YOUR PERFECT CAR

Analysis and Design Document

Student: Popovici Marius

**Group: 30238**

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| <dd/mmm/yy> | <x.x> | <details> | <name> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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

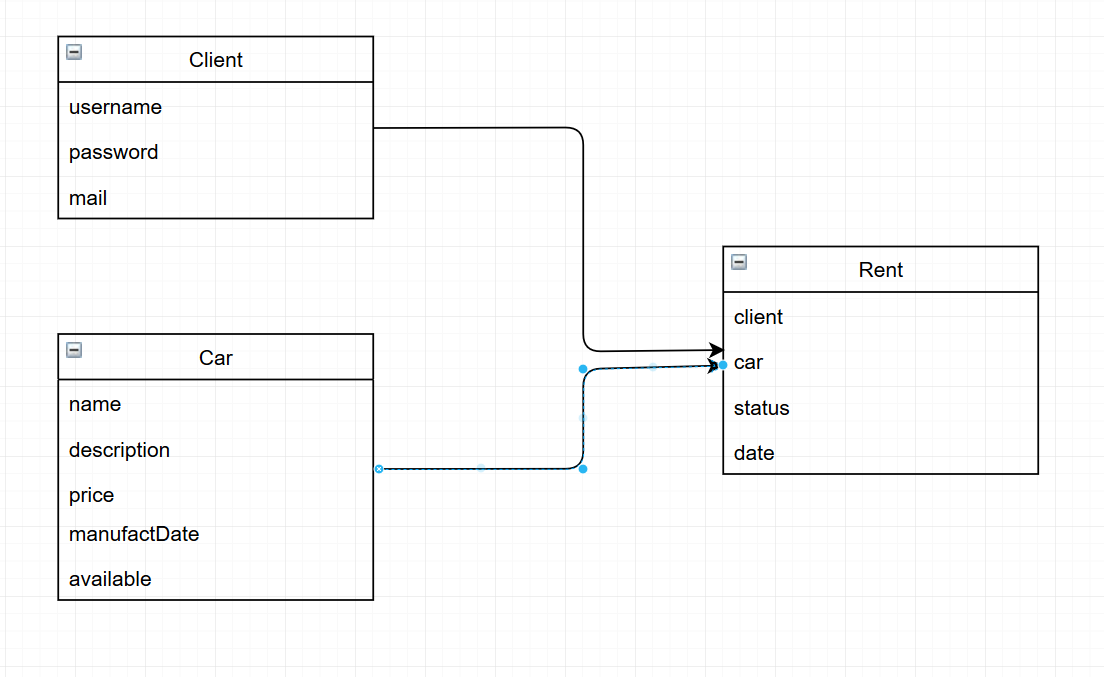
VI. Bibliography 5

# Project Specification

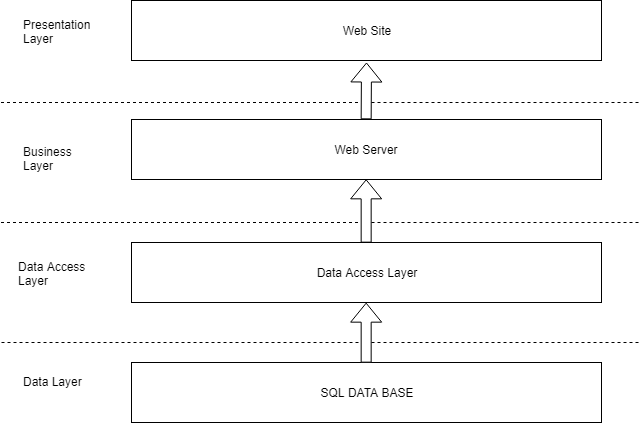
Proiectul reprezinta implementarea unei aplicatii pentru inchirieri de masini. Un client va avea posibilitatea de a crea un account si de a se loga pentru a putea inchiria o masina. Dupa logare, o lista cu masinile disponibile va fi prezentata clientului, avand posibilitatea de o alege pe cea pe care o prefera. Clientul va selecta perioada de imprumut si va alege daca doreste sa ridice personal masina (de la sediul firmei) sau sa ii fie livrata la o anumita adresa. De asemenea, un client va putea solicita un sofer, dar doar pentru intervalul 08:00-20:00. Taxa pentru sofer va fi platita la predarea masinii in functie de numarul de ore in care a condus/asteptat soferul.

# Elaboration – Iteration 1.1

# Domain Model

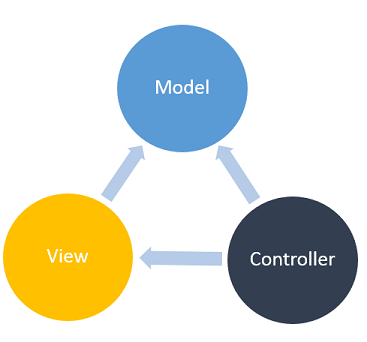


# Architectural Design

**

## Conceptual Architecture

Aplicatia este scris in Python si foloseste framework-ul Django, avand la baza pattern-ul MVC (Model-View-Controller).



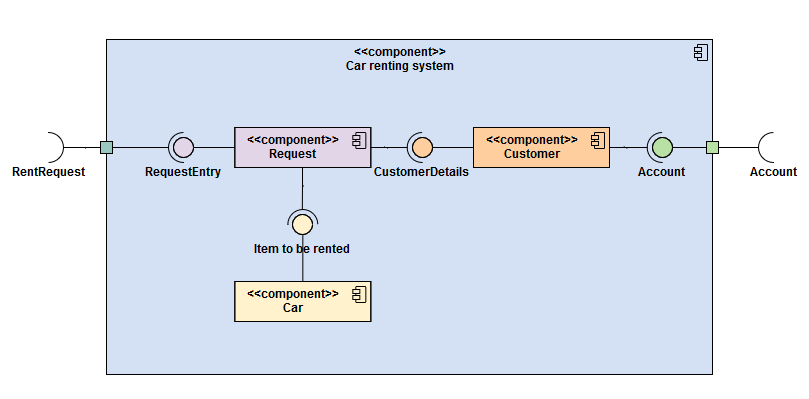
## Component and Deployment Diagrams

*[Create the component and deployment diagrams.]*

Deployment diagram:

Component diagram:

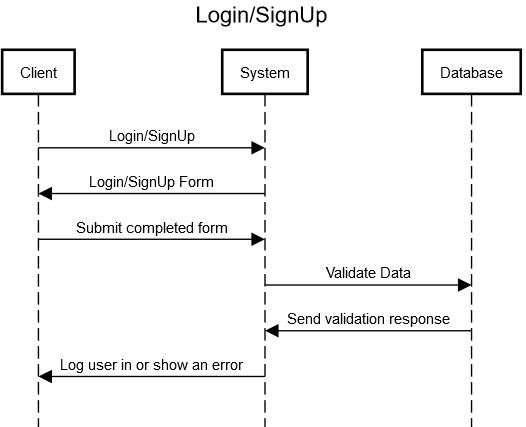
For renting operation:

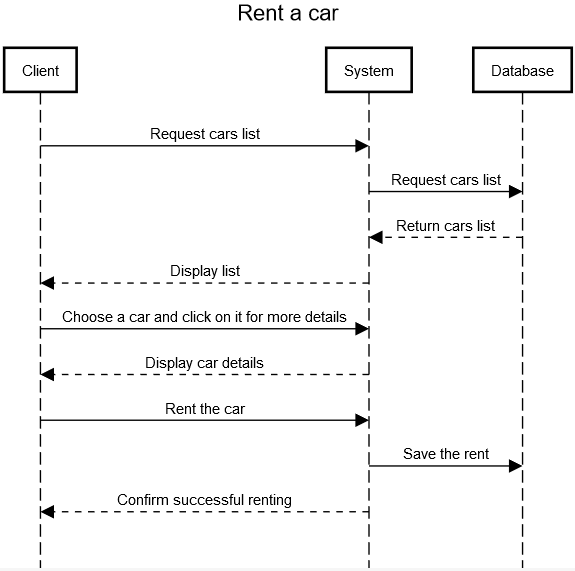


# Elaboration – Iteration 1.2

# Design Model

## Dynamic Behavior

**

**

## Class Design

The structure of the code is as follows: an “app” is created for every structure in the database. Example: accounts, cars, rents.

The main application is called “proiect”. All the other apps should be registered in this main app. It is the only class with a settings file.

An application contains some components as: urls, views, models, templates (for rendering html templates).

The urls file contains the reference to the implementation of the function which will be triggered when accesing a link. The implementation itself is done in views.

The model component is the “reflection” in code of the corresponding database table (because Django uses Active-Record Pattern).

# Data Model

# 

# 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

Sistemul va avea teste pentru fiecare actiune pe care o poate face un utilizator sau un angajat.

# Future improvements

*[Present future improvements for the system]*

Imbunatatiri ulterioare pot fi:

Crearea unui chat pentru dialogul cu un angajat in mod direct, avand masinile prezentate in fata.

Afisarea unei harti de unde poate ridica masina.

# Bibliography

<https://docs.djangoproject.com/en/2.2/>

<https://docs.python.org/2.7/>