Booking System

~ Website pentru Rezervare de

Unitati de Cazare ~

Name: Flinta Claudiu

Group: 30236

Table of Contents

[Deliverable 1 3](#_Toc64843130)

[Project Specification 3](#_Toc64843131)

[Functional Requirements 3](#_Toc64843132)

[Use Case Model 3](#_Toc64843133)

[Use Cases Identification 3](#_Toc64843134)

[UML Use Case Diagrams 3](#_Toc64843135)

[Supplementary Specification 3](#_Toc64843136)

[Non-functional Requirements 3](#_Toc64843137)

[Design Constraints 3](#_Toc64843138)

[Glossary 3](#_Toc64843139)

[Deliverable 2 3](#_Toc64843140)

[Domain Model 3](#_Toc64843141)

[Architectural Design 4](#_Toc64843142)

[Conceptual Architecture 4](#_Toc64843143)

[Package Design 4](#_Toc64843144)

[Component and Deployment Diagram 4](#_Toc64843145)

[Deliverable 3 4](#_Toc64843146)

[Design Model 4](#_Toc64843147)

[Dynamic Behavior 4](#_Toc64843148)

[Class Diagram 4](#_Toc64843149)

[Data Model 4](#_Toc64843150)

[System Testing 4](#_Toc64843151)

[Future Improvements 4](#_Toc64843152)

[Conclusion 4](#_Toc64843153)

[Bibliography 4](#_Toc64843154)

# Deliverable 1

## Project Specification

Proiectul Booking System reprezinta un website destinat zonei de turism, mai exact promoveaza diferite unitati de cazare, iar prin intermediul site-ului se pot realiza rezervari pentru unitatiile de cazare disponibile. Dat fiind faptul ca industria de turism incepe sa se dezvolte din ce in ce mai tare in ultima perioada, apare o cerere in directia digitalizarii acestei zone.

Principalul obiectiv este acela de a dezvolta o aplicatie “user oriented”, in care experienta de utilizare sa fie cat mai intuitiva, simpla si rapida. Un alt obiectiv ar fi acela de a intercoecta si de a reusi sa aducem intru-un singur loc cat mai multe unitati de cazare, atat in regim hotelier, cat si in regim propriu, si de a le expune unei mese extinse de indivizi.

Privind procesul de dezvoltare, aplicatie este create cu ajutorul urmatoarelor tehnologii de dezvoltare software: Java&Spring, React, MySQL.

## Functional Requirements

* Registration and Log-In: Utilizatorii au posibilitatea de a se înregistra și de a crea un cont pe platformă prin furnizarea anumitor informațiilor personale.
* Funcționalitatea de căutare și rezervare: Aplicația permite utilizatorilor să navigheze și să caute proprietățile disponibile în funcție de preferințele lor, cum ar fi locația, gama de prețuri, tipul de proprietate etc. Utilizatorii ar trebui, de asemenea, să poată rezerva proprietăți direct prin intermediul aplicației.
* Gestionarea unitatiilor de cazare: Administratorul poate lista cu ușurință cazarile, inclusiv imagini, detalii despre proprietate și prețuri.
* Comunicare: Aplicația va avea un sistem de comunicare care să permita interactiunea intre utilizatori. Va fi un fel de forum, in care utilizatorii vor putea imparatasi experiente, vor putea pune intrebari si vor primi raspunsuri.
* User review and rating system: Aplicația permite utilizatorilor să evalueze și să ofere o nota unitatii de cazare, ajutând alți utilizatori să ia decizii informate în timp ce aleg o unitate de cazare.

## Use Case Model 1

### Use Cases Identification

**Use-Case**: Registration

**Level**: primary level

**Primary Actor**: Administrator/Normal User

**Main success scenario**: User-ul introduce datele in fieldurile associate, dupa care contul de salveaza in tabela corespunzatoar utilizatorilor platformei

**Extensions**: -

**Use-Case**: Log-in

**Level**: primary level

**Primary Actor**: Administrator/Normal User

**Main success scenario**: User-ul introduce datele in fieldurile associate, dupa care se valideaza si in caz de success, se deschide pagina corespunzatoare tipului de user.

**Extensions**: schimbarea parolei in cazul in care a fost uitata

**Use-Case**: Add/Delete/Update Hotels

**Level**: primary level

**Primary Actor**: Administrator

**Main success scenario**: Administratorul poate realiza diferite operatiuni asupra unitatiilor de cazare disponibile pe site. Poate adauga unitati noi, modifica datele celor existente sau le poate sterge definitive.

**Extensions**: -

**Use-Case**: View Reservations

**Level**: primary level

**Primary Actor**: Normal User

**Main success scenario**: User-ul poate sa vizualizeze rezervarile pe care le are active, dar si pe cele precedente.

**Extensions**: -

**Use-Case**: Search for hotels by certain criteria

**Level**: primary level

**Primary Actor**: Normal User

**Main success scenario**: User-ul poate filtra unitatiile de cazare in functie de anumite criterii, cum ar fi: data, regiunea, tip-ul camerei etc.

**Extensions**: -

**Use-Case**: Make a reservation

**Level**: primary level

**Primary Actor**: Normal User

**Main success scenario**: User-ul poate crea o rezervare a unei unitati de cazare pentru o anumita perioada de timp.

**Extensions**: -

### UML Use Case Diagrams

Diagram

Description automatically generated

## Supplementary Specification

### Non-functional Requirements

* Securitate: Aplicația trebuie să ofere un mediu sigur pentru toti utilizatorii, inclusiv pe partea de criptare a datelor (unde este cazul) si pe partea de autentificarea utilizatorilor.
* Fiabilitate: Aplicația trebuie să fie extrem de fiabilă, cu timpi minimi de nefuncționare, sa gestioneze eventualele erori și sa fie remediate in timp minimal.
* Utilizare: Aplicația trebuie să fie user oriented, simpla, cu instrucțiuni clare și concise pentru oaspeți și gazde.
* Performanță: Aplicația trebuie să aibă timpi de răspuns rapizi, timpi rapizi de încărcare a paginii și latență minimă pentru a asigura o experiență buna a utilizatorului.

### Design Constraints

* Limbaj de programare : Java&Spring (backend), React (frontend) si MySQL (database).
* Unit Testing : JUnit
* Integration Testing
* IDE : Intellij

# Deliverable 2

## Domain Model

* [Define the domain model and create the conceptual class diagrams]

## Architectural Design

### Conceptual Architecture

* [Define the system’s conceptual architecture; use an architectural style and pattern - highlight its use and motivate your choice.]

### Package Design

* [Create a package diagram]

### Component and Deployment Diagram

* [Create the component and deployment diagrams.]

# Deliverable 3

## Design Model

### Dynamic Behavior

* [Create the interaction diagrams (1 sequence, 1 communication diagrams) for 2 relevant scenarios]

### Class Diagram

* [Create the UML class diagram; apply GoF patterns and motivate your choice]

## Data Model

* [Create the data model for the system.]

# System Testing

* [Describe the testing methides and some test cases.]

# Future Improvements

* [Present some features that apply to the application scope.]

# Conclusion

# Bibliography