Assignment 1

Travel agency

Student(s):

* S1
* S2

**Group:**

**Contents**

[I Project specification](#_heading=h.gjdgxs) **3**

[1.1 Domain Model Diagram](#_heading=h.30j0zll) 3

[II Use-Case model](#_heading=h.1fob9te) **3**

[2.1 Users and stakeholders](#_heading=h.f2irg1azp7rm) 4

[2.2 Use-Case identification](#_heading=h.3znysh7) 4

[2.3 UML Use-Case diagram](#_heading=h.2et92p0) 4

[III Architectural design](#_heading=h.tyjcwt) **5**

[3.1 Conceptual architecture](#_heading=h.3dy6vkm) 5

[3.2 Package diagram](#_heading=h.1t3h5sf) 5

[3.3 Class diagram](#_heading=h.4d34og8) 5

[3.4 Database (E-R/Data model) diagram](#_heading=h.2s8eyo1) 6

[3.5 Sequence diagram](#_heading=h.17dp8vu) 6

[3.6 Activity diagram](#_heading=h.3rdcrjn) 6

[IV Supplementary specifications](#_heading=h.26in1rg) **6**

[4.1 Non-functional requirements](#_heading=h.lnxbz9) 6

[4.2 Design constraints](#_heading=h.35nkun2) 6

[V Testing](#_heading=h.1ksv4uv) **7**

[5.1 Testing methods/frameworks](#_heading=h.44sinio) 7

[5.2 Future improvements](#_heading=h.2jxsxqh) 7

[VI Bibliography](#_heading=h.z337ya) **7**

# I Project specification

TheTravelling Agencyshould be able to:

1.add vacation destination

2.add vacation packages for aspecific destinationa.shouldcontaininformation:name,price,period,extradetails,numberof people that can book the vacation

3.edit an existing vacation package

4.delete an existing vacation package

5.viewallitslistedvacationpackages(withstatus:BOOKED,NOT\_BOOKED,IN\_PROGRESS)

6.delete vacation destination

TheRegular Usershould be able to:

1.registerontheplatformusingsomecredentials(username/email-unique&password)

2.login on the platform

3.view allavailablevacation packages

4.filter vacation packages by destination/price/period

5.book a vacation

6.view all its booked vacations

## 1.1 Domain Model Diagram

*< Schema Domain Model a aplicatiei. Se va discuta la laborator. />*

# II Use-Case model

*< Se va scrie o mica introducere./>*

## 2.1 Users and stakeholders

Users

Travel Agency

## 2.2 Use-Case identification

*< Aici se vor prezenta 3-4 use-case-uri mai importante din applicatie dupa urmatorul model*

***Nume Use case/Use case name: < Nume use-case>.***

***Nivel/Level: < User-Goal, Subfunction, Summary > .***

***Actor principal/Main actor: < Actorul scenariului> .***

***Scenariul principal de success/Main success scenario: <Descriere detaliata a scenariului>.***

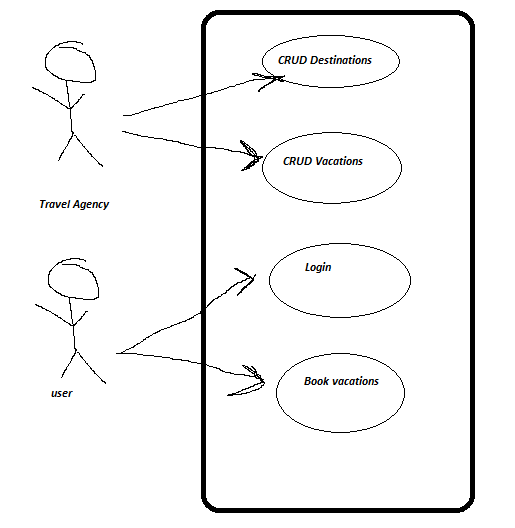
***Extensie/Extension: <Un caz particular al scenariului, fie pozitiv sau negativ>***

*Se va discuta la laborator mai detaliat.*

*/>*

Travel Agency CRUD

## 2.3 UML Use-Case diagram



# III Architectural design

Layered architecture

## 3.1 Conceptual architecture

* **The presentation layer** contains all of the classes responsible for presenting the UI to the end-user or sending the response back to the client (in case we’re operating deep in the back-end).
* **The application layer** contains all the logic that is required by the application to meet its functional requirements and, at the same time, is not a part of the domain rules. In most systems that I've worked with, the application layer consisted of services orchestrating the domain objects to fulfill a use case scenario.
* **The domain layer**represents the underlying domain, mostly consisting of domain entities and, in some cases, services. Business rules, like invariants and algorithms, should all stay in this layer.
* **The infrastructure layer (also known as the persistence layer)**contains all the classes responsible for doing the technical stuff, like persisting the data in the database, like DAOs, repositories, or whatever else you’re using.

## 3.3 Class diagram

*< (Class Diagram)/>*

## 3.4 Database (E-R/Data model) diagram

*< (Data Model)/>*

## 3.5 Sequence diagram

*< (Sequence Diagram)/>*

## 3.6 Activity diagram

*< (Activity Diagram)/>*

# IV Supplementary specifications

*< Se va scrie o mica introducere./>*

## 4.1 Non-functional requirements

*< Specificatiile non-functionale ale aplicatiei. Se va discuta la laborator./>*

## 4.2 Design constraints

Technical constraints:●Desktop application, written inJava+Java Swing/JavaFX●Connectivity torelational database+ storage of data○each table in the database should contain at least 5 entries●Implement Data Access Layer usingORM(eg. Hibernate)●Implement the app usingLayered Architecture●Documentation●Validationoftheinputsonspecificflows(validdates,non-negativevalues, unique username etc.)