<Car Renting Application>

Analysis and Design Document

Student:Ionel Carmen

**Group:30237**

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| <03/04/2017> | <1.0> | <details> | <Ionel Carmen> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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 5

V. Construction and Transition 5

1. System Testing 5

2. Future improvements 5

VI. Bibliography 5

# Project Specification

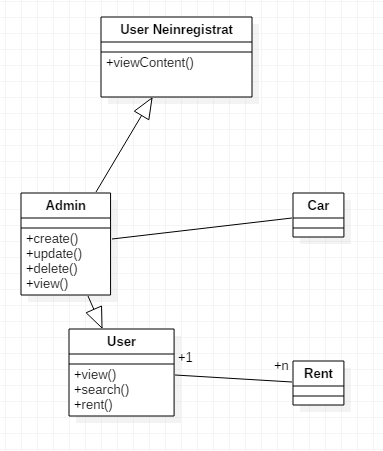
# Implementati o aplicatie client-server pentru a inchiria o masina Pony, Aceasta aplicatie poate fi utilizata atat de clienti cat si de un administrator. Aplicatia va permite unui client sa vizualize masinile disponibile(caracteristicile masinii,locatia),preturile percepute,dar si ofertele promotionale. Pentru a putea accesa masina, clientul trebuie sa isi completeze datele personale(date de identificare, de contact, datele contului si datele permisului auto). Dupa completarea datelor, clientul va primi de la administrator un cod unic, prin care va putea accesa masina.

# Aplicatia va avea operatii CRUD asupra clientilor si masinilor.

# Elaboration – Iteration 1.1

# Domain Model

Pentru descrierea modelului utilizam o diagrama de clase:



# Architectural Design

## Conceptual Architecture

Voi folosi pattern-ul architectural Layers, deoarece proiectul va contine o baza de date cu clienti si masini si va fi mai usor de accesat in acest fel.

Deasemenea voi utiliza patternul MVC , deoarece aplicatia me ava fi una web.

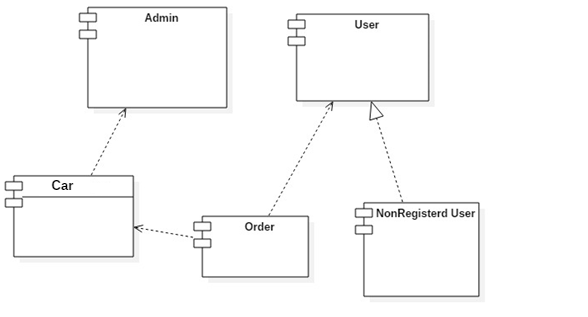
## Package Design

Arhitectura proiectului va fi Layers si in acest caz voi folosi Package By Layer. Asadar, proiectul va contine un pachet principal al aplicatiei, iar in acesta vor fi packetele Data access Layer, Bussiness Layer si Presentation Layer.

Arhitectura va contine sit rei pachete mari, care vor reprezenta Modelul, View-ul si Controller-ul.

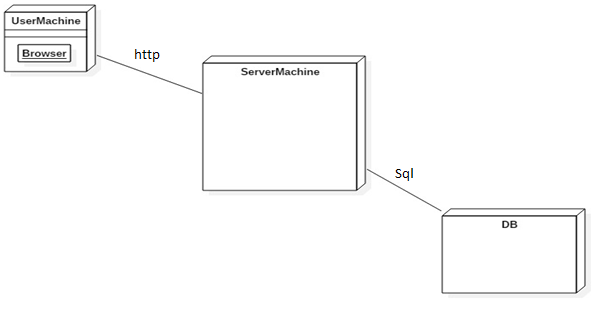
## Component Diagrams

Diagramele de componente sunt folosite pentru a modela aspectele fizice ale unui sistem, pentru a vizualiza organizarea și relațiile dintre componentele unui sistem. Aceste diagrame sunt de asemenea folosite pentru a face sisteme executabile.Aplicatia va avea trei tipuri de utilizatori ( user neinregistrat, inregistrat si adminul), comanda(rent) si masina.



**Deployment Diagram**

Diagramele de desfășurare sunt folosite pentru a vizualiza topologia componentelor fizice ale unui sistem.Sunt compuse din noduri și relațiile lor.



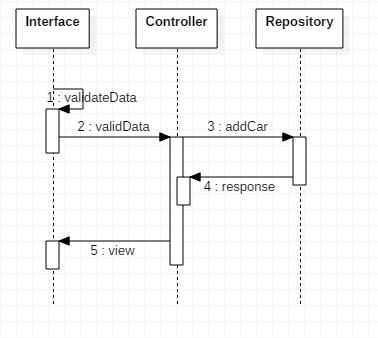
# Elaboration – Iteration 1.2

# Design Model

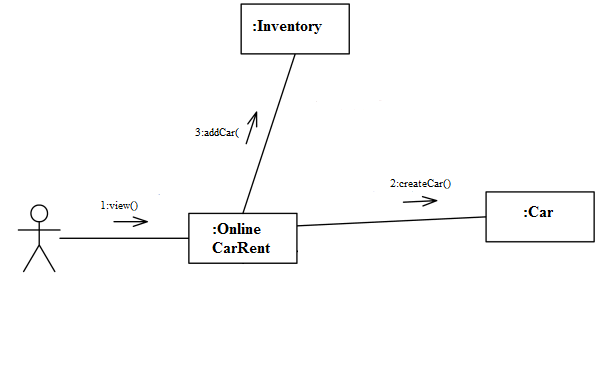
## Dynamic Behavior

Add Car

Sequence diagram:

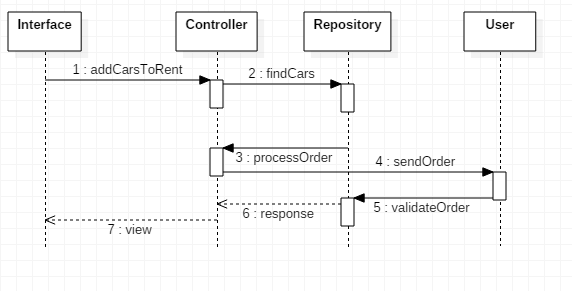


Communication diagram:

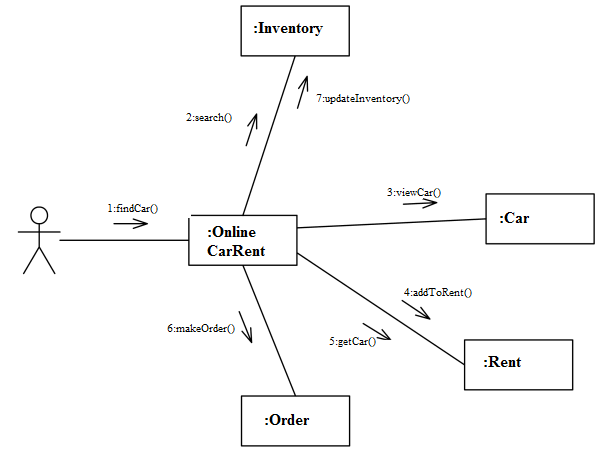


Add Cars to Rent

Sequence diagram:



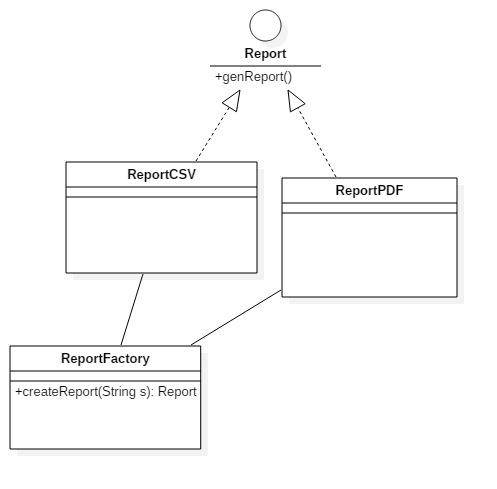
Communication diagram:

**

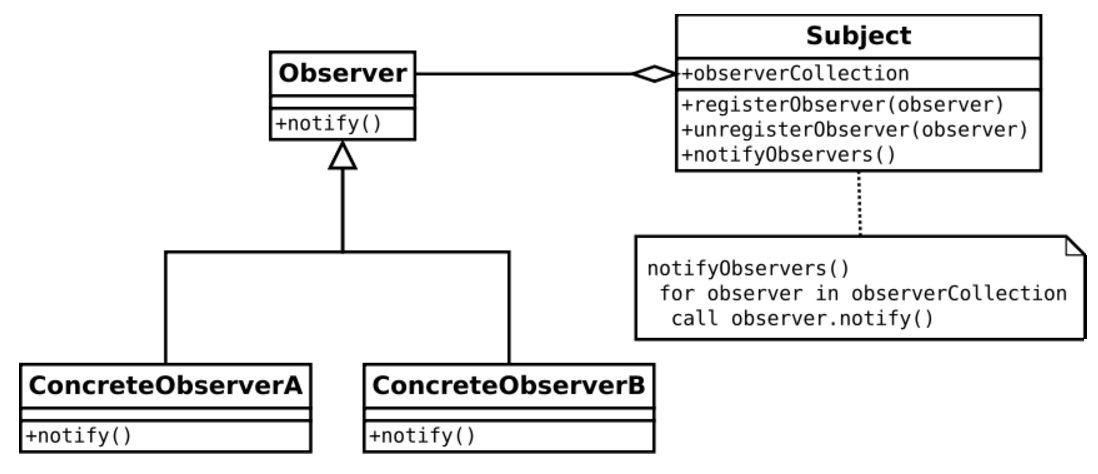
## Class Design

Utilizatorii cu rol de administrator vor putea genera rapoarte pe baza masinilor existente in baza de date pentru masinile care sunt inchiriate.

Pentru generarea rapoartelor voi folosi design pattern-ul Factory.



Deasemenea administratorul va fi notificat cand o persoana inchiriaza o masina. Pentru aceasta functionalitate voi utiliza desgn patternul Observer.



# Data Model

Baza de date va contine patru tabele: User (care va contine toti userii la care sunt adaugate roluri acestora), Cars ce va contine informatii legate de masini( nume, tip, numar de inmatriculare, descriere, pret), Rent (Date legate de user, masina imprumutata, numarul de zile pentru care s-a inchiriat, suma platita) si Cart o tabela unde se vor salva date temporare (masinile pe care clientul le adauga dar nu continua inchirierea).

# Unit Testing

Pentru a testa aplicatia se vor crea unit teste pe principalele operatii: CRUD pe Cars si CRUD pe utilizatori. De asemenea se vor face si teste din interfata, pentru a simnula toate cazurile posibile de interactiune cu utilizatorul, precum cautare masinilor, vederea locatiei acestora si inchirierea uneia, dar si cazuri pentru userii noi, pentru crearea contului si validarea acestuia.

# 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

*[Describe how you applied integration testing and present the associated test case scenarios.]*

# Future improvements

*[Present future improvements for the system]*

# Bibliography