layer) contine clasele care defines interfata cu utilizatorul.

## Diagrams

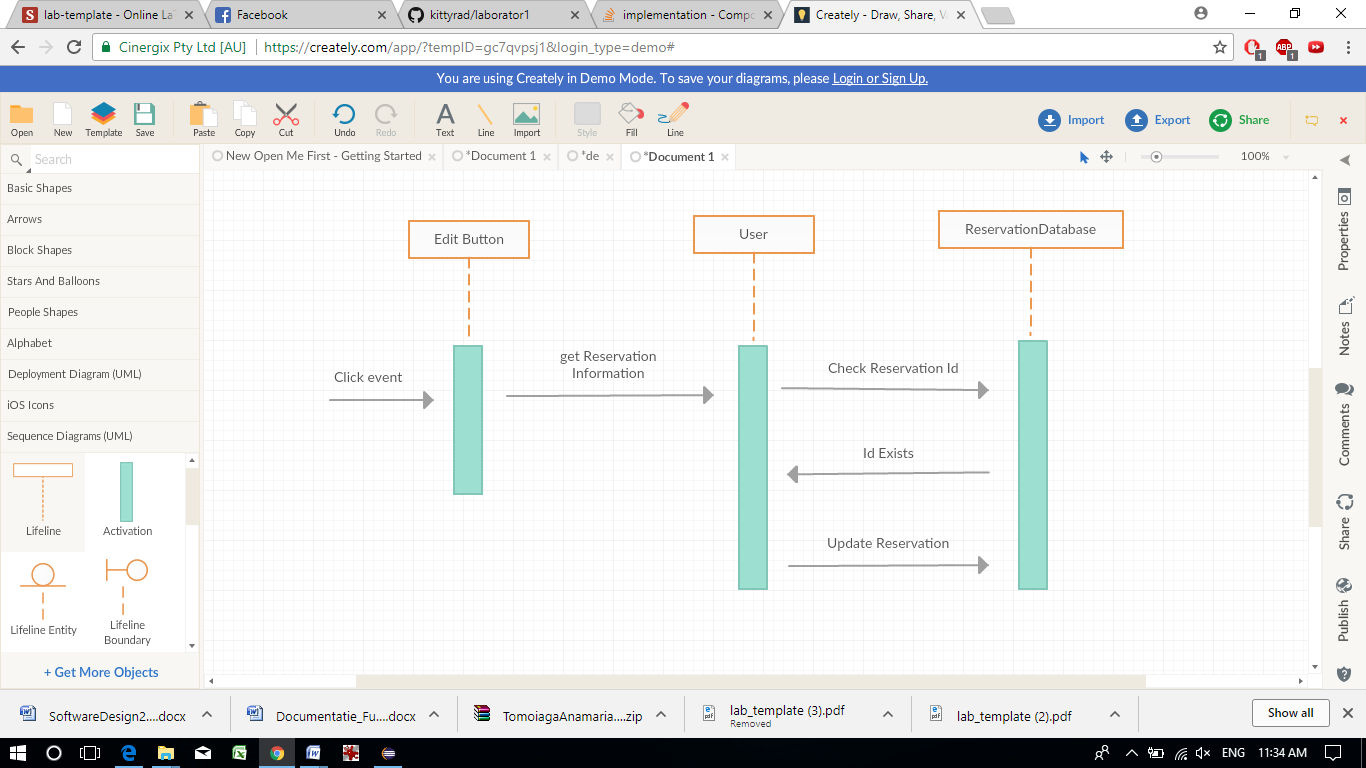
Package diagram



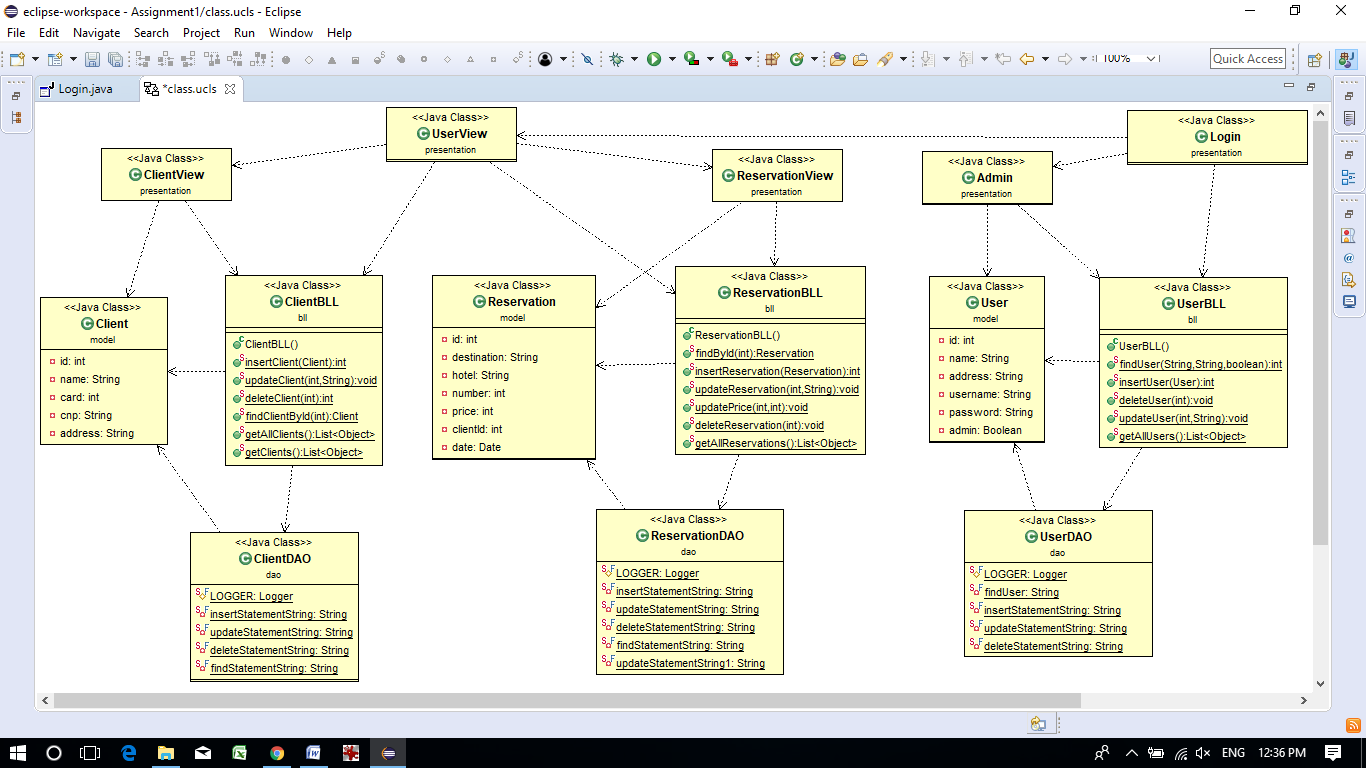
Deployment diagram



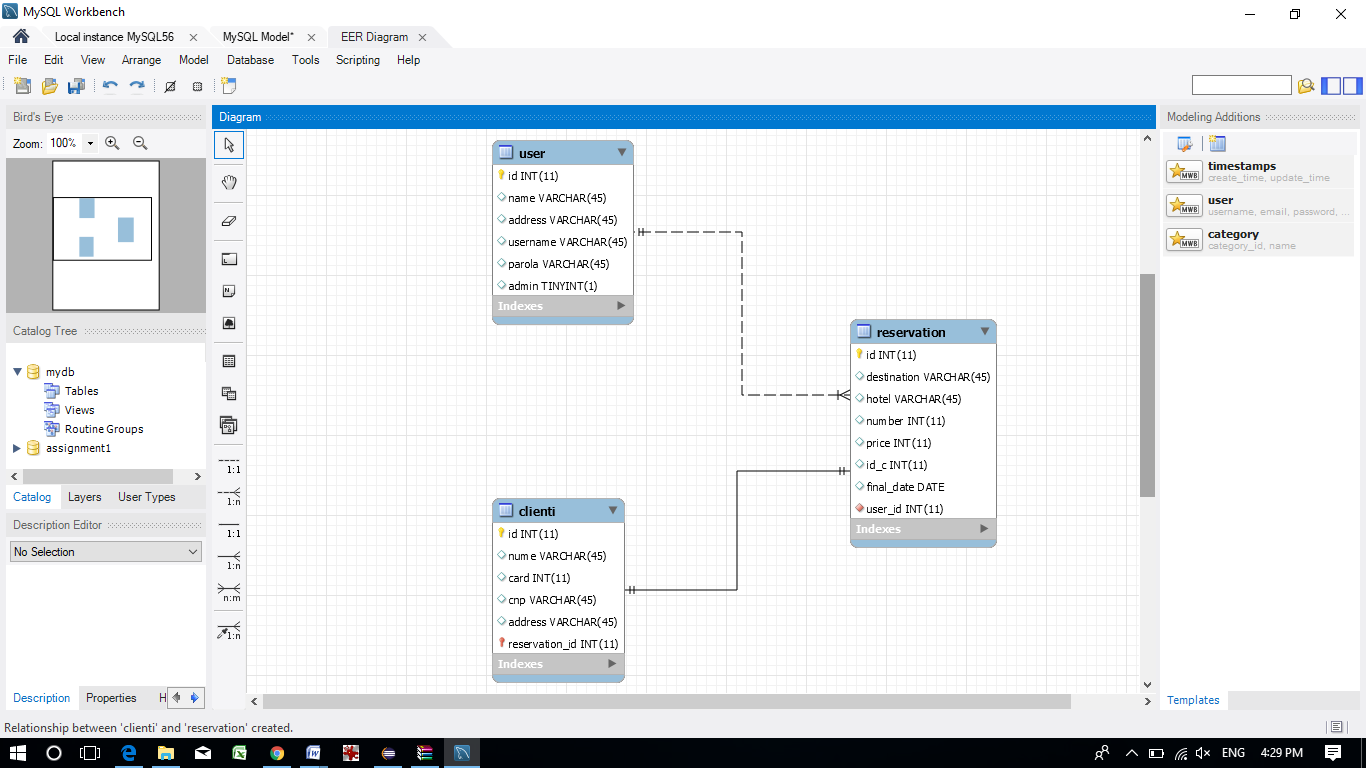
# UML Sequence Diagrams

**

# Class Design

**

# Data Model

**

# System Testing

Am verificat daca codul functioneaza prin realizarea de Unit Tests.

# Bibliography

https://martinfowler.com/bliki/PresentationDomainDataLayering.html