



UNIVERSITATEA DIN  
BUCUREŞTI



FACULTATEA DE  
MATEMATICĂ ȘI  
INFORMATICA

SPECIALIZAREA INFORMATICĂ

Lucrare de licență

**APLICAȚIE WEB PENTRU  
GESTIONAREA SPAȚIILOR  
COMUNE ALE UNEI FIRME  
(OFOPS)**

Absolvent

Iftimi Ruxandra

Coordonator științific

Conf. univ. dr. Boriga Radu Eugen

București, iunie - iulie 2024

## **Rezumat**

Aplicația WEB prezentată în această lucrare de licență se numește OfOps, aceasta având ca scop gestionarea spațiilor comune ale unei firme astfel încât desfășurarea activităților de birou să fie eficiente.

Spațiile uneori insuficiente pentru toți angajații firmei, necesitatea de a ajunge mai devreme pentru a găsi liber biroul preferat sau chiar un loc de parcare sunt experiențe neplăcute, trăite, care au reprezentat baza dezvoltării acestei aplicații. Dorința de a cunoaște disponibilitățile birourilor, sălilor de ședință și al locurilor de parcare libere mi s-a părut un punct de plecare esențial pentru ca aplicația OfOps să ne scape de unele griji inutile.

Aplicația dispune de un calendar și un ceas pentru a selecta intervalul de timp în care se dorește o vizualizare a disponibilităților spațiilor la un anumit moment, dar și de hărți interactive. Acestea au rolul să faciliteze experiența navigării și să evidențieze spațiile colorate diferit în funcție de statusul rezervării lor. Paleta de culori folosită pentru a marca locurile este: verde – loc liber –, roșu – ocupat – și galben – rezervarea a fost făcută de utilizatorul autentificat în aplicație –.

Rezervarea se poate efectua printr-un simplu click pe locul dorit. Însă, în situația în care mai există deja rezervări pentru acel loc, utilizatorul va fi anunțat printr-o notificare în cazul în care în intervalul dorit este programată altă rezervare. Pentru sălile de ședință, user-ul va primi o recomandare de sală disponibilă pentru intervalul solicitat în cazul în care sala selectată inițial nu este disponibilă în acel moment.

Această lucrare va prezenta, în cele ce urmează, toate etapele care au stat la baza dezvoltării aplicației de la gândirea flow-ului inițial și baza sa de date până la API-urile folosite pentru integrarea funcționalităților sale.

## **Abstract**

The WEB application presented in this bachelor's thesis is called OfOps, its purpose being the management of shared spaces within a company, so that office activities can be optimized.

The sometimes insufficient seating space for all employees, the rush to arrive early to catch the preferred desk or even a parking lot are unpleasant experiences that were the basis of this application. The need to know the availability of desks, meeting rooms, and parking lots was an important starting point for me to develop OfOps in order to relieve us of the unnecessary worries.

The application features a calendar and a clock to select the desired time of availability, as well as interactive maps. These maps are designed to enhance the navigation experience and, on them, places are color-coded depending on their reservation status. The color palette used to mark them is: green – available place –, red – occupied – and finally, yellow – reservation has been made by the authenticated user–.

The booking process can be done with a simple click on the desired spot. However, if there is already another reservation for that place, the user will be notified about it and will not be able to make it. For meeting rooms, on the other hand, the user will receive a recommendation for an available meeting room for the desired time if the wanted meeting room is not available in the initial interval.

This paper will present all the stages that formed the basis of OfOps's development, from the initial design and its database structure to the APIs used for integrating its functionalities.

# Cuprins

<b>1 Introducere</b>	<b>6</b>
1.1 Motivația lucrării . . . . .	6
1.1.1 Problemă . . . . .	6
1.1.2 Scop . . . . .	6
1.1.3 Obiective . . . . .	7
1.2 Structura lucrării . . . . .	7
<b>2 Preliminarii</b>	<b>8</b>
2.1 Tehnologii folosite . . . . .	8
2.1.1 Stocarea datelor . . . . .	9
2.1.2 Backend . . . . .	10
2.1.3 Frontend . . . . .	11
2.2 Asigurarea securității . . . . .	12
2.3 Mijloace de testare . . . . .	16
<b>3 Prezentarea aplicației</b>	<b>20</b>
3.1 Flow-ul aplicației . . . . .	20
3.2 Diagrama E/R . . . . .	21
3.3 Pagina principală și mențiuni generale . . . . .	21
3.4 Partea de ADMIN . . . . .	22
3.5 Partea de USER . . . . .	24
3.5.1 Pagina de înregistrare a unui utilizator . . . . .	24
3.5.2 Pagina de Sign In . . . . .	26

3.5.3	Rezervarea unui loc . . . . .	27
3.5.4	Rezervarea unei săli de ședință . . . . .	31
3.5.5	Pagina My Reservations . . . . .	33
<b>4</b>	<b>Concluzii</b>	<b>35</b>
	<b>Bibliografie</b>	<b>36</b>

# Capitolul 1

## Introducere

### 1.1 Motivația lucrării

#### 1.1.1 Problemă

În cadrul firmei în care am efectuat practica vara trecută, prezența la birou mi s-a părut utilă pentru facilitarea colaborării în cadrul echipei de proiect, însă acest lucru venea și cu dezavantaje. Gândul că există posibilitatea să nu găsesc un birou liber lângă cei de la care învățam mă neliniștea întrucât relaționarea și comunicarea nu mai erau facile. Din cauza acestei probleme, încercam să ajung dimineața cât mai devreme astfel încât să găsesc un birou liber lângă cei cu care lucram în cadrul proiectului. Își nu numai eu aveam această problemă. Își colegii mei din departament se confruntau cu acest inconvenient. Deseori intervineau probleme legate de identificarea spațiilor de lucru, dificultăți care se extindeau și în zona locurilor de parcare, aspecte care generaun disconfort și o grija inutilă pentru a începe o nouă zi de muncă.

#### 1.1.2 Scop

Scopul OfOps este de a eficientiza și de a utiliza corespunzător resursele comune existente la nivelul firmei, iar motivația realizării aplicației vizează îmbunătățirea desfășurării activităților angajaților la birou creând un mediu propice atingerii obiectivelor firmei.

### **1.1.3 Obiective**

Obiectivele pe care și le propune aplicația OfOps să le îndeplinească sunt:

- Rezervarea birourilor și a locurilor de parcare, fără ca rezervările să se suprapună cu altele deja existente;
- Eficientizarea utilizării sălilor de ședințe;
- Sugerarea unei alternative în cazul în care sala este ocupată în intervalul dorit;
- Utilizarea hărților interactive pentru a face mai ușoară experiența utilizatorului.

## **1.2 Structura lucrării**

Lucrarea este structurată astfel:

1. Introducerea - prezintă problema, scopul și obiectivele OfOps;
2. Preliminarii - capitol despre tehnologiile folosite pentru dezvoltarea aplicației, securitatea sa și mijloacele de testare folosite;
3. Prezentarea aplicației - descrierea în detaliu a funcționalităților OfOps, alături de flow și diagrama E/R;
4. Concluzii - concluziile aplicației și posibilitatea de dezvoltare ulterioară;
5. Bibliografia - sursele folosite pentru preluarea informațiilor.

# Capitolul 2

## Preliminarii

### 2.1 Tehnologii folosite

Aplicația WEB reprezintă un program de tip software stocat pe un server remote, devenind accesibil pentru o masă mare de utilizatori care doresc să acceseze date rapid și ușor. Componentele sale pot comunica prin protocoale bazate pe internet cum ar fi HTTP (Hyper Text Transfer Protocol) sau HTTPS (Hyper Text Transfer Protocol Secure) [8]. Principalele avantaje ale aplicațiilor WEB le reprezintă accesibilitatea – accesarea lor se face folosind doar un browser WEB indiferent de circumstanțele externe (ora și loc) –, ușurința utilizatorului de a naviga pe platformă fără a avea nevoie de resurse suplimentare și scalabilitatea – poate gestiona un volum mare de date fără a se degrada –.

Aceasta are la bază arhitectura de tipul „client-server” în care clienții efectuează diferite operațiuni asupra serverului, iar răspunsurile serverului sunt primite prin intermediul internetului.

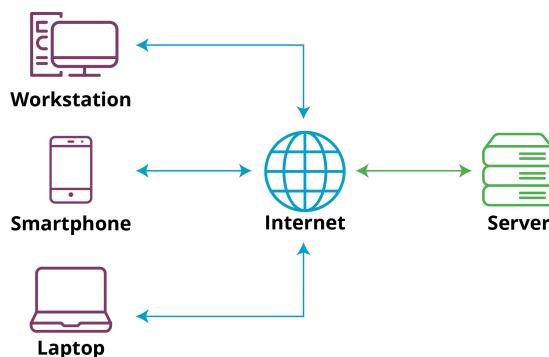


Figura 1: Comunicarea client-server

Aplicația OfOps a fost construită având partea de server dezvoltată cu ajutorul MySQL, Spring Boot și Java, iar partea de client a fost implementată folosind instrumente specifice frontend-ului și anume Angular, HTML (HyperText Markup Language), CSS (Cascading Style Sheets) și Typescript. Aceste tehnologii vor fi descrise în subcapitolele ce urmează.

### 2.1.1 Stocarea datelor

MySQL este un sistem pentru management-ul bazelor de date relationale reprezentând baze de date care stochează date în tabele separate [14]. Structura sa este menită să fie un instrument flexibil pentru programare, întrucât se pot seta reguli pe baza cărora să fie construită baza de date, relațiile între tabele (one-one, many-one, one-many, many-many), cheile primare sau străine, constrângeri etc. Utilizarea MySQL pentru stocarea bazei de date este una eficientă, întrucât modul în care această aplicație a fost construită nu permite inconsistență, duplicarea sau lipsa datelor.

Luând în considerare avantajele MySQL, am ales ca baza de date a aplicației OfOps să fie păstrată în MySQL Workbench versiunea 8.0 CE datorită volumului mare de date pe care îl suportă. Aceasta oferă o interfață intuitivă și ușor de utilizat, stocarea tabelelor este bine organizată și, printr-un singur click, ai la dispoziție atât detalii despre tabele, cât și datele păstrate în ele, făcând interacțiunea cu MySQL Workbench una facilă și rapidă.

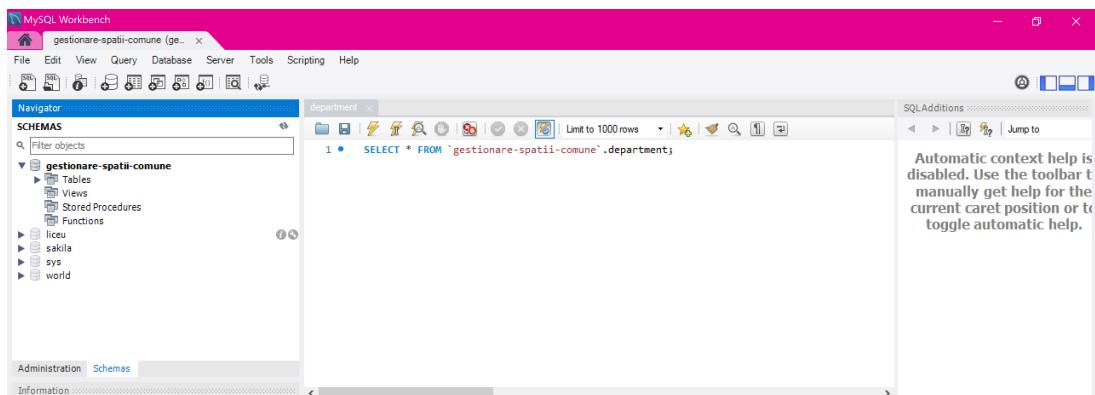


Figura 2: Interfață MySQL Workbench

## 2.1.2 Backend

Pentru implementarea backend-ului am utilizat Java și Spring Boot.

### Java

Java este un limbaj de programare orientat pe obiecte, high-level, creat în jurul anilor 1990. Până atunci, C și C++ erau cele mai răspândite limbaje de programare, însă utilizarea lor la scară largă era restrânsă și costisitoare. Motivația dezvoltării Java a reprezentat-o nevoia unui limbaj de programare care să poată fi folosit pentru diferite dispozitive electronice sau chiar pentru dispozitive cu control remote [11]. Versatilitatea și simplitatea acestui limbaj l-au adus printre cele mai populare și răspândite modalități de a coda, inclusiv și în prezent.

De asemenea, Java joacă un rol important și în implementarea acestei aplicații, întrucât majoritatea codului dezvoltat are la bază acest limbaj de programare. Codul a necesitat instalarea în prealabil a unui JRE (Java Runtime Environment), astfel încât compilarea să se realizeze cu succes.

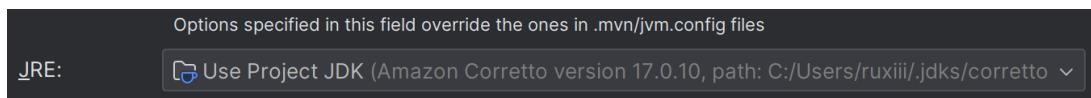


Figura 3: JRE utilizat

### Spring Boot

Spring este un framework care vine în ajutorul construirii rapide a unor aplicații stand-alone, lucrând cu alte librării deja integrate, ceea ce înseamnă că dezvoltarea aplicației nu va necesita o configurare foarte mare în plus [12].

Am ales Maven pentru adăugarea dependințelor, iar, pe lângă cele deja integrate în proiect, am mai adăugat: Spring-Boot-Starter-Data-JPA (interacțiune backend - baza de date), Spring-Boot-Starter-WEB (simplifică construirea aplicațiilor WEB cu Spring), MySQL-Connector-J (conexiunea cu baza de date), Lombok (generare de cod comun), Spring-Boot-Starter-Test (pentru testele unitare și de integrare), Spring-Boot-Starter-Security și Java-JWT (pentru securitatea aplicației).

### 2.1.3 Frontend

Pentru implementarea frontend-ului am utilizat Angular, HTML, CSS și TypeScript.

#### Angular

Angular este un framework pentru frontend care ajută la crearea unei aplicații de tipul single-page. Ce diferențiază Angular de celelalte framework-uri pentru frontend este structura sa bazată pe componente. Astfel, arhitectura codului este mai ușor de organizat, deoarece fiecare componentă are rolul ei bine definit, iar identificarea unei potențiale greșeli este mai ușor de găsit [3].

```
@Component({  
    selector: 'app-my-reservation',  
    templateUrl: './my-reservation.component.html',  
    styleUrls: ['./my-reservation.component.css']  
})  
export class MyReservationComponent {}
```

#### HTML

Expunerea componentelor din Angular se realizează în pagină cu ajutorul elementelor de HTML. Acesta oferă o structurare bine pusă la punct a conținutului paginii, existând diferențe între tipuri de text (titlu, bold, italic etc.), liste, tabele și multe alte elemente care fac experiența user-ului în aplicație mai placută.

```
<h1>Account created successfully!  
<br/> Don't forget your user ID for login!</h1>
```

#### CSS

Înfrumusețarea aplicației și, implicit, a tag-urilor HTML se efectuează cu ajutorul modurilor de stilizare din CSS. Ele au scopul să creeze un aspect vizual inedit al aplicației, raportându-ne de la culorile de background ale aplicației până la mici elemente de finețe cum ar fi alerte sau butoane.

```
h1{  
    text-align: center;  
    font-size: 40px;  
    font-weight: bold;  
}
```

### TypeScript

TypeScript este cel mai important limbaj pentru dezvoltarea unei aplicații în Angular. Multitudinea avantajelor acestuia precum folosirea interfețelor și claselor, structura simplă și ușor de înțeles, îmbinarea caracteristicilor de JavaScript cu declararea statică a tipurilor de date fac TypeScript-ul temelia unei aplicații în Angular.

```
export class DepartmentsComponent {  
    departmentId: string;  
    departmentName: string;  
}
```

## 2.2 Asigurarea securității

Securitatea OfOps se menține la două niveluri:

- Autentificarea propriu-zisă în aplicație printr-un sistem de tipul user-parolă;
- Accesibilitatea datelor este limitată pentru fiecare rol, oferind doar anumite opțiuni de utilizare a aplicației în funcție de rolul atribuit.

Acste lucruri s-au realizat la nivel de implementare cu ajutorul JWT (JSON Web Token) care a rezolvat atât problema de autentificare, cât și pe cea de autorizare.

JWT este un standard bazat pe transmiterea în siguranță a unor date care, prin simplitatea, siguranța și versatilitatea sa, a ajuns să fie folosit în cele mai mari framework-uri web [10]. Token-ul este format dintr-un header, un payload și o semnătură/ datele criptate. Primele două sunt obiecte de tip JSON cu o structură deja definită, însă ultima parte

depinde în mod direct de tipul de criptare utilizat. Mai întâi, din browser, se trimite către backend, printr-o metodă HTTP de tipul POST pe endpoint-ul de /login, un username și o parolă pentru care server-ul crează un JWT. Acest token este trimis înapoi la frontend, moment în care acesta trimite înapoi header-ul pentru autorizare. Se verifică semnătura și se preiau detaliile utilizatorului și se trimit răspunsul, astfel, către client.

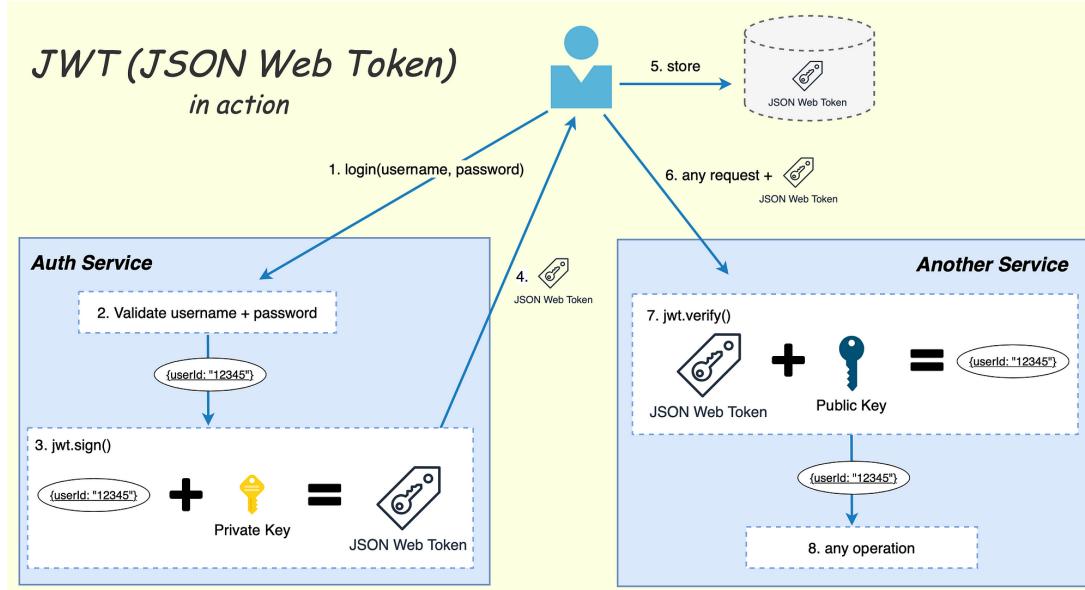


Figura 4: Modul de lucru al JWT

Folosirea token-ului pentru autentificare se face în modul următor:

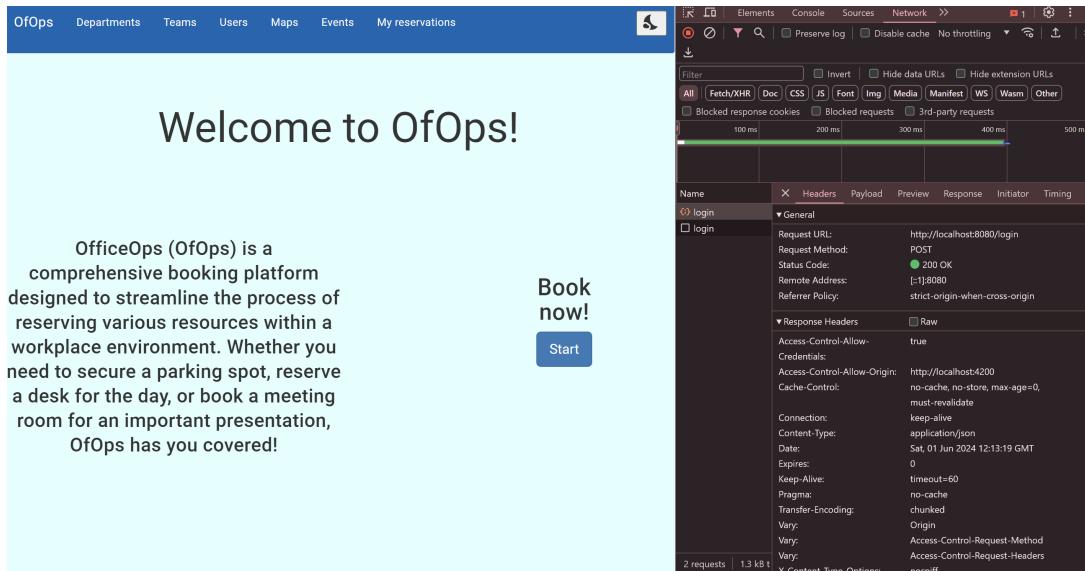


Figura 5: Autentificare

Iar răspunsul primit de la backend este de forma (tot din motive de securitate, valorile

pentru parola și token nu sunt cele reale):

```
{  
    "authentication": {  
        "userId": "T0",  
        "userPassword": "$2a$10$WTkxlkyVeNAxnbHVdY  
WiOtzdv.aRlPnWAFA/5MIW",  
        "authorities": [ {  
            "roleId": 1,  
            "authority": "ADMIN"  
        } ],  
        "password": "$2a$10$WTkxlkyVeNAxnbHVdY  
WiOtzdv.aRlPnWAFA/5MIW",  
        "enabled": true,  
        "username": "T0",  
        "accountNonExpired": true,  
        "credentialsNonExpired": true,  
        "accountNonLocked": true  
    },  
    "token": "eyJhbGciOiJSUzI1NiJ9.eyJpcRNSU4ifQ.I8x0c3gG85g  
4PVOp-NqPU65pW1PiOyFljylKL6QpCIe5Em-JTsCsDg71wr0omEIsWZ  
igqpy4BqpfCAknt2i97fMR5DYtBqen9CA"  
}
```

Se poate observa, de asemenea, faptul că, în răspunsul returnat, valoarea parolei nu este cea reală (cea tastată de utilizator), ci este valoarea criptată a parolei. Pentru acest lucru, am utilizat interfața **PasswordEncoder** oferită de pachetul Spring-Boot-Starter-Security. În momentul în care un user încearcă să se logheze pe aplicație, va face un request de tip POST către backend. Astfel, se va apela metoda **loginUser** din **LoginService** care va verifica dacă parola introdusă de utilizator este aceeași cu cea stocată în baza de date, prin intermediul metodei **matches** tot din cadrul interfeței **PasswordEncoder**. În

situația în care parolele sunt la fel, user-ul va avea drepturi în aplicație, altfel va primi un warning și nu va fi lăsat să se logheze în OfOps.

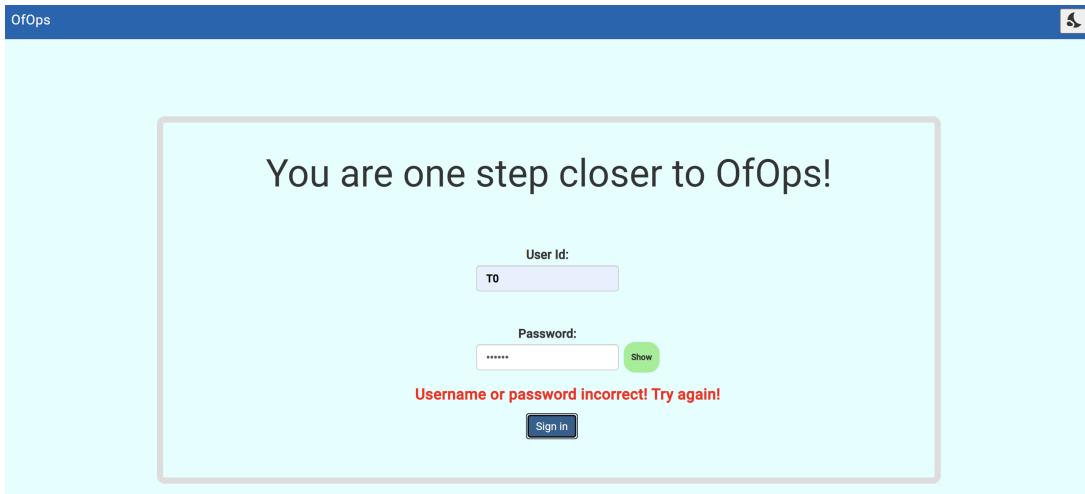


Figura 6: Autentificare incorectă

Acest lucru ne aduce la al doilea nivel de securitate al aplicației și anume acela de autorizare. Autorizarea se realizează cu ajutorul tableei **Roles** în care sunt stocate rolurile de **ADMIN** și **USER**. **ADMIN-UL** are acces complet în aplicație, în timp ce **USER-UL** poate doar să rezerve spațiul de care are nevoie și să-și vadă propriile rezervări. Diferența dintre roluri este cel mai bine evidențiată la nivelul butoanelor din **Navbar**, întrucât acestea diferă atât pentru cele două roluri, cât și dacă utilizatorul nu este logat pe aplicație.

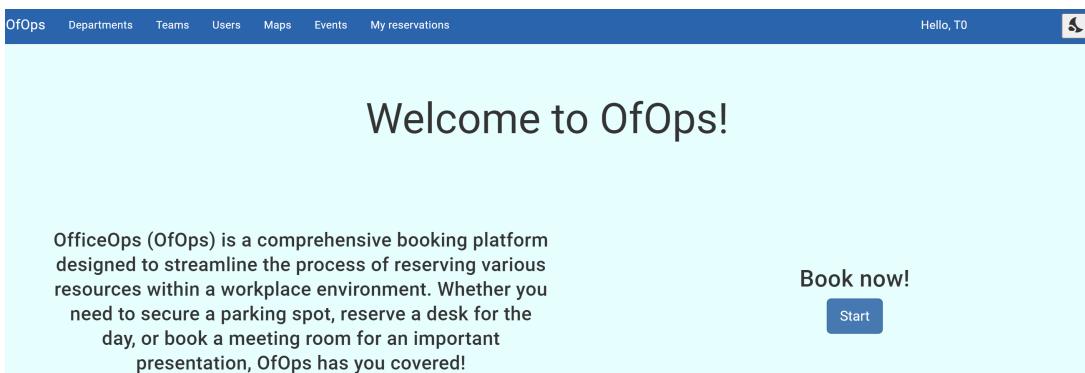


Figura 7: Autentificare admin



Figura 8: Autentificare utilizator

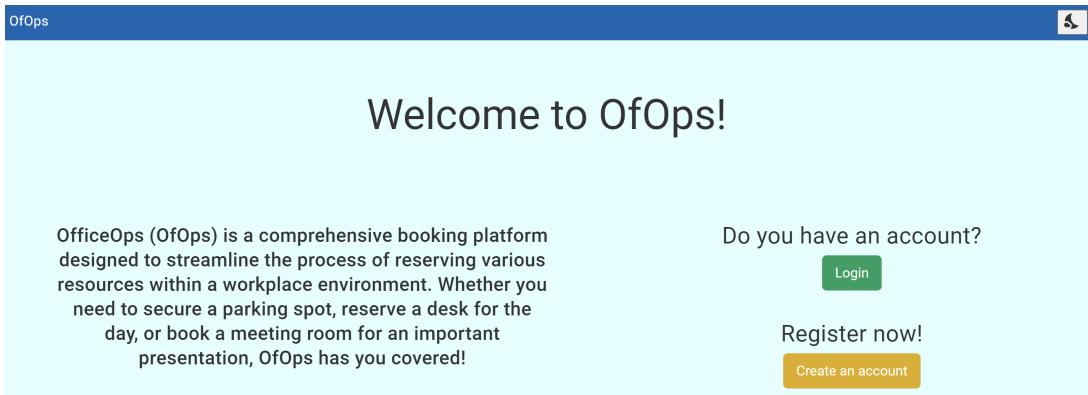


Figura 9: Nu există autentificare

## 2.3 Mijloace de testare

Până la crearea frontend-ului, partea de backend a fost testată cu ajutorul testelor unitare, de integrare și cu ajutorul aplicației Postman.

Testele unitare contribuie la crearea de la zero și dezvoltarea unui proiect durabil [6]. Importanța lor este dată de faptul că, folosindu-le, observarea potențialelor greșeli în scrierea de cod este iminentă. Practic, ele urmăresc flow-ul metodei și sunt scrise cu scopul de a vedea cum reacționează codul în diferite cazuri.

```
private DepartmentService departmentService;
```

```
@Mock
```

```
private DepartmentRepository departmentRepository;
```

```

@BeforeEach
public void setUp(){
    MockitoAnnotations.initMocks( this );

    departmentRepository = Mockito.mock
        ( DepartmentRepository . class );
    departmentService = new DepartmentService
        ( departmentRepository );
}

@Test
public void deleteDepartmentTest_department_not_found() {
    when( departmentRepository . existsById( any() ) )
        . thenReturn( false );
    assertThrows( DepartmentNotFoundException . class ,
        () -> departmentService . deleteDepartment( any() ) );
}

@Test
public void deleteDepartmentTest_department_found()
    throws DepartmentNotFoundException {
    when( departmentRepository . existsById( any() ) ) .
        thenReturn( true );
    departmentService . deleteDepartment( any() );
    Mockito . verify( departmentRepository , Mockito . times( 1 ) )
        . deleteById( any() );
}

```

Am folosit adnotarea `@Mock` prin care se simulează interacțiunea cu baza de date, astfel încât executarea testelor să nu afecteze în mod direct intrările bazei de date. Acest lucru se întâmplă înaintea fiecărui test executat în metoda `setUp`. Primul test verifică faptul că, în cazul în care un departament care se dorește a fi șters din baza de date nu există, codul va arunca excepția `DepartmentNotFoundException`. Al doilea test simulează comportamentul în care un departament există și trebuie șters. Astfel, verificăm ca metoda `deleteById(any())` este apelată, executându-se ștergerea. Ușurința acestor teste unitare o reprezintă faptul că, în esență, nu ne interesează tipul de date cu care este apelată metoda, ci doar cum se comportă codul și dacă acesta urmărește flow-ul corespunzător. De aceea, toate metodele au ca parametru `any()`.

Am realizat și câteva teste de integrare pentru partea de departament. Testele de integrare oferă un mijloc de verificare a testelor unitare [1], dar cu date reale, mai aproape de posibilele date primite de la frontend.

```
private DepartmentEntity createDepartmentEntity(){
    DepartmentEntity departmentEntity= new DepartmentEntity();
    departmentEntity.setDepartmentId ("HR");
    departmentEntity.setDepartmentName ("Human Resources");
    return departmentEntity;
}

@Test
public void getDepartmentByIdTest_departments_found(){
    DepartmentEntity departmentEntity=
        createDepartmentEntity ();
    List<DepartmentEntity> departmentEntities =
        new ArrayList<>();
    departmentEntities.add(departmentEntity);
```

```

when( departmentRepository . findAll () )
    . thenReturn ( departmentEntities );

departmentService . getDepartments ();
Assertions . assertEquals (
    departmentService . getDepartments () . size () , 1 );
Assertions . assertEquals ( departmentService . getDepartments ()
    . get ( 0 ) . getDepartmentId () , "HR" );
Assertions . assertEquals ( departmentService . getDepartments ()
    . get ( 0 ) . getDepartmentName ()
    , "Human Resources " );
}

}

```

De asemenea, backend-ul a fost testat înainte și prin request-uri HTTP în Postman către portul 8080 pe care rula aplicația. Postman este o aplicație cu interfață deja integrată care vine în ajutorul programatorului în cazul în care dezvoltă backend-ul unei aplicații de la zero, fără a avea un frontend disponibil în care să testeze interacțiunea cu baza de date.

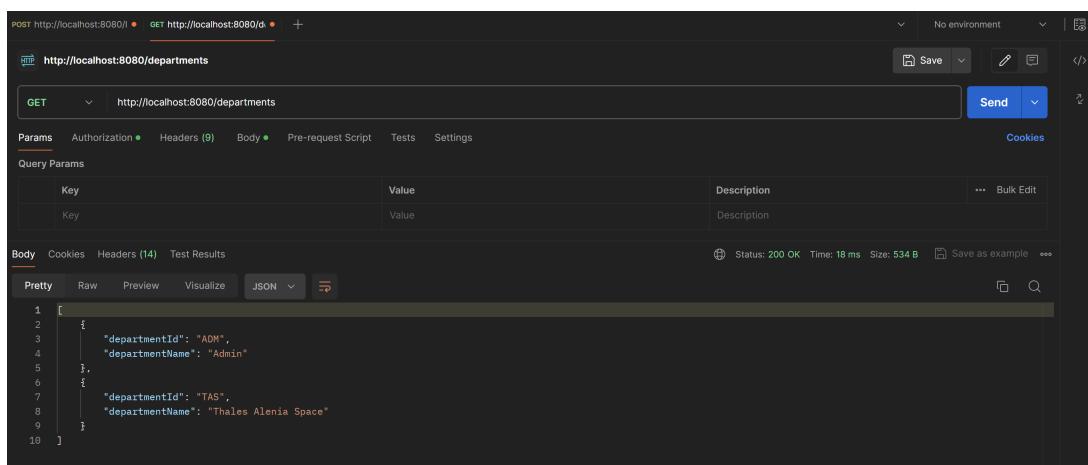


Figura 10: Testare Postman

# Capitolul 3

## Prezentarea aplicației

OfOps este o aplicație WEB concepută pentru simplificarea procesului de rezervare a diferitelor resurse dintr-un mediu de lucru. Indiferent dacă aveți nevoie să vă asigurați un loc de parcare, să rezervați un birou pentru o anumită perioadă de timp sau o sală de ședință, OfOps vine în ajutorul dumneavoastră!

### 3.1 Flow-ul aplicației

Navigarea pe OfOps urmează următorul curs:

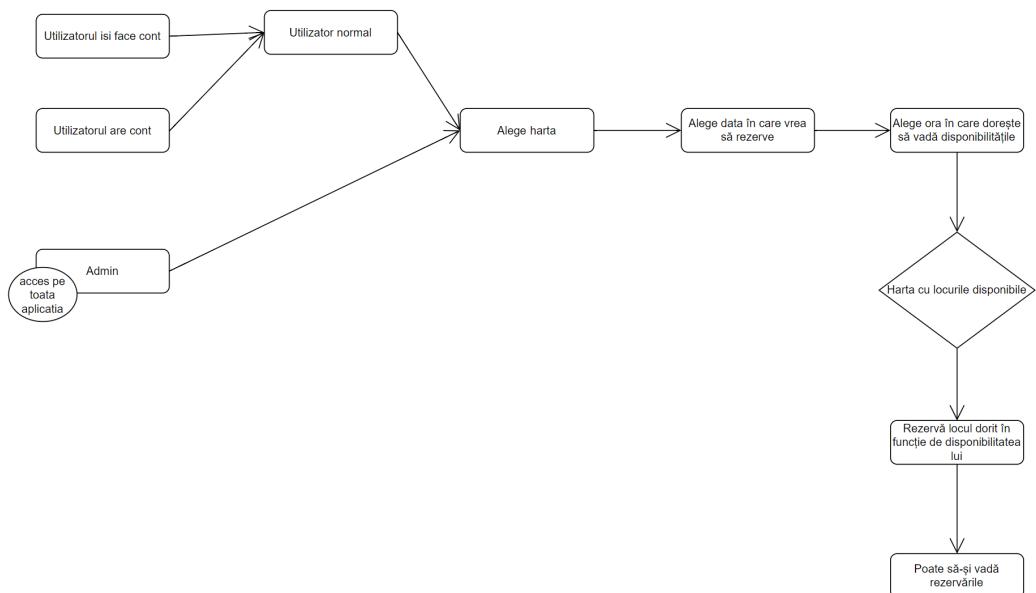


Figura 1: Flow OfOps

## 3.2 Diagrama E/R

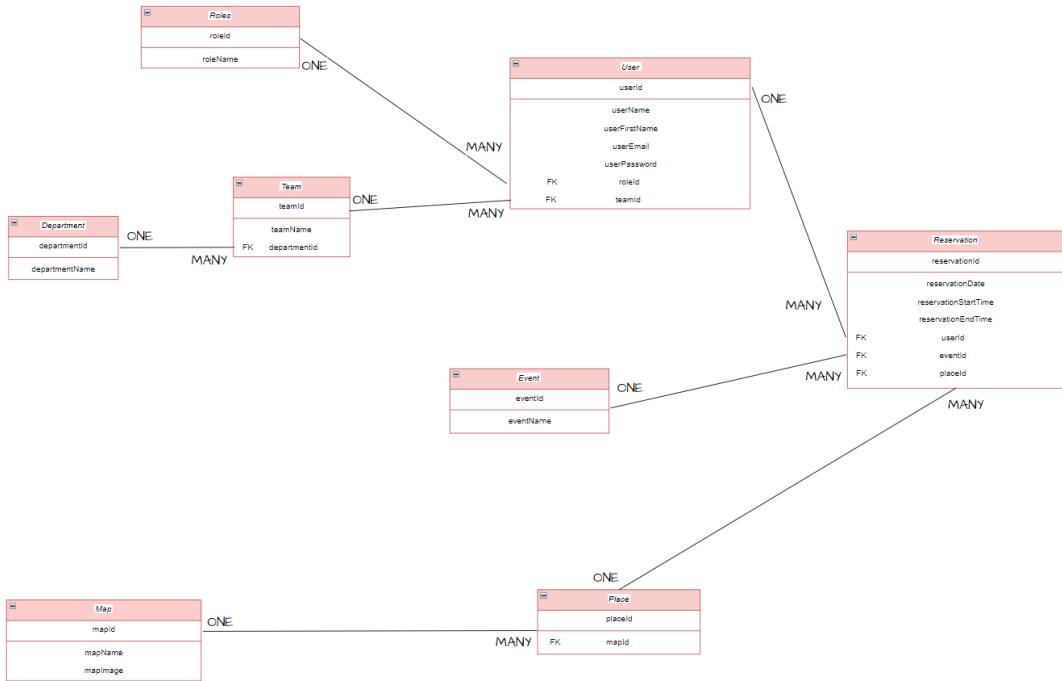


Figura 2: Diagrama E/R

## 3.3 Pagina principală și mențiuni generale

Pagina principală este cea care întâmpină utilizatorul în momentul în care navighează pentru prima oară pe OfOps. Aceasta oferă opțiuni limitate utilizatorului și anume cele de **Login** și **Create an account**.

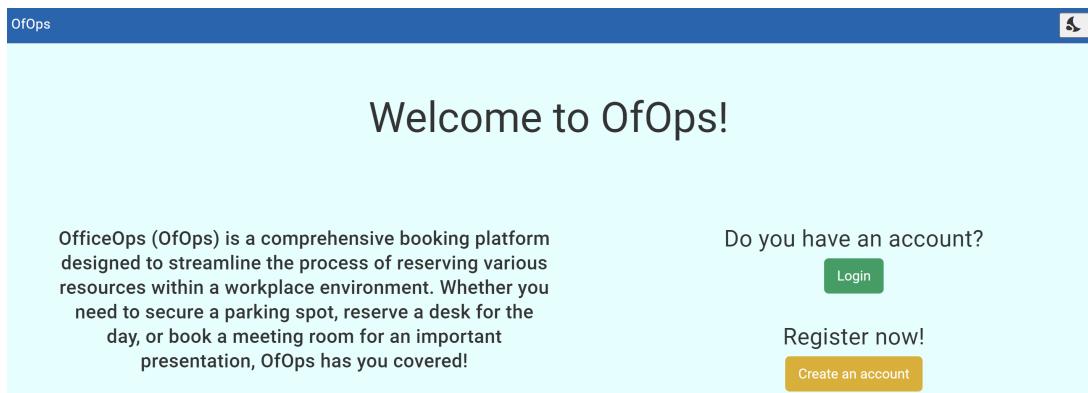


Figura 3: Pagina principală

De asemenea, OfOps dispune și de varianta pentru dark mode. Această opțiune, cunoscută și sub denumirea de night mode, a devenit populară în ultimii ani, mai exact, din anul 2019 în care Google a introdus-o pe Android OS, fapt ce i-a făcut pe toți gigantii din industria IT să-i urmeze modelul [5]. Posibilitatea utilizatorului de alege modul de navigare pe aplicație denotă prioritarea dorințelor acestuia, rezultând astfel o experiență mult mai plăcută pentru el. Schimbarea la dark mode se realizează prin apăsarea butonului din dreapta sus a paginii.

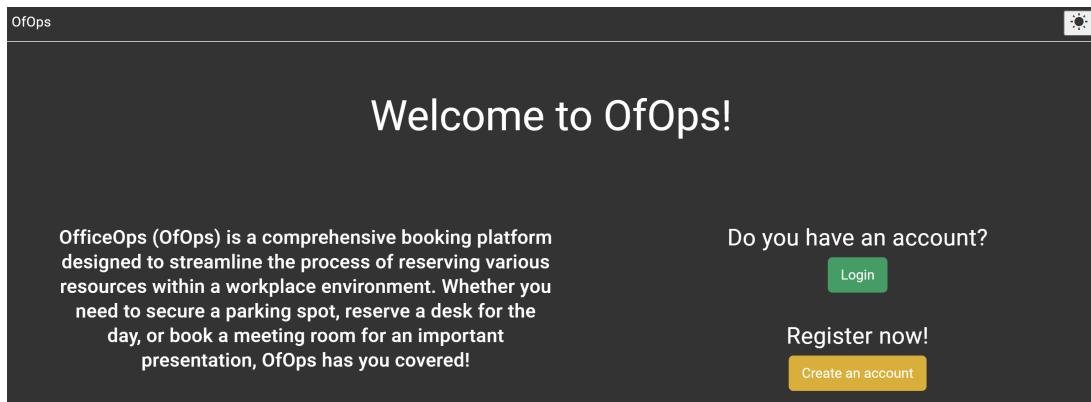


Figura 4: Dark mode

### 3.4 Partea de ADMIN

Partea de **ADMIN** a OfOps are același curs ca partea de **USER**, însă dispune de câteva funcționalități în plus.



Figura 5: Navbar pentru ADMIN

Rolurile adminului sunt:

- **Departments:** poate adăuga un nou departament, edita sau șterge unul deja existent;
- **Teams și Users** se comportă ca **Departments**: în plus la **Users**, în cazul în care un user dorește să-și schimbe parola, acesta va trebui să facă o solicitare adminului aplicației pentru a-și putea schimba parola;

OfOps Departments Teams Users Maps Events My reservations Hello, T0

### Create a department

Department Id: HR

Department Name: Human Resources

**Submit**

Figura 6: Adăugarea unui departament

OfOps Departments Teams Users Maps Events My reservations Hello, T0

Department Id	Department Name	Actions
ADM	Admin	<b>Update</b> <b>Delete</b>
HR	Human Resources	<b>Update</b> <b>Delete</b>
LAS	Land and Air Systems	<b>Update</b> <b>Delete</b>
NAV	Naval	<b>Update</b> <b>Delete</b>
SIX	Secure communications and protection systems	<b>Update</b> <b>Delete</b>
TAS	Thales Alenia Space	<b>Update</b> <b>Delete</b>

« Previous 1 Next »

New department

Figura 7: Lista departamentelor

OfOps Departments Teams Users Maps Events My reservations Hello, T0

### Create a department

Department Id: HR

Department Name: Human Resources

This department already exists!

**Submit**

Figura 8: Adăugarea unui departament deja existent

- **Maps:** poate adăuga o hartă nouă, pe lângă cele deja existente care trebuie ulterior configurată; aceasta va fi disponibilă în dropdown-ul de hărți;
- **Events:** poate adăuga un eveniment nou pe lângă cele deja existente (Daily booking - pentru rezervarea zilnică a unui birou sau loc de parcare, Meeting / Training - pentru rezervarea unei săli de ședință, Visit - pentru rezervarea unor birouri pentru persoane dintr-o delegație străină).

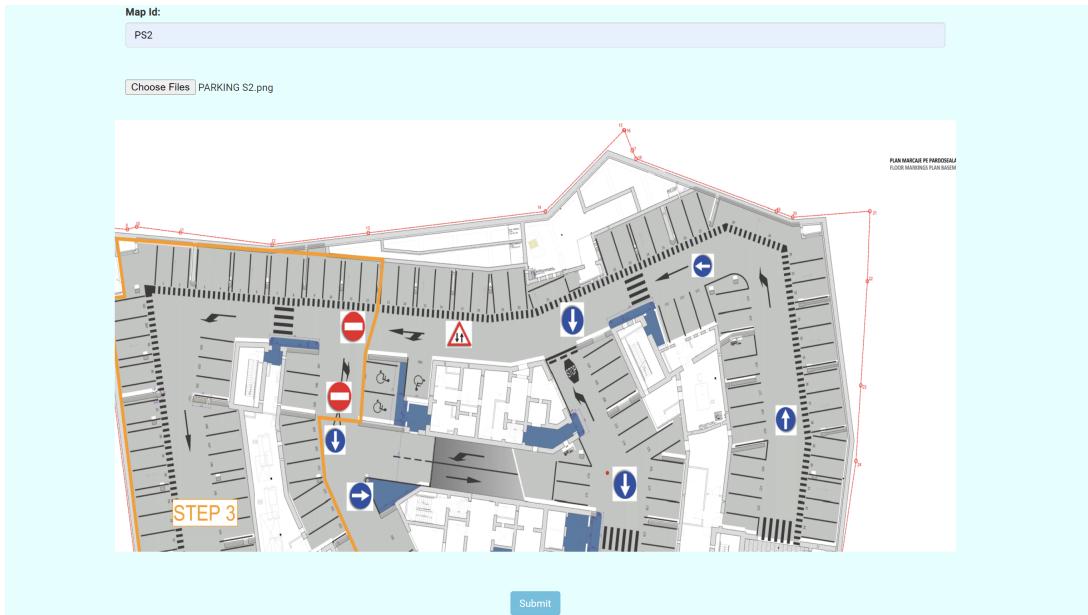


Figura 9: Adăugarea unei hărți

The screenshot shows the event management section of the application. At the top, there is a navigation bar with links for 'OfOps', 'Departments', 'Teams', 'Users', 'Maps', 'Events', and 'My reservations'. On the far right of the navigation bar, it says 'Hello, T0' and has a user icon. Below the navigation bar is a large rectangular container with a light blue background and a thin brown border. Inside this container, the word 'Events' is centered in a bold black font. Below 'Events' is a green button labeled 'New event'. Underneath the 'New event' button is a list of event types: 'Daily booking', 'Meeting / Training', and 'Visit', each listed in a separate row with a thin gray border.

Figura 10: Lista evenimentelor și butonul New event

## 3.5 Partea de USER

### 3.5.1 Pagina de înregistrare a unui utilizator

Pagina de înregistrare a unui utilizator este disponibilă în momentul click-ului pe butonul **Create an account** din pagina principală. Utilizatorul este direcționat către formularul de înregistrare.

The screenshot shows the 'Create an account' page of the OfOps application. The form fields are as follows:

- Name:** Enter your name
- First Name:** Enter your first name
- Team:** A dropdown menu showing 'CAL - TAS'
- Email:** Enter your email
- Password:** Enter your password with a 'Show' button next to it.
- Confirm password:** Re-enter your password with a 'Show' button next to it.

A sidebar on the left lists password requirements: > 6 characters, capital letters, lower letters, numbers, special characters, and passwords match.

At the bottom right is a blue 'Submit' button.

Figura 11: Înregistrarea pe OfOps

Crearea unui cont se realizează prin completarea tuturor câmpurilor. Acestea au implementate validări pe care dacă user-ul nu le îndeplinește va duce la inactivarea butonului de **Submit**. În caz contrar, butonul **Submit** se va activa. Totodată, există un set de reguli pentru parolă, astfel încât aceasta să fie cât mai greu de spart. Utilizatorul are posibilitatea să își vadă parola în cazul în care nu se potrivește cu cea trecută la **Confirm password**.

The screenshot shows the 'Create an account' page of the OfOps application with validation errors:

- Name:** Popescu (highlighted in light purple)
- First Name:** Please enter your first name! (highlighted in red)
- Team:** CAL - TAS
- Email:** ruxandra.iftimi (highlighted in red)
- Password:** ..... (highlighted in red)
- Confirm password:** ..... (highlighted in red)

A sidebar on the left lists password requirements: > 6 characters, capital letters, lower letters, numbers, special characters, and passwords match.

At the bottom right is a blue 'Submit' button.

Figura 12: Validările câmpurilor nu sunt respectate

La apăsarea butonului de **Submit**, user-ul va fi redirectionat către pagina de **Sign in**, primind un pop-up legat de viitoarea autentificare în aplicație.

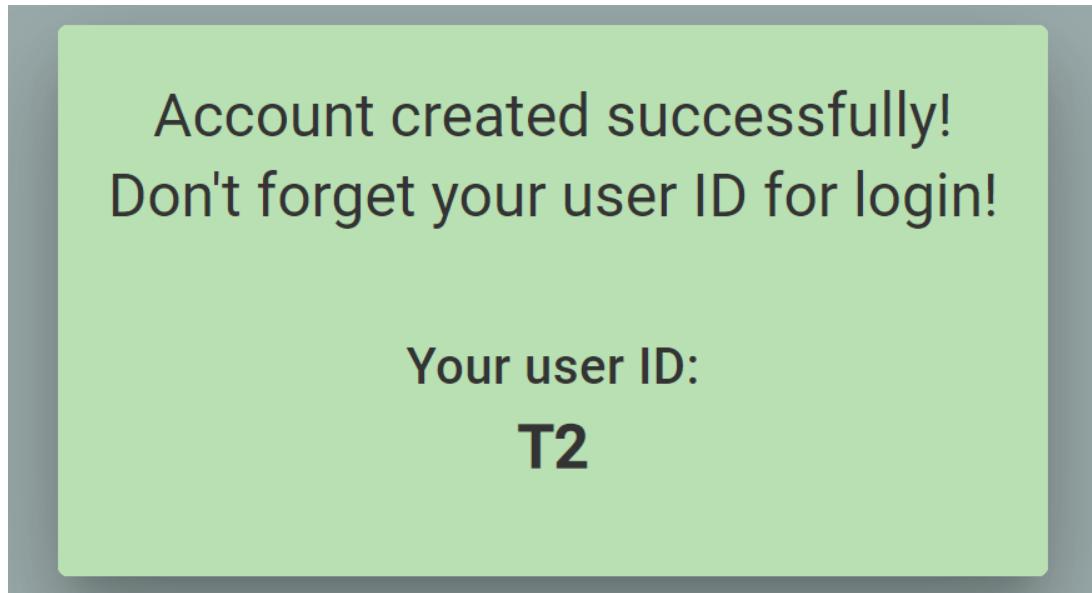


Figura 13: Înregistrare reușită

### 3.5.2 Pagina de Sign In

Pagina de **Sign in** este locul în care utilizatorul se loghează pe OfOps.

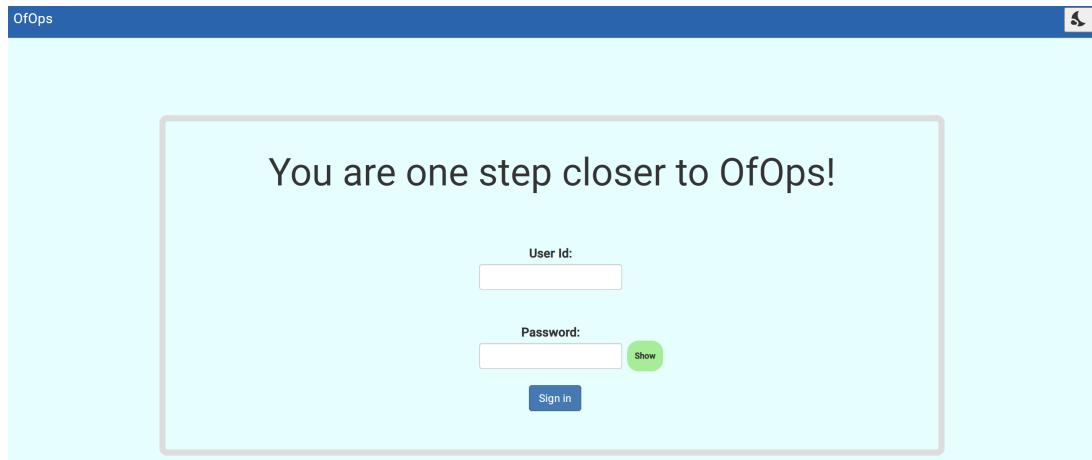


Figura 14: **Sign in**

De asemenea, după logarea în aplicație, utilizatorul are posibilitatea de logout.

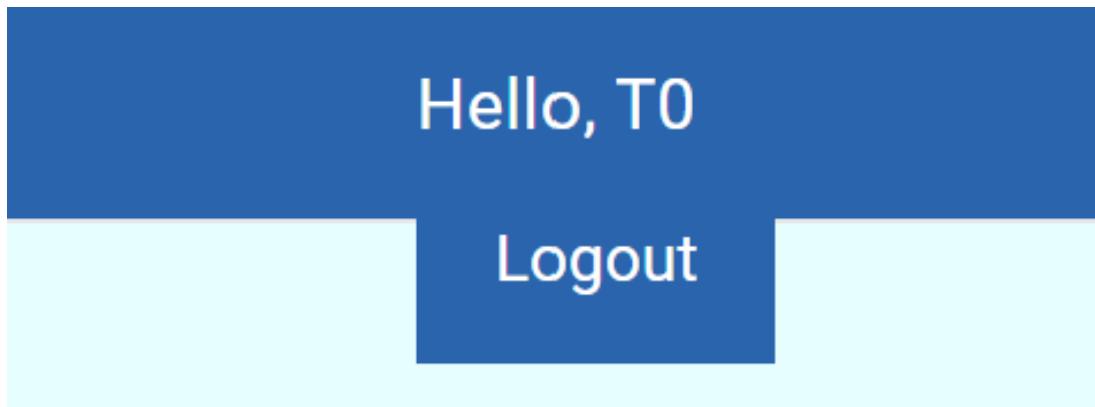


Figura 15: **Logout**

### 3.5.3 Rezervarea unui loc

Rezervarea unui loc este principala funcționalitate a aplicației. Aceasta începe prin apăsarea butonul de **Start** din pagina principală după ce utilizatorul s-a autentificat sau apăsând pe butonul **Maps**. Ulterior, user-ul va fi redirecționat către fereastra din care poate alege una dintre hărțile din dropdown.

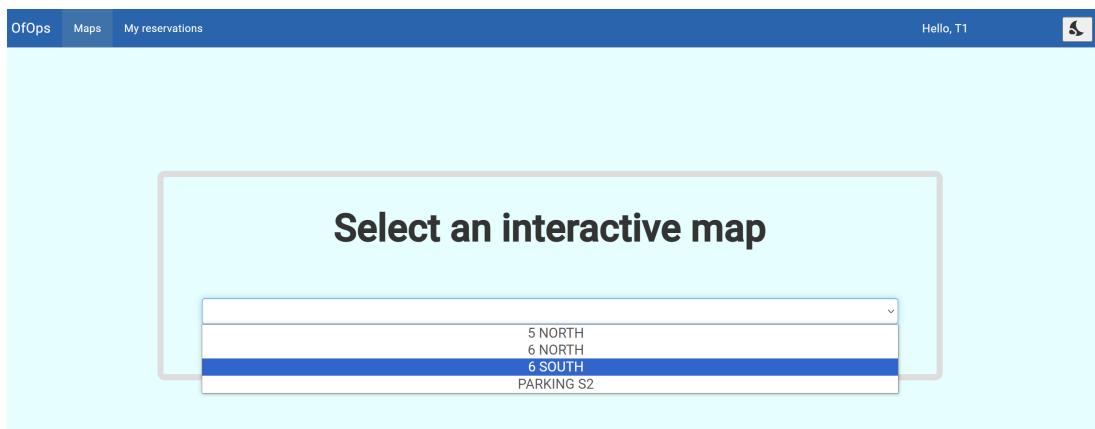


Figura 16: Hărțile disponibile

Odată selectată o hartă, va trebui aleasă ziua în care se dorește a fi făcută rezervarea. Din calendarul apărut, utilizatorul nu poate alege zilele de weekend sau zilele din trecut, întrucât reținerea locurilor în trecut sau în zilele de sămbătă și duminică este inutilă.

Selectarea datei este însoțită de un pop-up pentru exprimarea dorinței de a rezerva în acea zi pe care utilizatorul trebuie să o confirme sau nu, după preferințele sale.



Figura 17: Calendarul lunii mai

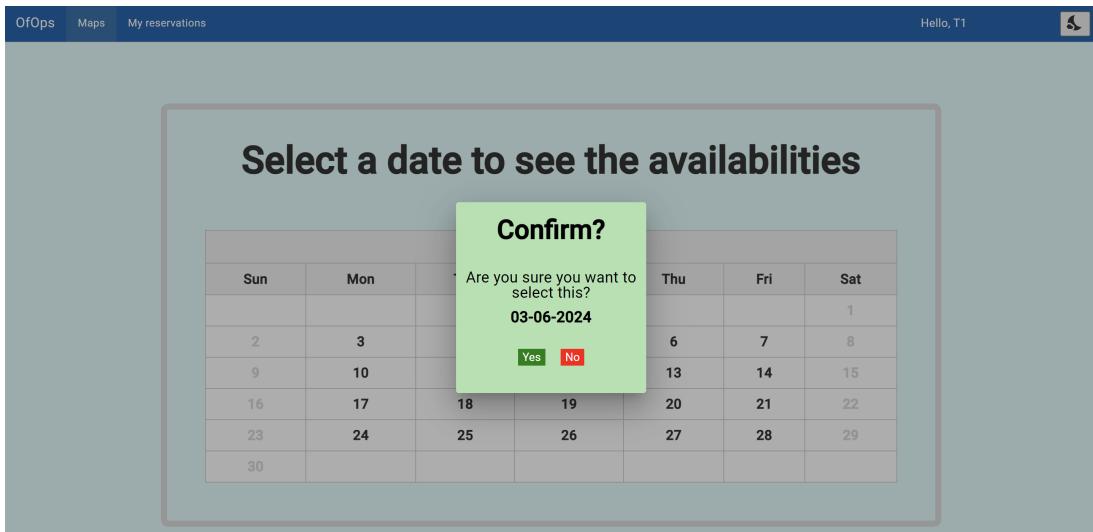


Figura 18: Pop-up pentru confirmare

Ulterior, se va face redirectarea către ceasul de unde se va alege momentul la care utilizatorul vrea să vadă disponibilitățile. Mai întâi se selectează ora. Click-ul pe ora dorită va declanșa apariția butoanelor AM-PM, iar alegerea formatului va duce la apariția minutelor. Am decis ca implementarea butoanelor să fie pentru fiecare minut, întrucât pot exista momente în care se dorește o verificare a locurilor libere într-un moment actual, în care ora nu este fixă. Alegerea completă a timpului duce la apariția butonului de **Choose time**, însotit de rezultatul selectării timpului.

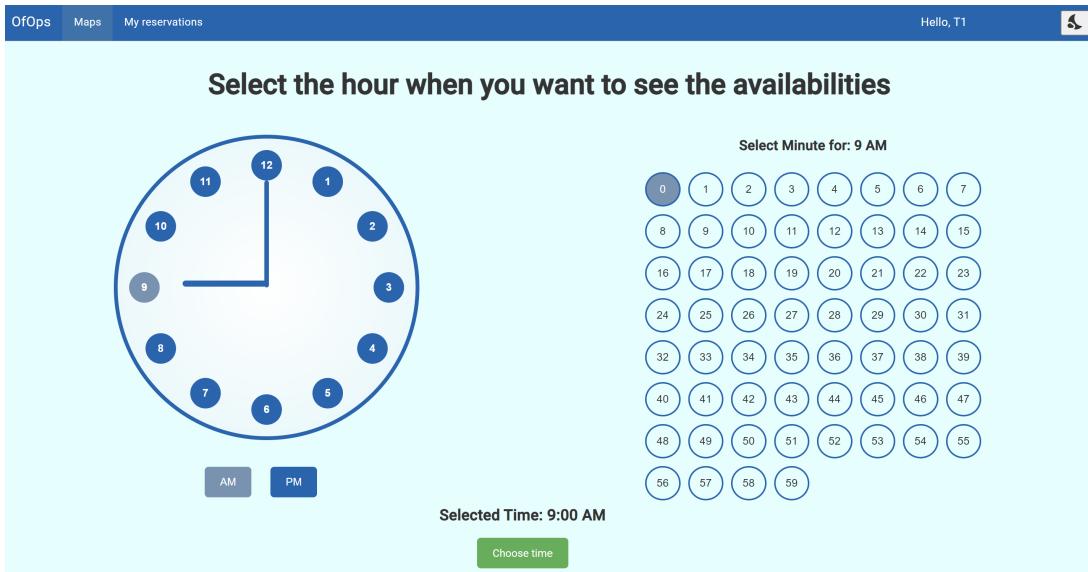


Figura 19: Selectarea corectă a timpului

Toți acești pași ne conduc către harta interactivă pentru alegerea unui birou sau a unui loc de parcare în funcție de preferință.

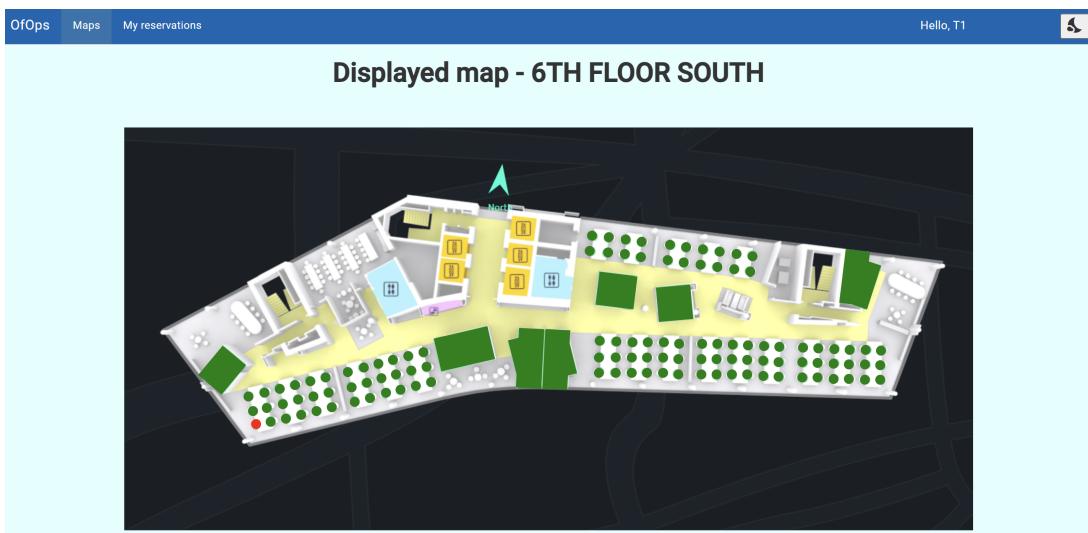


Figura 20: Harta interactivă pentru etajul 6 SUD

Printr-o privire foarte rapidă asupra hărții, se poate remarcă că o bulină (reprezentând un birou) este roșie. Aceasta semnifică faptul că biroul este rezervat în momentul în care utilizatorul a ales să verifice disponibilitatea. Bulinele verzi indică faptul că nu există o rezervare la ora dorită pentru birourile respective. Fiind harta interactivă, alegerea unui loc se face apăsând direct pe biroul respectiv.

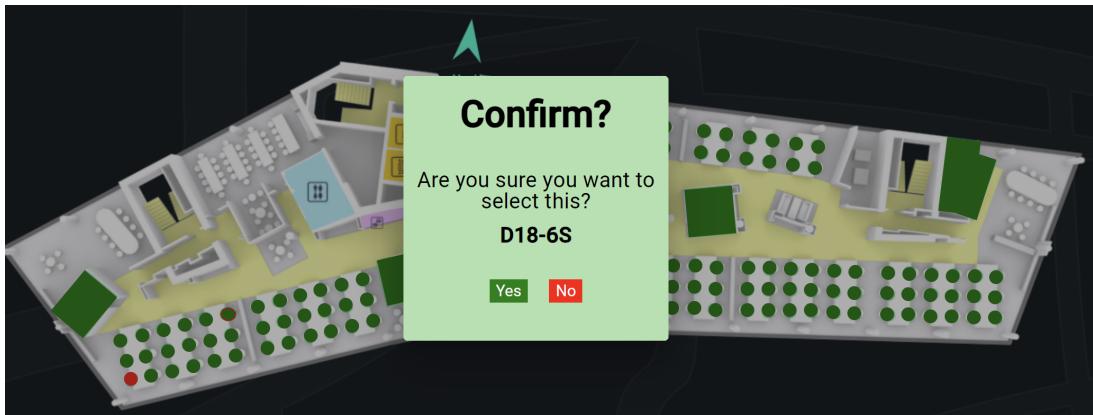


Figura 21: Confirmare loc

Înainte de rezervarea propriu-zisă, utilizatorului îi va apărea un pop-up de confirmare pentru selectarea locului. Se poate observa, de asemenea, că locul selectat este înconjurat de o linie punctată, astfel încât user-ul să vadă precis ce loc dorește să rezeze înainte de a apăsa butonul **Yes**.

În cazul în care există o rezervare viitoare pentru locul selectat, utilizatorul va primi o notificare în acest sens. Dacă utilizatorul alege să selecteze acest birou, rezervarea sa trebuie făcută în cunoștință de cauză sau poate alege alt loc.

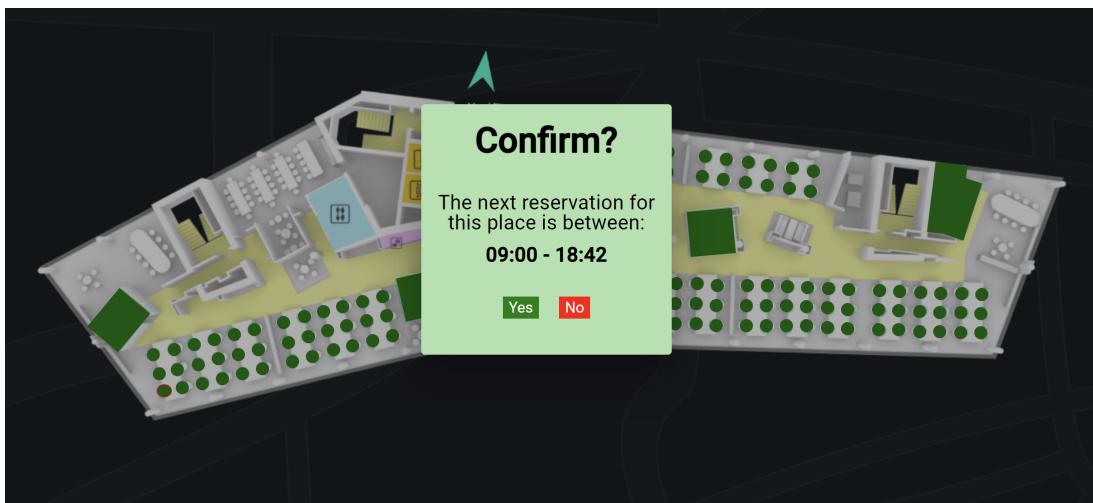


Figura 22: Notificare rezervare următoare

Rezervarea unui birou se realizează în pagina dedicată acesteia. Câmpurile de completat sunt cele de **Start Time**, **End Time** și **Event**. **Selected Date** nu se poate modifica, având rolul de a-i aminti user-ului data selectată. Dacă acesta nu mai dorește o rezervare în ziua selectată, trebuie să reia întreg procesul de rezervare. Butonul de **Submit** nu

se va activa până când datele nu sunt completate, evitând, astfel, introducerea în baza de date a unor câmpuri goale. Dacă totul este în regulă, se poate efectua rezervarea, iar utilizatorul va fi redirectionat către pagina de **My reservations** care va fi detaliată într-un subcapitol următor.

The screenshot shows the 'Book D18-6S' page. At the top, there are navigation links: 'OfOps', 'Maps', and 'My reservations'. On the right, it says 'Hello, T1' and has a user icon. Below the title, 'Selected Date:' is set to '03-06-2024'. Under 'Start Hour:' and 'End Hour:', the times are '09:00 AM' and '06:00 PM' respectively. A note below states 'You're booking for 9 hours and 0 minutes.' The 'Event:' dropdown is set to 'Daily booking'. At the bottom is a blue 'Submit' button.

Figura 23: Rezervare corectă

Există posibilitatea ca rezervarea pe care dorește să o facă utilizatorul în momentul curent să se suprapună cu o altă rezervare deja făcută pentru același loc. OfOps nu va permite întâmplarea acestui lucru! Încercarea de a rezerva totuși va duce la apariția unui mesaj care va indica ora următoarei rezervări, iar butonul de **Submit** se va dezactiva.

The screenshot shows the 'Book D1-6S' page. The layout is similar to Figura 23, with 'OfOps', 'Maps', and 'My reservations' at the top, and 'Hello, T1' on the right. 'Selected Date:' is '03-06-2024'. 'Start Time:' is '08:00 AM' and 'End Time:' is '10:00 AM'. A note says 'You're booking for 2 hours and 0 minutes.' The 'Event:' dropdown is 'Daily booking'. A red message at the bottom states 'The next reservation for this place is at 09:00!' Below this message is a disabled 'Submit' button.

Figura 24: Rezervare incorectă

### 3.5.4 Rezervarea unei săli de ședință

Rezervarea unei săli de ședință urmează același curs ca rezervarea unui birou sau al unui loc de parcare, însă are ceva în plus față de celealte. Atunci când un utilizator

încearcă să rezerve o sală de ședință, dacă intervalul dorit se suprapune chiar și un minut cu o altă rezervare, user-ul va primi ca alertă o propunere de sală liberă în perioada dorită. Acest lucru scoate OfOps în evidență, întrucât dispune de o metodă care eficientizează programarea sălilor de ședință. Acest algoritm implică obținerea tuturor sălilor de ședință și calcularea unui scor de disponibilitate după care sălile sunt sortate. Metoda de calculare a punctajului de disponibilitate se rezumă la numărul de rezervări existente care se suprapun cu intervalul de timp specificat. Scorul se va incrementa doar dacă există suprapunere între rezervări, sortarea crescătoare oferindu-ne, astfel, prima opțiune cu scorul cel mai mic. Utilizatorul poate alege varianta propusă, apăsând **OK** sau poate refuza propunerea prin apăsarea **Cancel**, fiind redirectionat către pagina de alegere a hărții. Pentru exemplu, în intervalul 09:00 - 10:00, voi încerca să rezerv sala **Sicily-6S** care este ocupată între 09:30 - 10:30.



Figura 25: Sicily-6S rezervată între 09:30-10:30

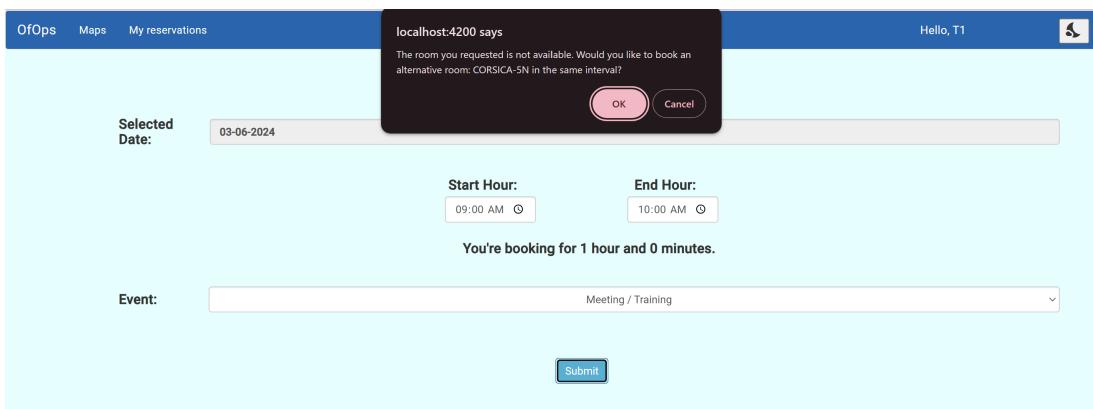


Figura 26: Propunerea de sală liberă în intervalul dorit

Dacă răspunsul este afirmativ, utilizatorul primește o alertă cum că rezervarea sa a fost realizată cu succes.

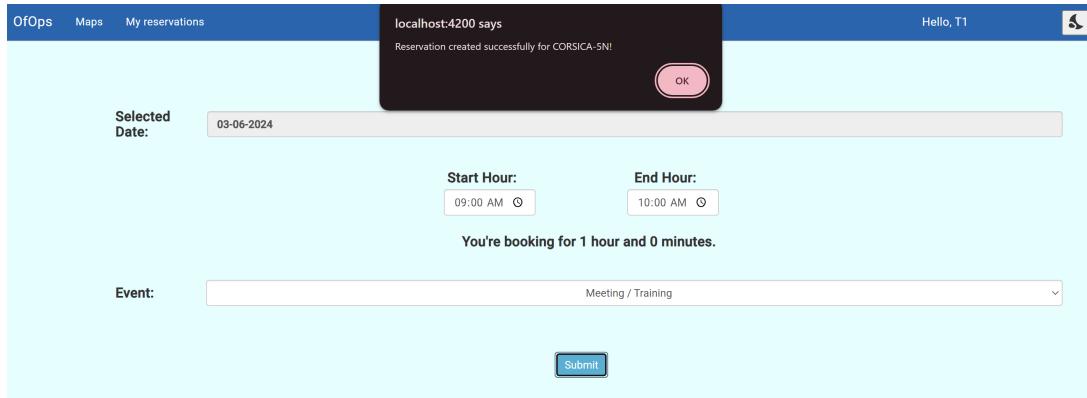


Figura 27: Rezervarea cu succes a sălii propuse

Astfel, user-ul va fi redirectionat către pagina de **My Reservations**.

### 3.5.5 Pagina My Reservations

Este pagina în care utilizatorul își poate vedea rezervările.

Date	From	To	Place Reserved	Event	Actions
03-06-2024	09:30	10:30	SICILY-6S	Meeting / Training	<a href="#">See on map</a> <a href="#">Delete</a>
03-06-2024	09:41	10:41	CAPRI-5N	Daily booking	<a href="#">See on map</a> <a href="#">Delete</a>
03-06-2024	09:00	18:42	D1-6S	Daily booking	<a href="#">See on map</a> <a href="#">Delete</a>
21-01-2024	12:00	12:00	D1-6S	Daily booking	<a href="#">See on map</a> <a href="#">Delete</a>

Figura 28: Pagina My reservations

Aceasta are, de asemenea, introdusă și funcționalitatea de paginare, astfel încât gestionarea rezervărilor să fie o experiență plăcută pentru user, fără a-l încărca de informație.

Culoarea roșie a ultimei rezervări marchează faptul că aceasta este expirată, iar celelalte reprezintă viitoarele rezervări. Se poate observa că sunt prezente detaliile rezervărilor (data, orele de început și sfârșit, evenimentul pentru care sunt rezervate) și două butoane: **See on Map** și **Delete**.

**Delete** va șterge rezervarea, după dorința utilizatorului.

OfOps	Departments	Teams	Users	Maps	Events	My reservations	Hello, T0	Logout																												
<b>My reservations</b>																																				
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>Date</th><th>From</th><th>To</th><th>Place Reserved</th><th>Event</th><th colspan="2">Actions</th></tr> </thead> <tbody> <tr> <td>03-06-2024</td><td>09:30</td><td>10:30</td><td>SICILY-6S</td><td>Meeting / Training</td><td><a href="#">See on map</a></td><td><a href="#">Delete</a></td></tr> <tr> <td>03-06-2024</td><td>09:41</td><td>10:41</td><td>CAPRI-5N</td><td>Daily booking</td><td><a href="#">See on map</a></td><td><a href="#">Delete</a></td></tr> <tr> <td>03-06-2024</td><td>09:00</td><td>18:42</td><td>D1-6S</td><td>Daily booking</td><td><a href="#">See on map</a></td><td><a href="#">Delete</a></td></tr> </tbody> </table>									Date	From	To	Place Reserved	Event	Actions		03-06-2024	09:30	10:30	SICILY-6S	Meeting / Training	<a href="#">See on map</a>	<a href="#">Delete</a>	03-06-2024	09:41	10:41	CAPRI-5N	Daily booking	<a href="#">See on map</a>	<a href="#">Delete</a>	03-06-2024	09:00	18:42	D1-6S	Daily booking	<a href="#">See on map</a>	<a href="#">Delete</a>
Date	From	To	Place Reserved	Event	Actions																															
03-06-2024	09:30	10:30	SICILY-6S	Meeting / Training	<a href="#">See on map</a>	<a href="#">Delete</a>																														
03-06-2024	09:41	10:41	CAPRI-5N	Daily booking	<a href="#">See on map</a>	<a href="#">Delete</a>																														
03-06-2024	09:00	18:42	D1-6S	Daily booking	<a href="#">See on map</a>	<a href="#">Delete</a>																														
« Previous <b>1</b> Next »																																				

Figura 29: Ultima rezervare ștearsă

Butonul **See on map** vine în ajutorul user-ului pentru a-i aduce aminte de locul rezervat. Acesta va fi redirecționat către harta și locul rezervat care va fi distins prin culoarea galbenă.

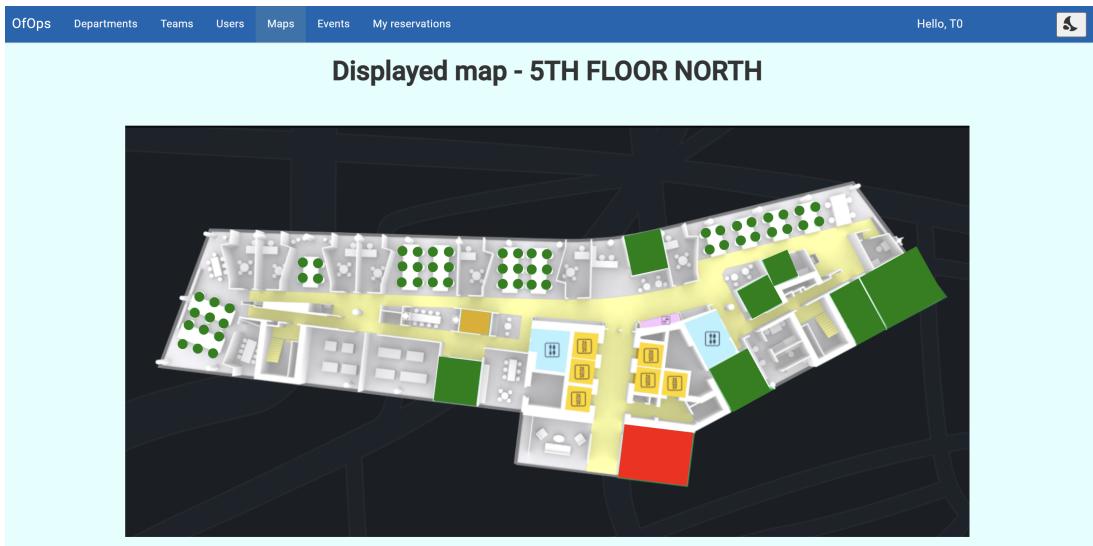


Figura 30: Rezervarea utilizatorului curent

# Capitolul 4

## Concluzii

OfOps oferă o soluție eficientă pentru gestionarea organizată a spațiilor comune ale unei firme și îmbunătățirea experienței angajaților la locul de muncă. Prin felul în care este construită aplicația, procesul de verificare și rezervare al locurilor disponibile contribuie la reducerea timpului și a eforturilor administrative, permitând oamenilor să se concentreze pe activitățile lor esențiale de zi cu zi. O asemenea soluție este un instrument valoros și modern pentru managementul resurselor comune.

O viitoare versiune a aplicației poate include, de asemenea, o nouă ară de aplicabilitate a OfOps în mediul universitar. Implementarea acestei soluții în universități ar putea avea multiple beneficii atât pentru studenți, cât și pentru personalul academic. Aceasta ar putea ajuta la planificarea eficientă a cursurilor, seminarilor și laboratoarelor, maximizând folosirea resurselor comune din cadrul universității.

În concluzie, aplicația OfOps poate fi implementată la nivelul unei companii private cu perspectiva ca versiunile următoare să permită utilizarea și în instituții de învățământ, inclusiv cu posibilitatea de dezvoltare ulterioară pe partea de aplicabilitate (notificarea unui utilizator cu 30 de minute înainte de a desfășura o activitate într-un spațiu rezervat).

# Bibliografie

- [1] Stuart Anderson, *Integration Testing*, 2011, URL: [https://www.inf.ed.ac.uk/teaching/courses/st/2011-12/Resource-folder/10\\_integration.pdf](https://www.inf.ed.ac.uk/teaching/courses/st/2011-12/Resource-folder/10_integration.pdf) (accesat în 1.6.2024).
- [2] *HTML: HyperText Markup Language*, 2024, URL: <https://developer.mozilla.org/en-US/docs/Web/HTML> (accesat în 31.5.2024).
- [3] *Introduction to the Angular docs*, 2024, URL: <https://angular.io/docs> (accesat în 31.5.2024).
- [4] *Java (programming language)*, 2024, URL: [https://en.wikipedia.org/wiki/Java\\_\(programming\\_language\)#cite\\_note-16](https://en.wikipedia.org/wiki/Java_(programming_language)#cite_note-16) (accesat în 31.5.2024).
- [5] Buse Kanal, *The Significance of Dark Mode in App Design and User Retention*, 2023, URL: <https://www.shyftup.com/blog/the-significance-of-dark-mode-in-app-design-and-user-retention/> (accesat în 1.6.2024).
- [6] Vladimir Khorikov, *Unit Testing - Principles, Practices, and Patterns*, 2020, URL: <https://books-library.net/files/books-library.net-07192142Kn9I3.pdf> (accesat în 1.6.2024).
- [7] *Maven - Introduction*, 2024, URL: <https://maven.apache.org/what-is-maven.html> (accesat în 31.5.2024).
- [8] Mr.K.J.Sharma, *WEB APPLICATION DEVELOPMENT*, 2015, URL: [https://baou.edu.in/assets/pdf/PGDCA-202\\_slm.pdf](https://baou.edu.in/assets/pdf/PGDCA-202_slm.pdf) (accesat în 31.5.2024).

- [9] *MySQL Workbench tutorial: Complete guide to the RDBMS tool*, 2024, URL: <https://www.educative.io/blog/mysql-workbench-tutorial#what> (accesat în 31.5.2024).
- [10] Sebastián E. Peyrott, *JWT HANDBOOK*, 2018, URL: [https://assets.ctfassets.net/2ntc334xpx65/o5J4X472PQUI4ai6cAcqg/13a2611de03b2c8edbd09c3ca14ae86b/jwt-handbook-v0\\_14\\_1.pdf](https://assets.ctfassets.net/2ntc334xpx65/o5J4X472PQUI4ai6cAcqg/13a2611de03b2c8edbd09c3ca14ae86b/jwt-handbook-v0_14_1.pdf) (accesat în 1.6.2024).
- [11] Herbert Schildt, *JAVA The Complete Reference Ninth Edition*, 2014, URL: <https://www.sietk.org/downloads/javabook.pdf> (accesat în 31.5.2024).
- [12] *Spring Framework Documentation*, 2022, URL: <https://docs.spring.io/spring-framework/docs/6.0.0/reference/pdf/spring-framework.pdf> (accesat în 31.5.2024).
- [13] *What is a Web Application?*, 2021, URL: <https://www.javatpoint.com/web-application> (accesat în 31.5.2024).
- [14] *What is MySQL?*, 2024, URL: <https://www.oracle.com/mysql/what-is-mysql/> (accesat în 31.5.2024).
- [15] *What is TypeScript, and why is it used in Angular?*, 2024, URL: <https://www.geeksforgeeks.org/what-is-typescript-and-why-is-it-used-in-angular/> (accesat în 31.5.2024).