UNIVERSITATEA ALEXANDRU IOAN CUZA IAŞI

### FACULTATEA DE INFORMATICĂ



LUCRARE DE LICENŢĂ

Hotel Manager

propusă de

***Adrian Ștefan***

Sesiunea: *iulie, 2019*

Coordonator ştiinţific

##### Colaborator Florin Olariu

UNIVERSITATEA ALEXANDRU IOAN CUZA IAŞI FACULTATEA DE INFORMATICĂ

Hotel Manager

***Adrian Ștefan***

Sesiunea: *iulie, 2019*

Coordonator ştiinţific

###### **Colaborator Florin Olariu**

DECLARAŢIE PRIVIND ORIGINALITATE ŞI RESPECTAREA DREPTURILOR DE AUTOR

Prin prezenta declar că Lucrarea de licenţă cu titlul “*Hotel Manager*” este scrisă de mine şi nu a mai fost prezentată niciodată la o altă facultate sau instituţie de învăţământ superior din ţară sau străinătate. De asemenea, declar că toate sursele utilizate, inclusiv cele preluate de pe Internet, sunt indicate în lucrare, cu respectarea regulilor de evitare a plagiatului:

* toate fragmentele de text reproduse exact, chiar şi în traducere proprie din altă limbă, sunt scrise între ghilimele şi deţin referinţa precisă a sursei;
* reformularea în cuvinte proprii a textelor scrise de către alţi autori deţine referinţa precisă;
* codul sursă, imagini etc. preluate din proiecte *open*-*source* sau alte surse sunt utilizate cu respectarea drepturilor de autor şi deţin referinţe precise;
* rezumarea ideilor altor autori precizează referinţa precisă la textul original.

Iaşi, *data*

Absolvent *Adrian Ștefan*

DECLARAŢIE DE CONSIMŢĂMÂNT

Prin prezenta declar că sunt de acord ca Lucrarea de licenţă cu titlul “*Hotel Manager*”, codul sursă al programelor şi celelalte conţinuturi (grafice, multimedia, date de test etc.) care însoţesc această lucrare să fie utilizate în cadrul Facultăţii de Informatică.

De asemenea, sunt de acord ca Facultatea de Informatică de la Universitatea Alexandru Ioan Cuza Iaşi să utilizeze, modifice, reproducă şi să distribuie în scopuri necomerciale programele-calculator, format executabil şi sursă, realizate de mine în cadrul prezentei lucrări de licenţă.

Iaşi, *data*

Absolvent *Adrian Ștefan*

## Acord privind proprietatea dreptului de autor

Facultatea de Informatică este de acord ca drepturile de autor asupra programele-calculator, format executabil şi sursă, să aparţină autorului prezentei lucrări, *Prenume Nume.*

Încheierea acestui acord este necesară din următoarele motive:

*[Se explică de ce este necesar un acord, se descriu originile resurselor utilizate în realizarea produsului-program (personal, tehnologii, fonduri) şi aportul adus de fiecare resursă.]*

Iaşi, *data*

Decan *Adrian Iftene* Absolvent *Adrian Ştefan*

**Cuprins**

Introducere6

Capitolul 1: Tehnologii folosite4

Descriere generala5

Type chapter title (level 3)6

Capitolul 2: Tehnologii folosite4

Descriere generala5

Type chapter title (level 3)6

**Introducere**

Tema licentei este construirea unei aplicații web care are ca și scop managemntul unui hotel. Toți utilizatori aplicației vor avea posibilitatea să vadă informații generale despre hotel și serviciile oferite de acesta. Principalele utilități ale aplicației pentru client sunt cea de rezervare a camerelor și de plasare a comenzilor de tip room-service. Accesul spre serviciile de rezervare de camere si comenzi room-service se face prin autentificarea cu un cont personal.

Aplicația oferă posibilitatea înregistrări cu trei tipuri de conturi, admin, angajat și utilizator. Ca și admin poți crea diferite categorii de camere. Odată creată o categorie poti adăuga una sau mai multe camere cu categoria respectivă acestea devenind disponibile spre rezervare. De asemenea, admini au și posibilitatea de a adauga diverse tipuri de produse pentru a putea fi comandate prin serviciul de room-service. Rezervarea camerelor poate fi facută de toți utilizatorii logați și plata se face prin PayPal pentru finalizarea rezervării. Comenzile room-service pot fi făcute de utilizatorii care sunt logați și au o rezervare de camera în momentul efectuării comenzii. Conturile de tip angajat pot vedea toate rezervările și comenzile de room-service. In plus, aceștia pot face rezervări și comenzi de room-service pentru clienți care nu au cont personal, plata pentru aceste rezervări poate fi de orice tip, nu doar PayPal.

O altă utilitate importantă aplicației este cea de generare a unui cod de bare la momentul înregistrării. Cu acest cod de bare clienții vor putea intra în camerele unde au rezervări pe parcursul rezervării. Fiecare camera a hotelului va avea un dipozitiv capabil sa scaneze codul de bare, iar în cazul în care este scanat codul de bare corect va descuia camera. Codul de bare necesar descuierii camerei este actualizat în funcție de rezervarea activă pentru acea camera.

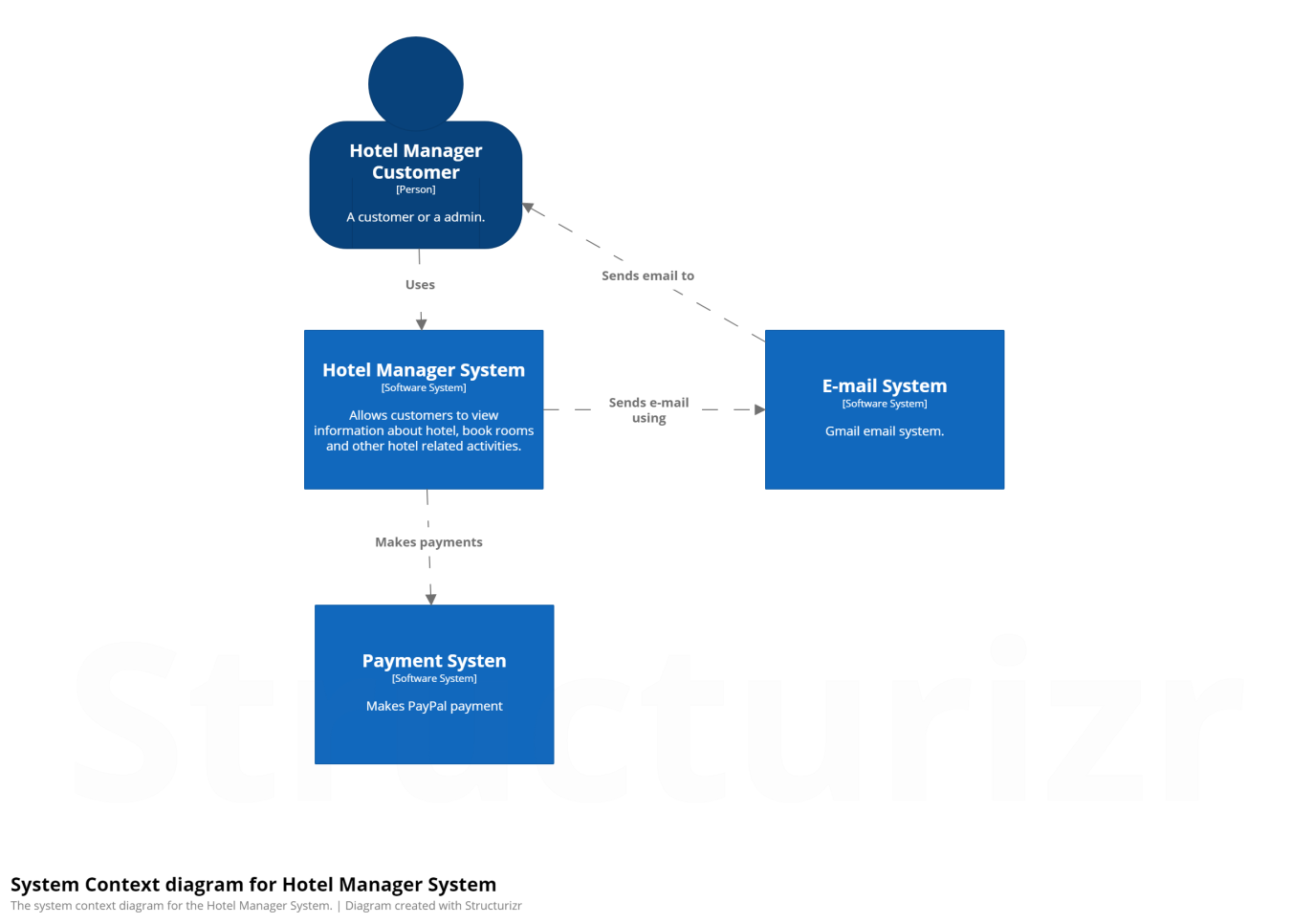
Motivul pentru care am ales această temă de licență este că, deși există multe hoteluri cu aplicații care să ofere posibilitatea rezervării unei camere, nu am găsit nici o aplicație care să faciliteze toată experiența ta la acel hotel prin intermediul aplicației. La aceste hoteluri chiar dacă îți faci rezervare la cameră online odată ajuns la hotel tot trebuie să mergi la recepție pentru a primi cheia camerei ceea ce în anumite cazuri poate duce la așteptarea la timp pierdut prin așteptarea la cozi lungi la recepție.

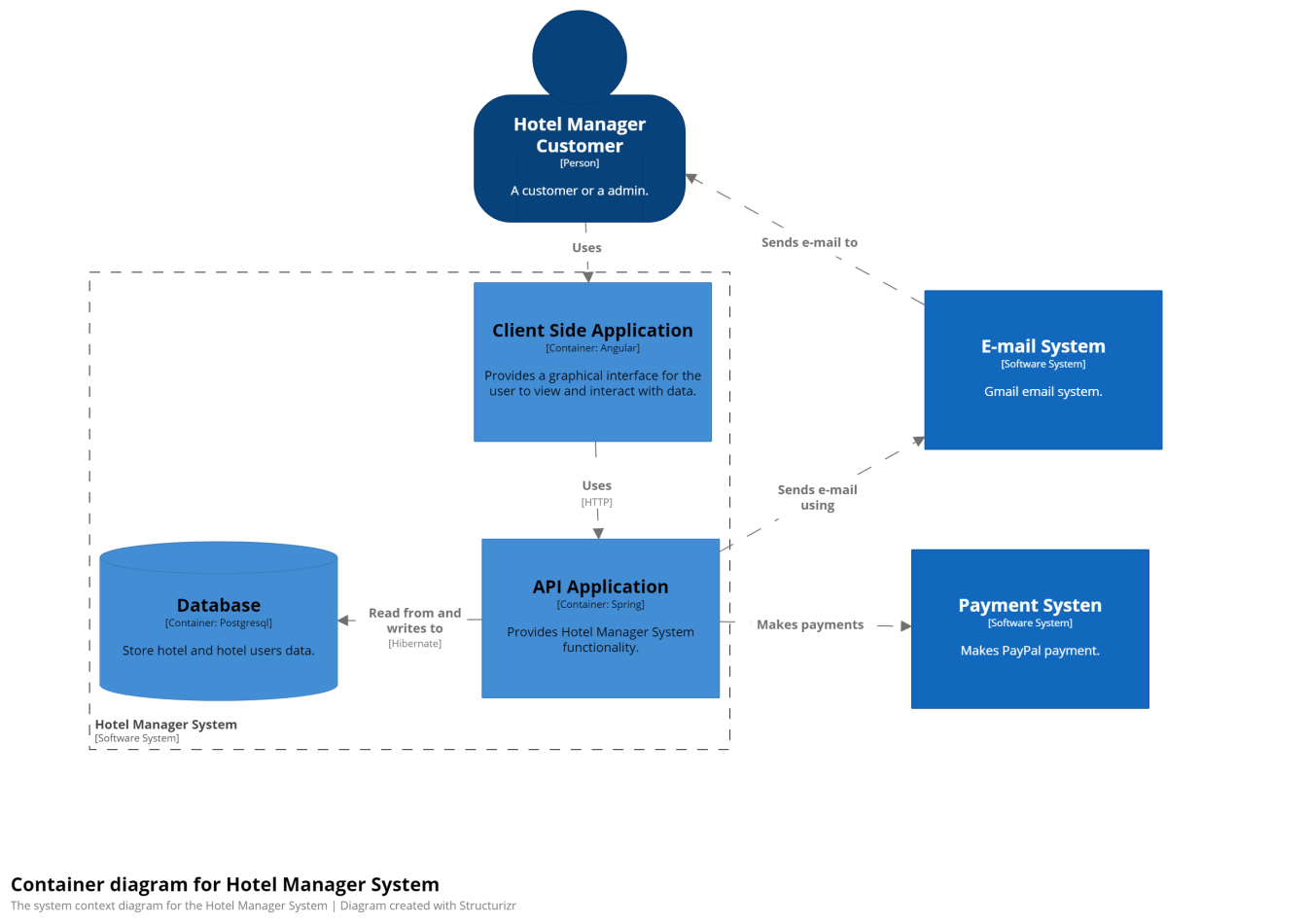
Alt element care nu l-am văzut la nici o altă aplicatie este posibilitatea plasării unei comenzi de room-service, și consider ca este mai mult pe placul lumii să plaseze o comandă prin internet și să poată vedea statusul comenzii decât prin intermediul telefonului sau în alte moduri.

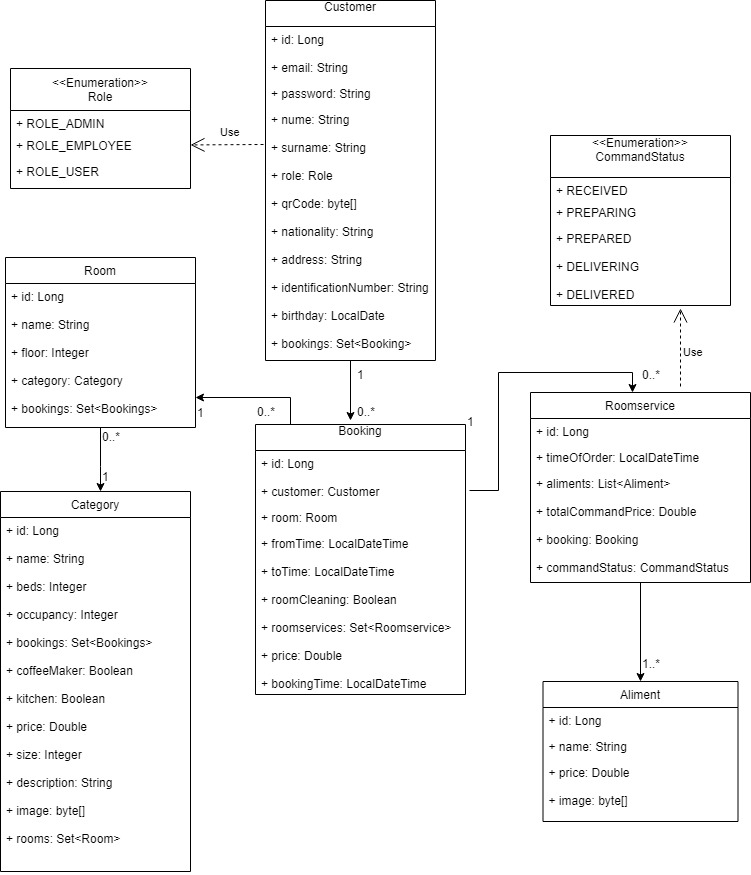
De asemenea, aplicația foloseste și algoritmi pentru a face prețul rezervării să varieze în funcție de diverse criterii, printer care și oferirea de reduceri în funcție de rezervările anterioare.

Așadar, consider că cel mai mare plus adus de aplicația mea este oferirea posibilității cazării și cererii serviciilor la un hotel doar prin intermediul aplicației, asta făcând atât cazarea cât și perioada de cazare la hotel mult mai simplă și mai plăcută pentru client.

**Diagrame arhitecturale**

****

****

****

**Tehnologii folosite**

Proiectul este împărțit în două aplicații, o aplicație este un API care se ocupă de prelucrarea informațiilor, cât și stocarea si preluarea informațiilor in baza de date, această parte este numită partea de back-end. A doua aplicație a proiectului este interfața grafică a proiectului, aceasta făcând apeluri spre partea de back-end și afisându-le în interfața grafică, această parte este numită front-end.

Tehnologiile folosite pentru back-end sunt limbajul de programare Java 8, frameworkul Hibernate pentru lucrul cu baza de date și frameworkul Spring. Baza de date folosită este PostgreSQL. Pentru aducerea de dependințe externe în proiect este folosit sistemul Maven. Spre exemplu, pentru folosirea frameworkurilor este folosit Maven. Pentru a folosi Maven se creează în proiect un fișier numit “pom.xml”. În acest fișier se pot face mai multe configurări, dar stric pentru aducerea de dependințe externe este folosit tagul <dependencies> </dependencies>, în interiorul căruia se adaugă diverse dependințe preluate de pe internet.

Hibernate este un framework care se ocupă cu maparea claselor folosite în cod la tabele în baza de date, astfel lucrând cu obiecte și clase direct pentru a prelua și adauga informații în baza de date. Pentru a folosi acest framework este folosit Java Persistence API, numit JPA, aceasta fiind specificția unei interfețe de programare a aplicațiilor care descrie managementul datelor relaționale în aplicații foloșind Hibernate pentru lucrul cu baza de date. JPA foloseste diverse adnotări și interfețe pentru lucrul cu baza de date. Unele din cele mai importante adnotări sunt @Entity folosită asupra unei clase pentru a mapa acea clasă la o tabelă în baza de date, @OneToOne, @OneToMany, @ManyToOne, @ManyToMany pentru a face legături între diferite tabele și adnotarea @Transactional pentru a specifica ca într-o anumită metodă sau clasă să se execute codul sub formă de tranzacție.

Spring este un framework cu multe utilități, cele mai importante utilități folosite în proiectul meu sunt Spring Data, Spring Boot, mecanism de inversiune de control, mecanism pentru cereri web de tipul verbelor HTTP cunoscute și Spring Security.

Spring Data oferă mai multe funcționalități peste JPA, adică multe metode pentru preluarea și stocarea infromațiilor în baza de date. În proiectul meu am folosit JpaRepository, aceasta este o interfață cu diferite implementări standard pentru lucrul cu baza de date, spre exemplu findById(id) pentru preluarea obiectului cu id-ul cerut din baza de date, findAll() pentru preluarea tuturor obiectelor din tabela mapată la clasa cerută, save(obiect) pentru adăugarea unui obiect în baza de date. De asemenea, JpaRepository oferă posibilitatea de a crea întrebări specifice la baza de date prin crearea unei metode în JpaRepository deasupra la care este pusă adnotarea @Query(“”) în interiorul căreia este scrisă întrebarea specifică dorită spre baza de date.

Spring Boot este un mecanism folosit pentru crearea configurațiilor aplicației și pentru rularea aplicației.

Mecansimul de inversiune de control este realizat prin injectare de dependințe. Pentru a putea injecta un serviciu trebuie declarată o interfață și pusă o adnotare de tip @Service peste o clasă care implementează acea interfață dupa care această clasă trebuie declarată în constructorul clasei și deasupra constructorului trebuie pusă adnotarea @Autowired.

Mecanismul pentru cereri web este realizat prin diverse adnotări pentru a zice ca o metoda este o reprezentare pentru un verb HTTP. Pentru a defini o clasă ca o clasă cu verbe HTTP este pusă adnotarea @RestController deasupra ei și adnotarea @RequestMapping(“”) pentru a zice calea la care vom găsi verbele HTTP definite în clasă. Adnotările @PostMapping, @GetMapping, @PutMapping, @PatchMapping sunt folosite deasupra unei metode din clasă pentru a marca acea metodă ca unul din verbele HTTP. Pe lânga aceste adnotări mai sunt folosite și adnotările @PathVariable, @RequestBody, @ReqeustParam pentru folosirea de varabile în URL, informații transime in corupl cererilor web sau ca parametri.

Spring Security este folosit pentru autentificarea și securizarea aplicației de back-end prin generarea de tokeni la autentificare și folosirea acestor tokeni pentru a accesa diferite cereri web în funcție de gradul lor de securitate.

Tehnologiile folosite pentru front-end sunt frameworkul Angular 7, limbajul de programare TypeScript, limbajul de scripting HTML5 si limbajul de modelare CSS3. De asemenea, sunt folosite și serviciile Bootstrap, Angular Material și PrimeFaces pentru folosirea de elemente de interfață grafică deja definite.

Angular este un framework care folosește ca bază module, componente și servicii. Modulele sunt folosite pentru organizarea codului, acestea fiind compuse din componente, servicii și/sau pipeuri. Componentele sunt create dintr-un fișier HTML, unul CSS si unul TypeScript din care se formează o pagină sau o parte a unei pagini web. Fisierele HTML și CSS oferă interfața grafică și fisierul TypeScript face apleuri folosind servicii la aplicația de back-end pentru preluare de date din baza de date și apoi prelucrarea acestora pentru a fi afișate în interfața grafică.

**Arhitectura proiect**

Proiectul are o arhitectura pe trei nivele, prezentare, aplicație și date.

Nivelul prezentare este interfața grafică a proiectului, deci acesta este reprezentat de aplicația de front-end. Acesta face cereri la aplicația de back-end și trimite informații browserelor în forma de HTML și CSS folosind frameworkul Angular.

Nivelul aplicație este aplicația de back-end. Aceasta folosește controllere pentru a prelua cererile web de la aplicația de front-end, apoi folosește servicii penru a prelucra informațiile și a adauga sau pentru a prelua anumite informații din baza de date.

Nivelul de date este reprezentat de baza de date, care se ocupa de păstrarea și recuperarea informatiilor de către utilizator. Baza de date folosită in proiect este PostgreSQL.

Aplicația de front-end este împărțită în trei module, core, modules și shared. În modulul core se găsesc modelele pentru lucrul cu date primite de la back-end, serviciile care fac cererile web și gărzile necesare pentru securitate în funcție de gradul de acces al utilizatorului. Modulul modules conține la rândul lui diferite module organizate care conțin componente necesare pentru a crea o pagina web. În final, ultimul modul este shared care conține informații comune pentru celelalte module cum ar fi componentele cu header și footer, pipeuri serviciu pentru validări și aducere de diverse module externe.

Aplicația de back-end este împarțită la rândul ei pe trei nivele, nivelul de preluare a cererilor web de la aplicația de front-end folosind controllere și apoi folosind servicii pentru prelucararea datelor, serviciile reprezentând al doilea nivel. Ultimul nivel constă în lucrul cu bazele de informații, adică preluarea informațiilor și adăugarea acestora în baza de date, prin intermediul interfeței JpaRepository. Nivelul de lucrul cu bazele de date este folosit de nivelul de servicii. Atât folosirea serviciilor cât și a interfețelor se face prin injectare de dependințe.

Pentru lucrul cu baza de date se folosesc modele mapate la tabele din baza de date (la figura x se poate vedea diagrama de claselor de modele). Dar pentru preluarea de informații din requesturile web și pentru returnare de informații sunt folosite alte clase asemănătoare cu modelel din baza de date, numite DTO. Motivul folosirii acestor clase este dat de faptul ca modelele trebuie să aibă aceeași formă ca și tabela din baza de date iar uneori putem să avem nevoie să primim sau sa oferim informații diferite fața de cele din model. Pentru a face conversia dintre aceste clase, adică de la model la DTO și invers, este folosit câte un serviciu numit mapper care conține metodele necesare pentru a mapa la tipul care avem nevoie. Procesul de mapare este făcut prin crearea unui obiect de noul tip și instanțierea acestuia cu valorile din obiectul dinspre care este mapat.

Aplicația folosește excepții proprii care extind clasa RuntimeException și le este dat un cod de status HTTP. Spre exemplu, dacă se caută o camera după un id în baza de date și nu este găsită acea cameră este aruncată excepția RoomNotFoundException cu codul de status 404.

De asemnea, sunt folosite și două sisteme externe, unul pentru plata prin PayPal și unul pentru trimiterea de emailuri prin Gmail.

Pentru partea de securitate și autorizare este folosit Spring Security pentru a genera un token la autentificare în funcție de rolul utilizatorului (admin pentru utilizatori de tip admini, employee pentru utilizatori care sunt angajati sau user pentru utilizatori normali care folosesc aplicația pentru a face o rezervare de cameră sau pentru a comanda room-service). Apoi sunt configurate cererile web pentru a oferi securitatea necesară în funcție de caz, securitatea acestora fiind verificată dupa informațiile din token.

Aplicația respectă Regulamentul General Privind Protecția Datelor prin codificarea datelor introduse în baza de date și decodificare acestora la preluarea datelor din baza de date. Pentru o securitate și mai buna pentru parolă este folosită o funcție hash înainte de a fi introdusă în baza de date, iar apoi pentru verificarea parolei la autentificare se vor compara hashul din baza de date cu cel al parolei introduse pentru autentificare.

**Rezervare camera**

**Comanda room-service**

**Algoritm preț variabil**

**Funcţii admini si angajați**

**Trimitere emailuri**

**Plata PayPal**

**Securitate**

**Testare unitara**

**Bibliografie**

<https://angular.io/tutorial>

<https://coursetro.com/posts/code/171/Angular-7-Tutorial---Learn-Angular-7-by-Example>

<https://material.angular.io/>

<https://www.jetbrains.com/help/idea/running-and-debugging-typescript.html>

<https://getbootstrap.com/>

<https://www.w3schools.com/bootstrap4/default.asp>

<https://www.primefaces.org/primeng/#/>

<https://angular.io/guide/styleguide>

<https://itnext.io/choosing-a-highly-scalable-folder-structure-in-angular-d987de65ec7>

<https://github.com/mathisGarberg/angular-folder-structure>

<https://malcoded.com/posts/angular-fundamentals-modules/>

<https://www.technouz.com/4772/angular-6-app-structure-with-multiple-modules/>

<https://code-maze.com/angular-best-practices/>

<https://www.freecodecamp.org/news/how-to-center-things-with-style-in-css-dc87b7542689/>

<https://css-tricks.com/snippets/css/a-guide-to-flexbox/>

<https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Flexible_Box_Layout/Ordering_Flex_Items>

<https://flexboxfroggy.com/>

<https://www.baeldung.com/security-spring>

<https://www.baeldung.com/securing-a-restful-web-service-with-spring-security>

<https://www.baeldung.com/spring-security-oauth-jwt>

<https://chariotsolutions.com/blog/post/angular-2-spring-boot-jwt-cors_part2/>

<https://mherman.org/blog/token-based-authentication-with-angular/#localstorage>

<https://codecraft.tv/courses/angular/dependency-injection-and-providers/tokens/>

<https://medium.com/engineerbabu/angular-authentication-using-jwt-d846c5ce0ac6>

<https://medium.com/@ryanchenkie_40935/angular-authentication-using-route-guards-bf7a4ca13ae3>

<https://nirajsonawane.github.io/2018/10/27/Angular-Material-Tabs-with-Router/>

<https://medium.com/@nikhildevre/creating-tabs-using-angular-material-2-and-angular-4-routing-3634c3d0f7cc>

<https://blog.angularindepth.com/angular-router-series-secondary-outlets-primer-139206595e2>

<https://www.w3schools.com/howto/howto_js_tabs.asp>

<https://css-tricks.com/left-and-right/>

<https://css-tricks.com/couple-takes-sticky-footer/>

<https://github.com/paypal/PayPal-Java-SDK>

<https://developer.paypal.com/docs/api/quickstart/payments/#additional-information>

<https://developer.paypal.com/developer/accountStatus/>

<https://c4model.com/>

<https://structurizr.com/express?src=/static/express/bigbankplc/system-context.json>

<https://www.draw.io/>

<https://hatchful.shopify.com/>

<https://www.pexels.com/search/hotel/>

<https://pixabay.com/images/search/hotel/>

<https://unsplash.com/search/photos/room-service>