<Hardware/Software Manager>

Use-Case Model

Version <1.0>

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| <23/03/2017> | <1.0> | <details> | <Oana Brudan> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

[Use-Case Model 4](#_Toc479804843)

[1.1 Use-Cases Identification 4](#_Toc479804844)

[1.2 UML Use-Case Diagrams 4](#_Toc479804845)

[Domain Model 6](#_Toc479804846)

[Arhitectural Design 7](#_Toc479804847)

[Component Diagram 8](#_Toc479804848)

# 

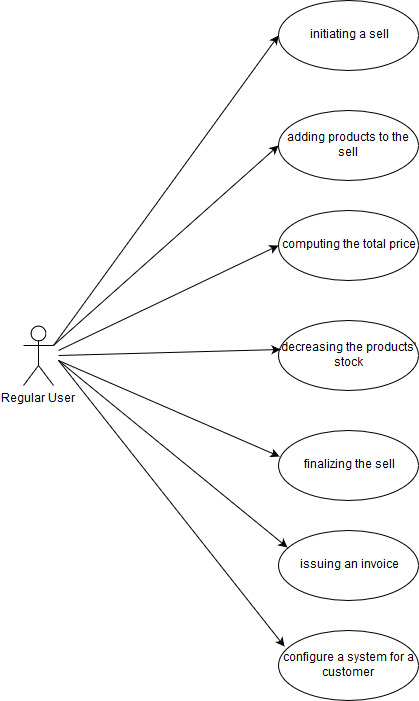
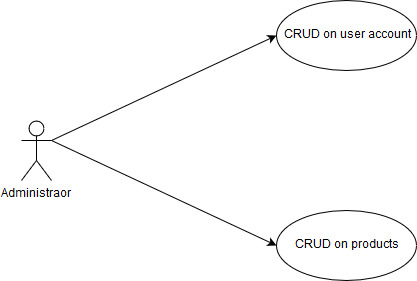
## Use-Cases Identification

Use case: defineste interactiunile dintre utilizator si sistem

Actor principal: Administrator

Scenariu principal: Administratorul creeaza un cont de utilizator. Utilizatorul initiaza si se ocupa de o vanzare pana la finalizarea acesteia.

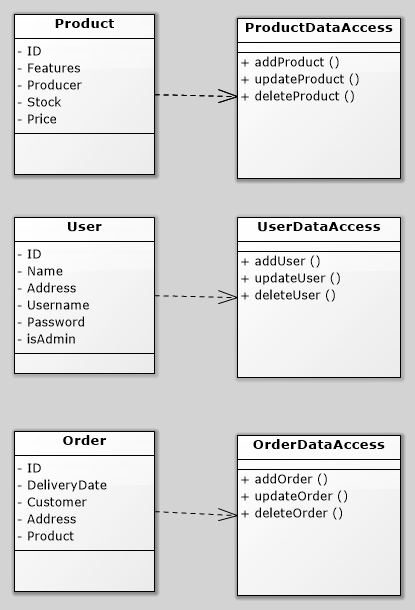
## UML Use-Case Diagrams



# Domain Model

Domain model defineste patternul folosit in proiect.

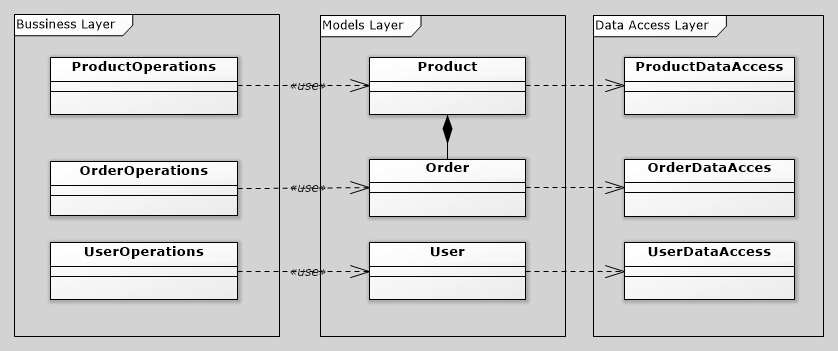
Pentru proiectul voi folosi DAO (Data Access Object) care este un caz particular al patternului Table Data Gateway. Acesta presupune ca fiecare table din baza de date are o clasa corespunzatoare in proiect prin intermediul careia se realizeaza comunicarea cu baza de date. Fiecarui obiect ii corespunde un Gateway, adica o clasa pentru accesarea datelor din obiectul respectiv.



# Arhitectural Design

Arhitectural Design ne arata modul in care interactioneaza clasele si obiectele intre ele.

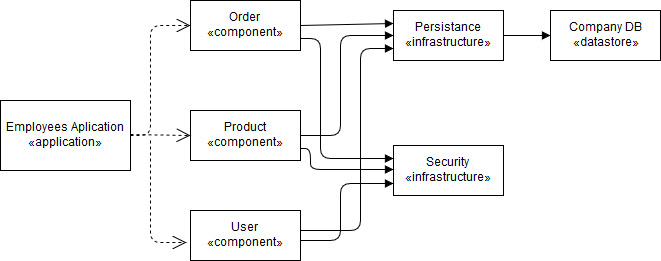
Avem 3 layere, BL (Bussiness layer) care se ocupa cu operatiile realizate asupra obiectelor, Models Layer care contine obiectele si DAL (Data Access Layer) care asigura accesul la catre date.



# Component Diagram

Diagrama de componente descrie componentele folosite pentru a face posibila functionarea proiectului.

Avem asadar aplicatia in sine, componentele pe baza carora se desfasoara activitatea aplicatiei, Order, Product si User, infrastructura care asigura siguranta si persistenta si baza de date unde se pastreaza datele.



**Deployment Diagram**

Deployment diagram arata o distributie a structurii proiectului.

Pe de o parte avem procesorul folosit de Server. Acesta se ocupa cu baza de date. Pe cealalta parte avem dispozitivul folosit de client. Aici avem aplicatia in sine.

