



EÖTVÖS LORÁND TUDOMÁNYEGYETEM
INFORMATIKAI KAR
PROGRAMOZÁSELMÉLET ÉS SZOFTVERTECHNOLÓGIAI
TANSZÉK

**Webalapú Hotelmenedzsment és Foglalási
Rendszer**

Témavezető:
Brányi László
Mesteroktató

Szerző:
Csapó Benedek István
programtervező informatikus BSc

Budapest, 2025

EÖTVÖS LORÁND TUDOMÁNYEGYETEM
INFORMATIKAI KAR

SZAKDOLGOZAT TÉMABEJELENTŐ

Figyelem! Nyári záróvizsga esetén a módosítás határideje február 1., őszi záróvizsga esetén augusztus 31.

Hallgató adatai:

Név: Csapó Benedek István

Neptun kód: LFUYG3

Képzési adatok:

Szak: programtervező informatikus, alapképzés (BA/BSc/BProf)

Tagozat : Nappali

Belső témavezetővel rendelkezem

Témavezető neve: Brányi László

munkahelyének neve, tanszéke: ELTE IK, Információs rendszerek Tanszék

munkahelyének címe: 1117, Budapest, Pázmány Péter sétány 1/C.

beosztás és iskolai végzettsége: Mesteroktató

A szakdolgozat címe: Webalapú Hotelmenedzsment és Foglalási Rendsze

A szakdolgozat témája:

(A témavezetővel konzultálva adja meg 1 / 2 - 1 oldal terjedelemben szakdolgozat témajának leírását)

Ez a szakdolgozat egy átfogó, webalapú szállodai menedzsment rendszer tervezését és fejlesztését tűzi ki célul, amely a szálloda munkatársai számára biztosít platformot a szálloda napi működésének digitális támogatására.

A rendszer célja, hogy gyors, felhasználóbarát platformot nyújtson a szálloda belső folyamatainak hatékony digitális kezelésére.

A téma iránti érdeklődésem személyes háttere, hogy családomban van, aki hotelmenedzsmenttel foglalkozik, így úgy vélem, valamennyire egyedi rálátásom van a szállodai rendszerek működésére és igényeire.

A fejlesztendő rendszer tervezése és funkciói:

Backend oldali funkciók:

Autentikáció és autorizáció – különböző hozzáférési szintek létrehozása a szálloda alkalmazottai számára (recepció, menedzser, housekeeping, stb.).

Foglalási adatok kezelése, Szobák elérhetőségének és állapotának kezelése.

Értesítési rendszerek.

Statisztikai modul.

Adatbázis-kezelés – a foglalások és szállodai adatok tárolása és optimális elérése.

Frontend oldali funkciók:

Frontdesk opciók – vendégek kezelése, check-in/check-out folyamatok, foglalások adminisztrálása és szobák listázása szűrési lehetőségekkel.

Housekeeping funkciók – szobák állapotának kezelése, takarítási feladatok tütemezése és nyomon követése.

Szálloda adatok kezelése – alapvető szállodai információk, szobatípusok, árak és szolgáltatások módosítása.

Statisztikai modul – jelentések és statisztikák megtekintése, foglaltsági adatok elemzése, nyomtatási funkciók különböző dokumentumokhoz.

Számlázási modul.

Reszponzív design.

Budapest, 2025. 08. 24.

Tartalomjegyzék

1. Bevezetés	4
1.1. Feladat bevezetése	4
1.2. A dolgozat szerkezete	5
2. Felhasználói dokumentáció	6
2.1. Bemutatkozás	6
2.2. Célcsoportok	6
2.2.1. Kiknek készült?	6
2.2.2. Mire?	6
2.3. Követelmények	7
2.3.1. Minimális követelmények	7
2.3.2. Optimális követelmények	7
2.4. Telepítés és első indítás	7
2.4.1. A rendszer elérése	7
2.4.2. Első bejelentkezés	8
2.4.3. Biztonsági beállítások első használat után	9
2.4.4. Hibaelhárítás	10
2.4.5. Technikai támogatás	11
2.5. Felületek és műveletek	11
2.5.1. Vezérlőpult	11
2.5.2. Alapadatok kezelése	12
2.5.3. Recepció és foglalások kezelése	25
2.5.4. Számla műveletek kezelése	31
2.5.5. Takarítások kezelése	36
2.6. Hibaüzenetek, figyelmeztetések és visszajelzések	38
2.6.1. Űrlap validációs hibák	39
2.6.2. Műveletek végrehajtási üzenetek	39

3. Fejlesztői dokumentáció	41
3.1. Bevezetés	41
3.2. Rendszer architektúra	41
3.3. Technológiai stack	43
3.3.1. Frontend technológiák	43
3.3.2. Backend technológiák	44
3.3.3. Adatbázis technológia	44
3.3.4. Fejlesztői eszközök	45
3.3.5. Technológiai döntések indoklása	45
3.4. Adatbázis felépítése	45
3.4.1. Adatbázis konfiguráció	45
3.4.2. Entitás-kapcsolat diagram (ERD)	46
3.4.3. Entitások áttekintése	46
3.4.4. Részletes entitás leírások	48
3.5. Modul- és osztályszerkezet	53
3.5.1. Architektúra áttekintő diagram	53
3.5.2. Főbb osztályok/modulok	54
3.5.3. Osztályok közötti kapcsolatok példa	57
3.5.4. UI-tervezés és navigáció	59
3.5.5. Felhasználói felület tervei	59
3.5.6. Navigációs diagram	63
3.6. Kiemelndő kodrészletek	64
3.6.1. Közös algoritmusok	64
3.6.2. Nem triviális megoldások	74
3.7. Telepítési utmutató	89
3.7.1. Rendszerkövetelmények	89
3.7.2. Docker telepítése	89
3.7.3. Projekt letöltése és indítása	89
3.7.4. Docker konfiguráció	90
3.7.5. Gyakori hibák és megoldásuk	94
3.8. Tesztek	95
3.8.1. Tesztelési módszertan	95
3.8.2. Tesztesetek	96

4. Teszt jegyzőkönyv	101
4.0.1. Bejelentkezési képernyő	101
4.0.2. Regisztrációs képernyő	102
4.0.3. Foglalások oldal	103
4.0.4. Vendégek oldal	111
4.0.5. Számlák oldal	114
4.0.6. Takarítás oldal	117
5. Összegzés	121
6. Függelék	122
6.1. Szerepkörök	122
Irodalomjegyzék	123
Ábrajegyzék	123
Táblázatjegyzék	125
Forráskódjegyzék	126

1. fejezet

Bevezetés

1.1. Feladat bevezetése

A legtöbb iparhoz hasonlóan az elmúlt évtizedben a szállodaipar is egy jelentős digitális átalakuláson ment keresztül. Manapság egy szálloda működése szinte elképzelhetetlen digitalizált menedzsment rendszerek nélkül. A foglalás kezelés, vendégelek adatok tárolása, szobakiosztás optimalizálása és sok egyéb más kulcs fontosságú területen nyújtanak hatékony megoldást ezen programok.

Azonban a piacon elérhető megoldások általában túlzottan bonyolultak vagy drágák kisebb, közepes méretű szállodák speciális igényeinek. Így előfordul, hogy ezen szállodák még mindig manuálisan, táblázatkezelőkkel vagy rosszabb esetben papíron vezetik minden nap munkájukat.

Szakdolgozatom célja egy modern, egyszerű, de hatékony webalapú hotelmenedzsment és foglalási rendszer készítése. A weblap kifejezetten kisebb szállodák igényeit tartja szem előtt. Webalapja miatt minden internettel rendelkező eszközről elérhető helytől függetlenül. Felületet biztosít szobák, foglalások, vendégek és egyéb hotelmenedzsment területek kezelését, miközben a hotel adatait biztonságosan tárolja.

A téma választást családi kapcsolat befolyásolta: családom tagja több mint 20 éve van az iparban és rajta keresztül tapasztaltam, hogy mennyire elavult, túlkomplikált szoftvert használnak több helyen is a hotel menedzs selésére. Így célként kitűztem egy egyszerű, modern rendszer kiépítését.

1.2. A dolgozat szerkezete

1. Felhasználói dokumentáció: Ismertetem hogyan kell használni és telepíteni az alkalmazást.
2. Fejlesztői dokumentáció: A rendszer belső működését beleértve a tervezési döntéseket, adatstruktúrákat és programozási megoldásokat mutatom be.
3. Tesztesetek tárháza.
4. Összefoglalás

2. fejezet

Felhasználói dokumentáció

2.1. Bemutatkozás

Webalapú szállodamenedzsment rendszert szeretnék bemutatni mely felületet biztosít kisebb vagy közepes méretű hotelek minden nap feladatainak egyszerűsítésére, megoldására és nyomon követésére.

Modern, egyszerűbb alternatívát kínál a drágább, bonyolultabb vagy elavultabb rendszerekkel szemben.

2.2. Célközönség

2.2.1. Kiknek készült?

Elsősorban a szálloda recepcióainak készült, de mellettük a menedzserek, vezetők és a szállodai gondnokság dolgozói (housekeeping) is hasznát vehetik minden nap munkájuk során.

2.2.2. Mire?

- Foglalások kezelésére, vizualizálására.
- Szálloda jelenlegi statisztikái megtekintésére.
- Vendég adatok tárolására, szerkesztésére.
- Szálloda szervizeinek menedzselésére.
- Különféle ÁFA-ráták nyilvántartására
- Számlák megtekintésére és műveleteire

- Housekeeping feladatok kezelésére
- Felhasználók, jogosultságok alakítására
- Szobák adatainak módosítására
- Különböző alap adatok módosítására (Szoba, Ágy típusok ...)

2.3. Követelmények

2.3.1. Minimális követelmények

Internetkapcsolattal rendelkező eszköz (PC, laptop, tablet, telefon). Modern webböngésző (Chrome 90+, Firefox 88+, Safari 14+, Edge 90+). 1280x720 képernyő-felbontás.

2.3.2. Optimális követelmények

Asztali számítógép vagy laptop. 4GB+ RAM. Stabil, széles sávú internetkapcsolat (min. 5 Mbps). 1920x1080 vagy nagyobb felbontás. Ajánlott böngésző: Chrome.

2.4. Telepítés és első indítás

2.4.1. A rendszer elérése

A rendszer egy webalapú alkalmazás, amely böngészőből érhető el. **Nem szükséges semmilyen szoftver telepítése az Ön számítógépére** a rendszer használatához

Helyi hálózati elérés

Amennyiben a rendszert a szálloda helyi hálózatán futtatják, az alapértelmezett elérési cím:

http://localhost:3000

Fontos: Az elérési cím változhat attól függően, hogy a rendszer gázda hogyan konfigurálta a szervert. A pontos címet a rendszer üzembe helyezésekor kapja meg az IT osztálytól vagy a rendszer gázdától.

Online demo verzió

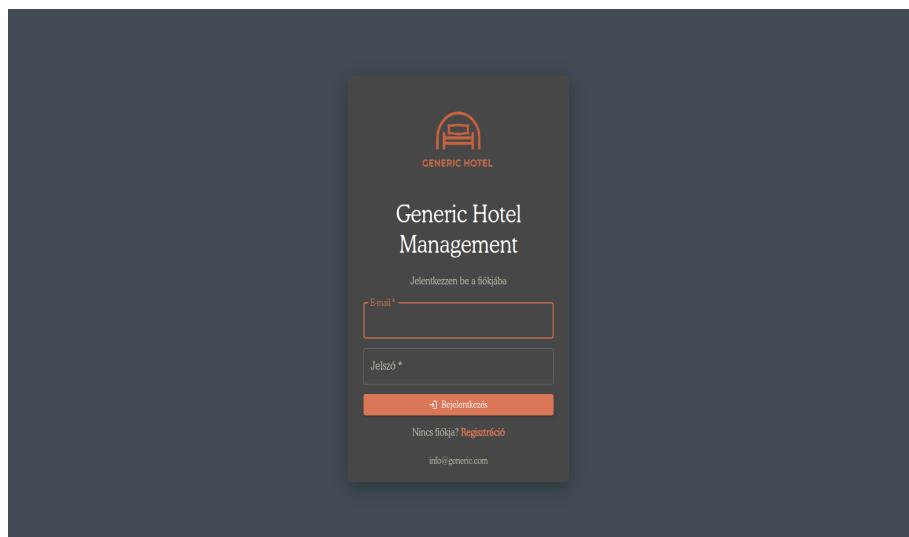
A rendszer egy működő demo verziója nyilvánosan elérhető az alábbi címen:

<https://hotelpmsdemo.up.railway.app>

A demo verzió ugyanazokkal a funkciókkal rendelkezik, mint az éles rendszer, és tesztelési célokra használható. **Figyelem:** A demo verzióban tárolt adatok nem valósak, rendszeresen törlésre kerülnek és alkalmazás ingyenes infrastruktúrát használ, ami lassabb működést eredményezhet. Türelmét köszönjük!

2.4.2. Első bejelentkezés

A rendszer megnyitásakor a bejelentkezési képernyő fogadja Önt:



2.1. ábra. Alapértelmezetten megjelenő bejelentkezési képernyő

Alapértelmezett belépési adatok

A frissen telepített rendszer az alábbi alapértelmezett bejelentkezési adatokkal rendelkezik tesztelési célokra:

E-mail cím	Jelszó
admin@hotel.com	admin123

2.1. táblázat. Alapértelmezett adminisztrátori fiók

Biztonsági figyelmeztetés: Ez a fiók csak kezdeti beállításhoz és teszteléshez használható!

Kezdeti adatok

A rendszer első indításakor alapértelmezett demo adatokkal rendelkezik, amelyek megkönnyítik a rendszer kipróbálását:

- Minta szobák (pl. Szoba 101, 102, 103)
- Szoba típusok
- Ágy típusok
- ÁFA kulcsok
- Egyéb alapadatok

Ezeket az adatokat **a rendszer használata előtt törölnie vagy módosítania kell** a szálloda tényleges adataival.

2.4.3. Biztonsági beállítások első használat után

FONTOS! Az első bejelentkezés után az alábbi biztonsági lépéseket **kötelezően** végre kell hajtani (Kivéve Demo weblap használata során):

1. Új adminisztrátori felhasználó létrehozása

Navigáljon a Felhasználók menüpontba, és hozzon létre egy új adminisztrátori jogosultságú felhasználót a saját e-mail címével és erős jelszóval.

The screenshot shows two windows side-by-side. On the left is a dark-themed sidebar menu with various icons and labels. The 'Felhasználók' item is highlighted with a brown background. On the right is a modal window titled 'Felhasználó létrehozása'. It contains five input fields: 'Keresztnév', 'Vezetéknév', 'Telefon', 'E-mail', and 'Jelszó'. Below these is a dropdown menu labeled 'Szerepkörök'. At the bottom are two buttons: 'Cancel' and 'Save'.

(a) Felhasználók menü kiválasztva

(b) Felhasználót létrehozó ablak

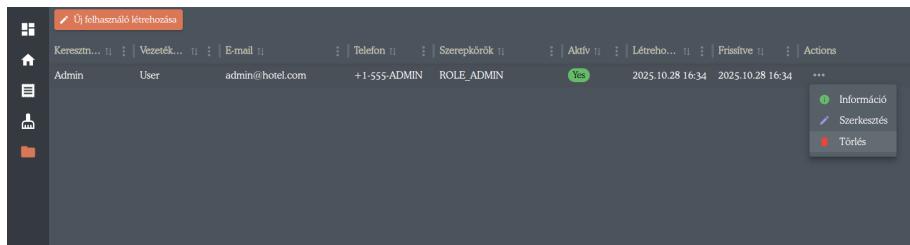
2.2. ábra. Új felhasználó létrehozását dokumentáló képek

2. Jelentkezzen ki, majd jelentkezzen be az új fiókkal

Ellenőrizze, hogy az új fiók megfelelően működik és rendelkezik adminisztrátori jogosultságokkal.

3. Törölje az alapértelmezett admin@hotel.com fiókot

A Felhasználók menüben válassza ki az alapértelmezett admin fiókot és törölje véglegesen.



2.3. ábra. Alapértelmezett admin fiók törlése

4. Törölje vagy módosítsa a demo adatokat

Navigáljon végig a Szobák, Vendégek és Beállítások menüpontokban, és cserélje le a demo adatokat a szálloda valós adataival.

2.4.4. Hibaelhárítás

Nem érem el a rendszert

Ha a megadott címen nem érhető el a rendszer:

- Ellenőrizze, hogy helyesen írta-e be a címet (<http://> vagy <https://>)
- Próbáljon meg más böngészőt használni
- Ellenőrizze, hogy a számítógépe kapcsolódik-e a hálózathoz
- Ellenőrizze, hogy a szerver fut-e (kérdezze meg a rendszergazdát)

Nem tudok bejelentkezni

Ha a bejelentkezési adatai nem működnek:

- Ellenőrizze, hogy helyesen írta-e be az e-mail címet és jelszót
- Figyeljen a kis- és nagybetűk különbségére
- Próbálja meg az alapértelmezett admin fiókot (csak új telepítésnél)
- Ha elfelejtette jelszavát, forduljon a rendszergazdához

2.4.5. Technikai támogatás

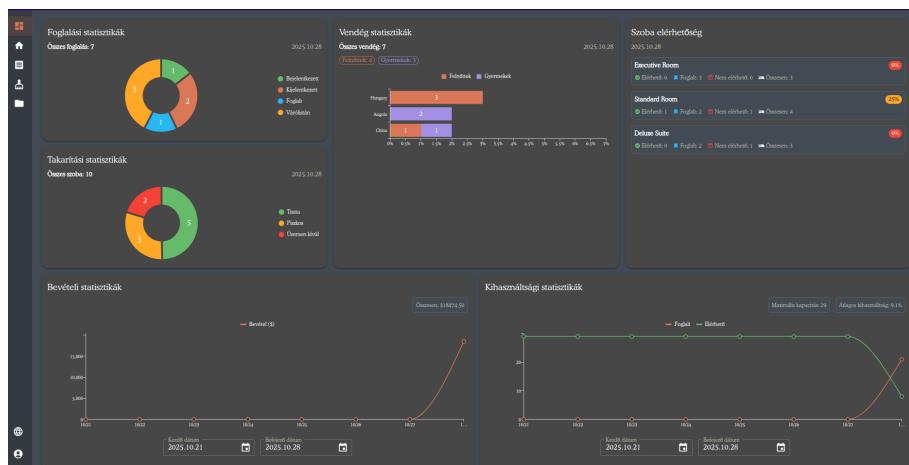
Ha a fenti megoldások nem segítenek, vagy bármilyen egyéb technikai problémája merül fel, kérjük, forduljon a rendszergazdához vagy az IT osztályhoz.

A rendszer telepítésével, konfigurálásával és szerver oldali karbantartásával kapcsolatos információk a **3.7. részben** találhatók (Elsősorban informatikai szakemberek részére).

2.5. Felületek és műveletek

2.5.1. Vezérlőpult

Bejelentkezés után a vezérlőpult fogad, amely a szálloda aktuális állapotáról nyújt gyors áttekintést. Itt megtekinthető a mai foglalásokat, vendégstatisztikákat és a bevételi adatokat.



2.4. ábra. Vezérlőpult megjelenése

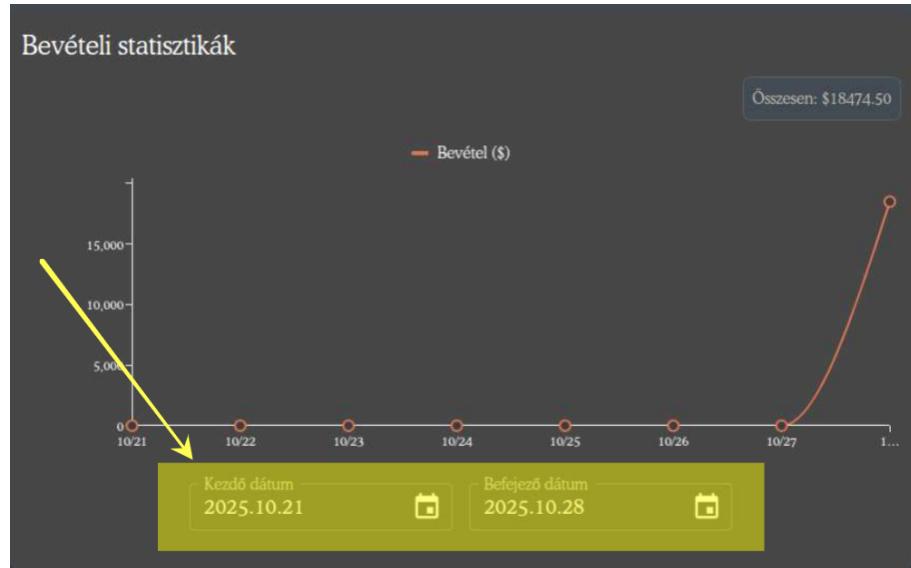
A következő statisztikák jelennek meg balról jobbra:

- Foglalási statisztikák:** Mai érkezések, távozások és aktív foglalások száma
- Vendégstatisztikák:** Jelenleg tartózkodó vendégek száma
- Elérhető szobatípusok:** Szabad szobák típusonként
- Takarítási statisztikák:** Takarítandó és kész szobák
- Bevételi gráf:** Napi bevételek alakulása
- Kihasználtsági gráf:** Szobaihasználtság

Időszak beállítása a gráfoknál

A bevételi és kihasználtsági gráfok esetében beállíthatom a megjeleníteni kívánt időszakot. Alapértelmezetten az elmúlt egy hétre szűr a rendszer.

Fontos: Minél hosszabb időszakot választok, annál több időt vesz igénybe a gráf betöltése.



2.5. ábra. Bevételi gráf időszak beállítása

2.5.2. Alapadatok kezelése

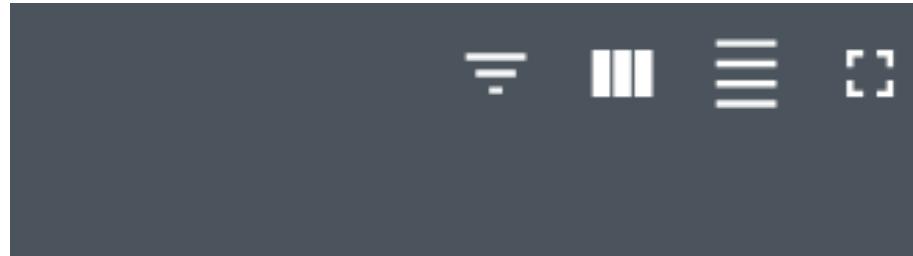
Ezeken a felületeken tárolódnak a szálloda alapvető adatai szűrhető táblákban. A táblázatok minden sornál megtekinthető a létrehozás és az utolsó módosítás dátumát. A táblázatokat minden felhasználó megtekintheti, azonban az adatok létrehozása és módosítása Admin vagy Data maintaner jogosultsághoz kötött (lásd: 6.1. függelék).

Megjegyzés: minden művelet után a rendszer sikeres mentésről vagy hibaüzenettel tájékoztat.

Szűrők és táblaműveletek A táblázatok jobb felső sarkában találhatóak a következő tábla műveletek (lásd: 2.6. ábra, balról jobbra):

1. Oszlopszűrők megjelenítése
2. Oszlopok megjelenítése vagy elrejtése
3. Táblázat sűrűségének beállítása
4. Teljes képernyős mód

Megjegyzés: Alapértelmezetten léteznek elrejtett oszlopok!



2.6. ábra. Táblaműveletek



2.7. ábra. Oszlopszűrés és rendezés felület

Alapadatok közös kezelési műveletek

A következő alapadat-táblák azonos módon kezelhetők:

- Épületek
- Szoba szolgáltatások
- Vendég címkek
- Ágytípusok
- Szerepkörök
- Felhasználók
- Szoba típusok
- Szobák

Mindegyik felületen az alábbi három alapművelet érhető el alapból a megfelelő jogosultsággal:

Új elem hozzáadása:

1. Megnyomom az **Új [elem] feltöltése** gombot
2. Megnyílik a létrehozó űrlap
3. Kitölöm a kötelező mezőket
4. Elmentem

Épület feltöltése

Aktív

Cancel Save

Név	Cím	Szűrés: Név ...	Szűrés:
Main Building	123 Hot		
Annex Building	124 Hot		
Suite Building	125 Hot		

(a) Épület hozzáadása gomb

(b) Épület létrehozó űrlap

2.8. ábra. Példa: Új épület hozzáadásának lépései

Meglévő elem módosítása:

1. A módosítandó elem sorának végén megnyomom a ... gombot
2. Kiválasztom a **Szerkesztés** menüpontot
3. Módosítom a kívánt mezőket
4. Elmentem



(a) Épület műveletek menü

Épület neve	Annex Building
Cím	124 Hotel Street
Város	Hotel City
Irányítószám	12345
Ország	Magyarország
Telefonszám	+1-555-0002
E-mail	annex@hotel.com
Leírás	Secondary hotel building
<input checked="" type="checkbox"/> Aktív	
<input type="button" value="Cancel"/> <input type="button" value="Save"/>	

(b) Épület szerkesztő űrlap

2.9. ábra. Példa: Épület szerkesztésének lépései

Elem törlése:

1. A törölendő elem sorának végén megnyomom a ... gombot
2. Kiválasztom a **Törlés** menüpontot
3. Megerősíttem a törlést

E-mail ↑	⋮ Aktív ↑	⋮ Műveletek
Szűrés: E-mail alapján	Szűrés: Aktív alapjára ▾	
main@hotel.com	Igen	...
annex@hotel.com	Igen	Szerkesztés
suites@hotel.com	Igen	Törlés

2.10. ábra. Példa: Épület törlése gomb

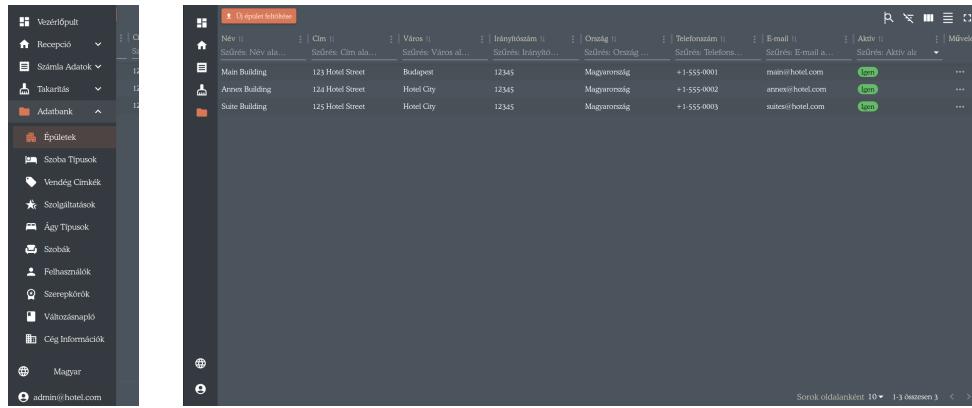
Fontos: Az elemek törlése csak akkor lehetséges, ha nincsenek hozzájuk kapcsolódó rekordok más táblákban. De erről majd pontosabban az egyes felületek ismertetésénél kitérek.

Épületek felület

Mikor használom? Az épületek kezelése oldalon állíthatom be, hogy a szálloda mely épületeiben találhatók szobák. Ez akkor hasznos, ha a szálloda több épületből áll (pl. főépület, melléképület).

Specifikus mezők:

- Név
- Cím
- Város
- Irányítószám
- Ország
- Telefonszám
- Email
- Aktív-e



The screenshot shows the 'Épületek' (Buildings) section of the system. On the left is a sidebar menu with categories like 'Számla Adatok', 'Takarítás', 'Adatbank', and 'Épületek'. Under 'Épületek', there are sub-options: Szoba Tipusok, Vendégl Címek, Szolgáltatások, Ágy Tipusok, Szobák, Felhasználók, Szerepkörök, Változásnapló, Cég Információk, and Magyar. At the bottom of the sidebar is the email 'admin@hotel.com'. The main panel displays a table titled 'Új épület felülete' (New Building Interface) with columns: Név (Name), Cím (Address), Város (City), Irányítószám (Postcode), Ország (Country), Telefonszám (Phone Number), E-mail (Email), Aktív (Active), and Műveletek (Actions). The table contains three rows: Main Building (123 Hotel Street, Budapest, 12345, Magyarország, +1 555-0001, main@hotel.com, Aktív, Actions), Annex Building (124 Hotel Street, Hotel City, 12345, Magyarország, +1 555-0002, annex@hotel.com, Aktív, Actions), and Suite Building (125 Hotel Street, Hotel City, 12345, Magyarország, +1 555-0003, suite@hotel.com, Aktív, Actions). A status bar at the bottom right indicates 'Sorok oldalanként 10 • 1-3 összesen 3 < >'.

(a) Épület elérése a menüből

(b) Épületek táblázat

2.11. ábra. Épületek felület

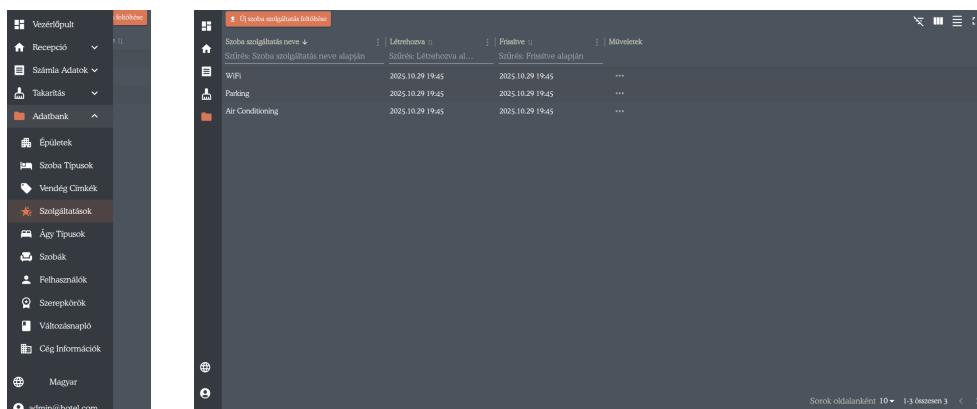
Törlési megszorítás: Az épületet csak akkor törölhetem, ha nincsenek hozzá tartozó szobák. Ellenkező esetben a rendszer hibaüzenetet jelenít meg.

Szoba szolgáltatások felület

Mikor használom? Ezen az oldalon állíthatom be, hogy a szálloda szobatípusai milyen szolgáltatásokkal rendelkezhetnek (pl. WiFi, léggondicionáló, minibar).

Specifikus mezők:

- Szoba szolgáltatás neve



The screenshot shows the 'Szoba szolgáltatások felülete' (Room Services Interface) section. On the left is a sidebar menu with categories like 'Számla Adatok', 'Takarítás', 'Adatbank', and 'Szolgáltatások'. Under 'Szolgáltatások', there are sub-options: WiFi, Parking, Air Conditioning, and others. At the bottom of the sidebar is the email 'admin@hotel.com'. The main panel displays a table titled 'Új szoba szolgáltatás felülete' (New Room Service Interface) with columns: Szoba szolgáltatás neve (Room Service Name), Létrehozva (Created), Frissítve (Last Update), and Műveletek (Actions). The table contains four rows: WiFi (2025.10.29 19:45, 2025.10.29 19:45, ...), Parking (2025.10.29 19:45, 2025.10.29 19:45, ...), Air Conditioning (2025.10.29 19:45, 2025.10.29 19:45, ...), and another row that is partially visible. A status bar at the bottom right indicates 'Sorok oldalanként 10 • 1-3 összesen 3 < >'.

(a) Szoba szolgáltatások elérése a menüből

(b) Szoba szolgáltatások táblázat

2.12. ábra. Szoba szolgáltatások felület

Törlési megszorítás: A szolgáltatást csak akkor törölhetem, ha nincsenek olyan szobatípusok, amelyek használják az adott szolgáltatást. Ellenkező esetben a rendszer hibaüzenetet jelenít meg.

Vendég címkek felület

Mikor használom? Ezen az oldalon állíthatom be, hogy milyen címkekkel lehet hatok egyes vendégeknek (pl. törzsvendég, mozgáskorlátozott, VIP). Így a megfelelő címkézással könnyíthetem a recepciós munkatársaim munkáját.

Specifikus mezők:

- Címke neve
- Aktív-e

Címke neve	Aktív	Létrehozva	Primitív	Műveletek
VIP	Igen	2025.10.29 19:45	2025.10.29 19:45	...
Frequent Guest	Igen	2025.10.29 19:45	2025.10.29 19:45	...
Corporate	Igen	2025.10.29 19:45	2025.10.29 19:45	...

(a) Vendég címkek elérése a menüből

(b) Vendég címkek táblázat

2.13. ábra. Vendég címkek felület

Törlési megszorítás: A címket csak akkor törölhetem, ha nincs egyetlen vendéghöz sem hozzárendelve.

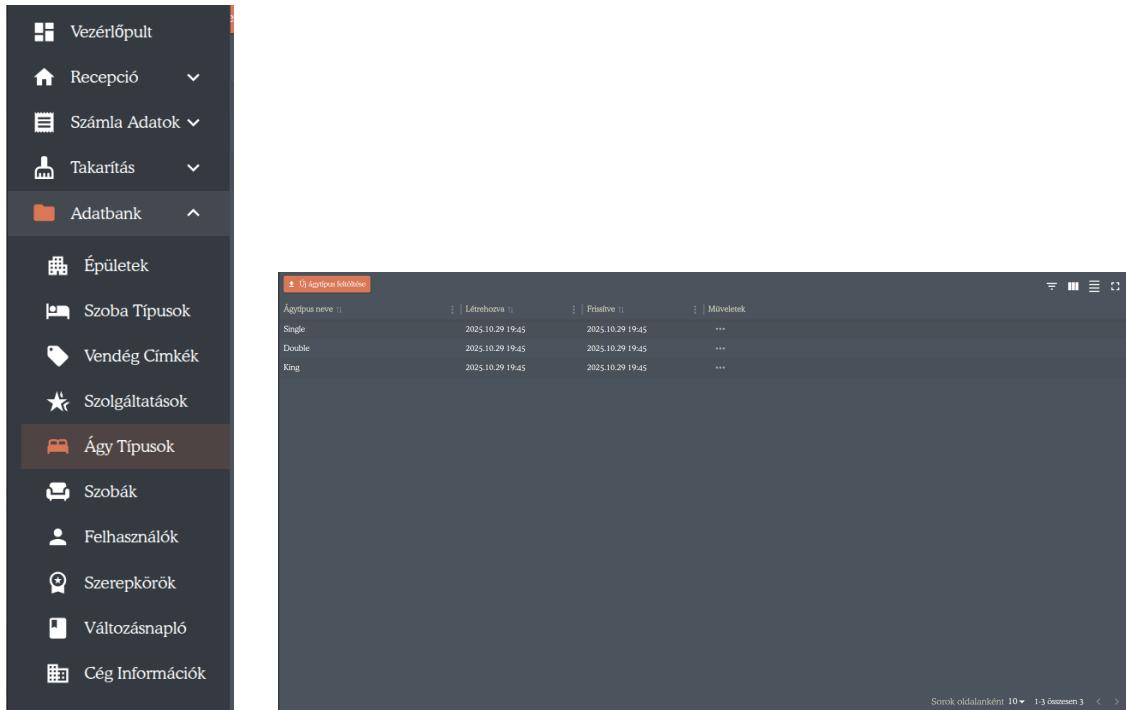
Ágytípusok felület

Mikor használom? Ezen az oldalon állíthatom be, hogy milyen ágytípusokat használunk a rendszerben (pl. egyszemélyes, franciaágy, kétszemélyes). Ezeket később

a szobatípusok felületen rendelhetem hozzá az egyes szobatípusokhoz megadott dárabszámmal.

Specifikus mezők:

- Ágytípus neve



Ágytípus neve	Létrehozva	Prissive	Műveletek
Single	2025.10.29 19:45	2025.10.29 19:45	...
Double	2025.10.29 19:45	2025.10.29 19:45	...
King	2025.10.29 19:45	2025.10.29 19:45	...

(a) Ágytípusok elérése a menüből

(b) Ágytípusok táblázat

2.14. ábra. Ágytípusok felület

Törlési megszorítás: Az ágytípust csak akkor törölhetem, ha nincs egyetlen szobatípushoz sem hozzárendelve.

Szerepkörök felület

Mikor használom? Ezen az oldalon állíthatom be, hogy milyen felhasználói szerepkörök léteznek a rendszerben (pl. recepciós, menedzser, housekeeping, adminisztrátor).

Specifikus mezők:

- Szerepkör neve
- Aktív-e

The screenshot shows a software interface with a dark-themed sidebar menu on the left and a main content area on the right.

Left Sidebar (Menü):

- Vezérlőpult
- Recepció
- Számla Adatok
- Takarítás
- Adatbank
- Épületek
- Szoba Típusok
- Vendég Címek
- Szolgáltatások
- Ágy Típusok
- Szobák
- Felhasználók
- Szerepkörök** (highlighted)
- Változásnapló
- Cég Információk

Main Content Area:

A table titled "Szerepkörök lista" (List of Roles) is displayed. The table has columns: Név (Name), Aktív (Active), Létrehozva (Created), Frissítve (Updated), and Művelek (Actions).

Név	Aktív	Létrehozva	Frissítve	Művelek
ROLE_ADMIN	Igen	2025.10.29 19:45	2025.10.29 19:45	...
Manager	Igen	2025.10.29 19:45	2025.10.29 19:45	...
Staff	Igen	2025.10.29 19:45	2025.10.29 19:45	...

At the bottom right of the table, there is a footer: "Sorok oldalankörül: 10 ▾ 1-3 összesen 3 < >"

(a) Szerepkörök elérése a menüből

(b) Szerepkörök táblázat

2.15. ábra. Szerepkörök felület

Törlési megszorítás: A szerepkört csak akkor törölhetem, ha nincs egyetlen felhasználóhoz sem hozzárendelve.

Megjegyzés: A szerepkörökhöz tartozó jogosultságokat a 6.1. függelékben részletezem.

Felhasználók felület

Mikor használom? Ezen az oldalon kezelhetem a rendszerben lévő felhasználók adatait, módosíthatom a szerepköreiket, valamint aktiválhatom vagy deaktiválhatom őket. A rendszerbe csak aktív felhasználó képes belépni.

Specifikus mezők:

- Keresztnév
- Vezetéknév
- Email
- Telefonszám
- Szerepkörök: egymás után vesszővel elválasztva
- Aktív-e

The image consists of two parts. Part (a) shows the left sidebar menu with 'Felhasználók' selected. Part (b) shows a table view of users with columns like Keresztnév, Vezetéknév, E-mail, Telefon, Szerepkörök, Aktív, Létrehozás-dátum, Frissítés-dátum, and Műveletek.

Keresztnév	Vezetéknév	E-mail	Telefon	Szerepkörök	Aktív	Létrehozás-dátum	Frissítés-dátum	Műveletek
Admin	User	admin@hotel.com	+1-555-ADMIN	ROLE_ADMIN	Tenni	2025.10.30 13:30	2025.10.30 13:30	...

(a) Felhasználók elérése a menüből

(b) Felhasználók táblázat

2.16. ábra. Felhasználók felület

Törlési megszorítás: A felhasználó törölhető, függetlenül attól, hogy vannak-e hozzá tartozó műveletek a rendszerben. Önmagamat nem tudom törölni.

Szobatípusok felület

Mikor használom? Ezen az oldalon kezelhetem a szobák típusait, amikor új szobatípushoz szeretnék létrehozni (pl. Standard kétágyas, Deluxe lakosztály), vagy módosítani szeretném egy meglévő típus paramétereit. Megadhatom, hogy egy adott típus hány és milyen ágyakkal, valamint milyen szoba szolgáltatásokkal rendelkezik. Az egy éjszakás alapárat is itt állíthatom be.

Specifikus mezők:

- Típus neve
- Ár
- Kapacitás
- Ágytípusok: ágyszám x ágytípus vesszővel elválasztva
- Kényelmi szolgáltatások: egymás után vesszővel elválasztva

- Aktív-e

(a) Szobatípusok elérése a menüből

(b) Szobatípusok táblázat

Szoba Színénél Típus neve alapján	Ár	Kapacitás	Ágytípusok	Kényelmi szolgáltatások	Műveletek
1 Standard Room	\$150.00	2	1x Double	Air Conditioning, WiFi	...
2 Deluxe Suite	\$350.00	4	1x King, 1x Single	Air Conditioning, WiFi, Parking	...
3 Executive Room	\$250.00	3	1x King	Air Conditioning, WiFi	...

2.17. ábra. Szobatípusok felület

Törlési megszorítás: A szobatípust csak akkor törölhetem, ha nincs egyetlen szoba sem az adott típushoz rendelve.

Szobák felület

Mikor használom? Ezen az oldalon kezelhetem a rendszerben létező szobákat, amikor új szobát szeretnék létrehozni vagy módosítani szeretném egy meglévő szoba paramétereit vagy tisztasági státuszát.

Specifikus mezők:

- Szobaszám
- Státusz: pillanatnyilag tiszta, piszkos vagy üzemen kívül van
- Emelet
- Épület
- Szobatípus
- Leírás

Sz...	Szobaszám	Státusz	Em...	Épület	Szobatípus	Lefrás	Műveletek
1	101	Tisztta	1	Main Building	Standard Room	Standard room with city view	...
2	102	Tisztta	1	Main Building	Standard Room	Standard room with garden view	...
3	201	Tisztta	2	Main Building	Standard Room	Standard room on second floor	...
4	202	Tisztta	2	Main Building	Standard Room	Standard room with balcony	...
5	A101	Tisztta	1	Annex Building	Executive Room	Executive room with workspace	...
6	A102	Tisztta	1	Annex Building	Executive Room	Executive room with lounge area	...
7	A201	Tisztta	2	Annex Building	Executive Room	Executive room premium view	...
8	S101	Tisztta	1	Suite Building	Deluxe Suite	Deluxe suite with living room	...
9	S201	Tisztta	2	Suite Building	Deluxe Suite	Deluxe suite with terrace	...
10	S301	Tisztta	3	Suite Building	Deluxe Suite	Penthouse deluxe suite	...

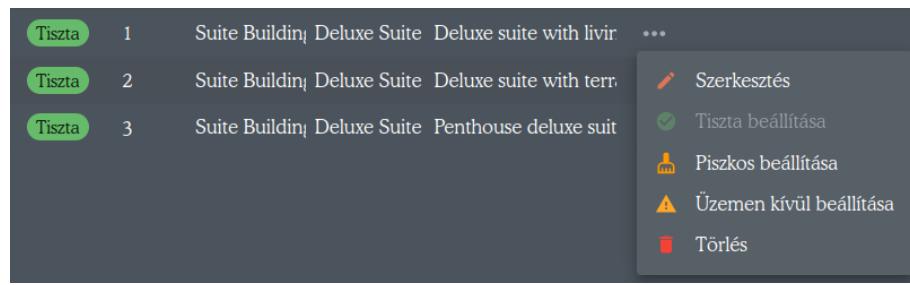
(a) Szobák elérése
a menüből

(b) Szobák táblázat

2.18. ábra. Szobák felület

Törlési megszorítás: A szobát csak akkor törölhetem, ha nincs hozzá kapcsolt foglalás. Ha van hozzákapcsolt takarítási kérvény, a szoba törlésével a kérvény is megszűnik.

Szoba specifikus műveletek: A sorok végén lévő ... gombra kattintva a megszokott funkciók mellett találunk 3 gyors gombot az adott szoba tisztasági státuszának beállítására (lásd 2.19. ábra).



2.19. ábra. Szoba műveletek

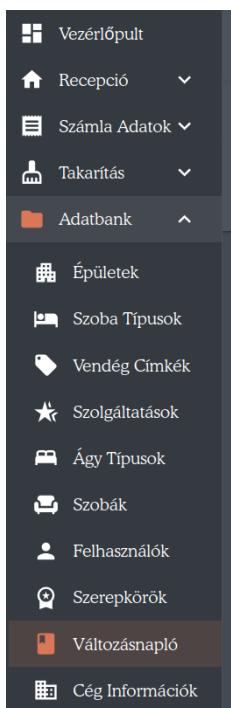
Változásnapló felület

Megjegyzés: A változásnapló kizárolag olvasható felület, nem rendelkezik létrehozás, módosítás vagy törlés funkciókkal.

Mikor használom? Itt tudom megnézni a rendszerben történt változtatások vagy fejlesztések listáját.

Specifikus mezők:

- Verzió
- Leírás



Azonosító	Verzió	Leírás	Létrehozva	Frisítve
3	1.2.0	Enhanced data initialization and improved service layer	2025.10.30 13:30	2025.10.30 13:30
2	1.1.0	Added PDF export functionality for invoices and reports	2025.10.30 13:30	2025.10.30 13:30
1	1.0.0	Initial release with basic hotel management features	2025.10.30 13:30	2025.10.30 13:30

(a) Változásnapló elérése a menüből

(b) Változásnapló táblázat

2.20. ábra. Változásnapló felület

Cég információ felület

Megjegyzés: Ez a felület lényegesen eltér a többi alapadatok felülettől, mivel nem táblázatos megjelenítésű. Nincs rajta törlési vagy új létrehozása funkció, kizárolag módosítani lehet a meglévő cég adatokat.

Mikor használom? Ha szeretném állítani a cégem információját.

Specifikus mezők:

- Cégnév
- Cím

- Telefonszám
- Vendégeknek szóló weboldal (ha van)
- Adószám
- Cégjegyzék szám

The screenshot shows the software's main interface. On the left is a vertical sidebar menu with the following items:

- Adatbank
- Épületek
- Szoba Típusok
- Vendég Címkek
- Szolgáltatások
- Ágy Típusok
- Szobák
- Felhasználók** (highlighted in grey)
- Szerepkörök
- Változásnapló
- Cég Információk

The main content area has a dark header with the title "Generic Hotel Management". Below the header are three sections:

- Alapvető Információk**

Cég neve	Generic Hotel Management
Cím	456 Business Avenue, Suite 100, Business City, BC 12345
- Kapcsolati Információk**

Telefon	+1-555-1323-1
E-mail	info@generic.com
- Jogi Információk**

Adószám	TAX-123456789
Cégjegyzékszám	REG-987654321

(a) Cég információ elérése a menüből

(b) Cég információ szerkesztő felület

2.21. ábra. Cég információ felület

2.5.3. Recepció és foglalások kezelése

A recepció menüpont három feleletűl rendelkezik: Vendégek, Foglalások, és szoba tükrök. Ezen felületek megtekintéséhez és szerkesztéséhez szükséges megfelelő jogosultság lásd: 6.1. függelék.

Vendégek felület

Mire használom? Ezen az oldalon kezelhetem a rendszerben tárolt vendégeket. Új vendéget vehetek fel, módosíthatom a meglévő vendégek adatait, vagy deaktiválhatom őket. Foglalások felvétele során tudom majd őket hozzájuk rendelni.

Táblázat és szűrők Ez a felület megjelenítésben és funkcióiban hasonlít az alapadatokon belül található felületekre. Ugyanúgy egy szűrhető táblázat jelenik meg különböző táblázat műveletekkel (lásd 2.5.2.).

Specifikus mezők:

- Keresztnév
- Vezetéknév
- Email
- Telefonszám
- Származási ország
- Vendég típus: Felnőtt, gyerek
- Vendég címek (egymás után vesszővel elválasztva)
- Aktív-e

Keresztnév	Vezetéknév	E-mail	Telefonszám	Származási ország	Vendég típus	Vendég címek	Aktív	Műveletek
John	Doe	john.doe@email.com	+1-555-1001	Hungary	Felnőtt	VIP, Frequent Guest, Corp	Igen	...
Jane	Smith	jane.smith@email.com	+1-555-1002	Hungary	Felnőtt	Frequent Guest	Igen	...
Alice	Johnson	alice.johnson@email.com	+1-555-1003	Hungary	Felnőtt		Igen	...

(a) Vendégek elérése a menüből

(b) Vendégek táblázat

2.22. ábra. Vendégek felület

Műveletek:

- **Új vendég felvétele:** A táblázat feletti **Új vendég felvétele** gombbal hozhatók létre új vendégek.
- **Szerkesztés:** A sorok végén lévő ... gombra kattintva módosíthatom a meglévő vendégek adatait.
- **Törlés:** Szintén a ... gombon keresztül érhető el. Megerősítés után a vendég nem törlődik vélegesen, hanem inaktív státuszba kerül, így megmaradnak a korábbi foglalásai és adatai a rendszerben.

Foglalások felület

Mire használom? Ezen az oldalon kezelhetem a rendszerben tárolt foglalásokat.

Név	Szobás Nev alapján	Szobasz. Státsz alapján	Szobás Számla státsz	Szobás Számla státsz	Bejelentkezés dátuma	Kijelentkezés dátuma	Szobák	Letrás	Műveletek
Test Booking 5	Foglaló	Szinkronizált	2025.10.20	2025.10.24	1-A102	Automated test booking for devlek			
Test Booking 7	Foglaló	Szinkronizált	2025.11.12	2025.11.15	3-S301, 1-102	Automated test booking for devlek			
Test Booking 1	Vevő	Nincs számla	2025.10.26	2025.10.29	2-202	Automated test booking for devlek			
Test Booking 2	Kijelentkezett	Nincs számla	2025.11.03	2025.11.08	1-S101, 2-A201	Automated test booking for devlek			
Test Booking 6	Foglaló	Nincs számla	2025.11.05	2025.11.10	3-S201	Automated test booking for devlek			
Test Booking 8	Vevő	Nincs számla	2025.11.03	2025.11.07	2-S201	Automated test booking for devlek			
Test Booking 10	Nincs jelent meg	Nincs számla	2025.11.10	2025.11.15	2-S201	Automated test booking for devlek			
Test Booking 4	Kijelentkezett	Nincs szinkronizált	2025.10.19	2025.11.05	2-202	Automated test booking for devlek			
Test Booking 9	Várólistán	Nincs szinkronizált	2025.10.27	2025.11.13		Automated test booking for devlek			

(a) Foglalások elérése a menüből

(b) Foglalások táblázat

2.23. ábra. Foglalások felület

Táblázat és szűrők Ez a felület megjelenítésben és funkcióiban hasonlít az alapadatokon belül található felületekre. Ugyanúgy egy szűrhető táblázat jelenik meg különböző táblázat műveletekkel (lásd 2.5.2.).

Specifikus mezők:

- Név: a foglalás neve
- Aktiv-e
- Státsz: Foglalt, bejelentkezett, kijelentkezett, várólistán, blokkolt, nem jelent meg, törölt
- Számla státsz: nincsen számla, szinkronizált, nincs szinkronizálva
- Bejeltenkezási dátum (Check-in-date)
- Kijelentkezási dátum (Check-out-date)
- Szobák: vesszővel elválasztva egymás után
- Leirás

Műveletek:

- Új foglalás létehozása
- Meglévő foglalás információs ablak megnyitása
- Meglévő foglalás szerkesztése
- Meglévő foglalás státsz váltása
- Számla művelete: létrehozás, szinkronizálás és megtekintés
- Törlés

Új foglalás létehozása művelet:

1. Rákattintok a táblázatt feletti **Új foglalás létrehozása** gombra Ekkor egy többlépcsős űrlap fogad.

2. 1. lépés - Alapadatok megadása:

- Megadom a foglalás nevét
- Kiválasztom a bejelentkezési és kijelentkezési dátumot. Ha szabálytalan dátumokat adnék meg, a rendszer értesít
- Opcionálisan megjegyzést is írhatok
- (lásd 2.24. (a) ábra)

3. 2. lépés - Vendégek kiválasztása:

- Kiválasztom a rendszerben már meglévő vendégeket
- Ha olyan vendéget szeretném felvenni, aki még nincs a rendszerben, azt megtehetem ezen lépés során az **Új vendég hozzáadása** opcionálisan
- Vendégek felvételéről és megtekintéséről lásd: 2.5.3
- (lásd 2.24. (b) ábra)

4. 3. lépés - Szobák kiválasztása:

- Az elérhető szobák listájából választok. Csak azok a szobák jelennek meg, amelyek elérhetők a tartózkodás időtartama során és más foglalás nem foglalja őket
- Szükség esetén leellenőrizhetem, hogy az adott szobának megvan-e minden kért funkciója
- Szoba nélküli foglalás esetén a rendszer automatikusan **Várólistán** státuszt állít be a **Foglalt** státusz helyett
- (lásd 2.24. (c) ábra)

5. Ellenőrzöm az adatokat és elmentem a foglalást a **Mentés** gombbal

2. Felhasználói dokumentáció

(a) 1. lépés: Alapadatok

(b) 2. lépés: Vendégek

(c) 3. lépés: Szobák

2.24. ábra. Új foglalás létrehozásának lépései

Meglévő foglalás információs ablak megnyitása A sorok végén lévő ... gombra kattintva megnyomhatom az **Információ** gombot, ami megnyit egy részletes leírást az adott foglalásról. Itt megtekinthető a foglaláshoz kapcsolódó vendégek, szobák és számla. Emellett a foglalás alap- és metainformációi is megjelennek.

Meglévő foglalás szerkesztése A sorok végén lévő ... gombra kattintva megnyomhatom a **Szerkesztés** gombot, amivel a foglalás adataival előre kitöltött többlépcsős űrlap nyílik meg. Ugyanazokkal a lépésekkel rendelkezik, mint az **Új foglalás létrehozása** űrlap (lásd 2.24. ábra).

Meglévő foglalás státusz váltása A sorok végén lévő ... gombra kattintva megnyomhatom a **Státusz módosítás** gombot, amivel megnyílik egy egyetlen mezővel rendelkező űrlap, ahol kiválaszthatom a kívánt státuszt (foglalt, bejelentkezett, ki-jelentkezett, várólistán, blokkolt, nem jelent meg, törölt).

Nem minden esetben fogadja el a rendszer a státusz módosítást. Például ha egy kijelentkezett státusból próbálok foglaltra visszaváltani, de a foglalásban lévő szobákat már valaki más lefoglalta, ilyenkor a rendszer tájékoztat a hibáról. Emellett ha egy foglalást várólistás státuszba helyezek, az megszünteti a szobáival való kapcsolatot.

Számla műveletek: létrehozás, szinkronizálás és megtekintés A sorok végén lévő ... gombra kattintva találom meg az alábbi gombokat:

- **Számla létrehozása:** Létrehoz egy új számlát és hozzárendeli ehhez a foglaláshoz
- **Számla szinkronizálás:** Szinkronizálja a számlához kiszámolt teljes és szobánkénti árat a foglalás adataival. Akkor van rá szükség, ha a foglalás adatai változtak, pl. még két napot szeretnének maradni a vendégek
- **Számla megtekintése:** Átnavigál a számlák felületre (lásd 2.5.4. oldal)

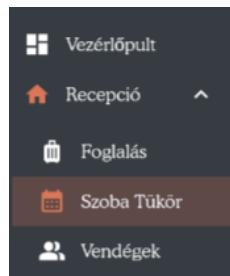
Törlés A sorok végén lévő ... gombra kattintva megnyomhatom a **Törlés** gombot, amivel megerősítés után törlésre kerül az adott foglalás. Későbbi visszakereshetőség érdekében a foglalás tárolva marad, csak nem lesz többé aktív.

Szoba tükr felület

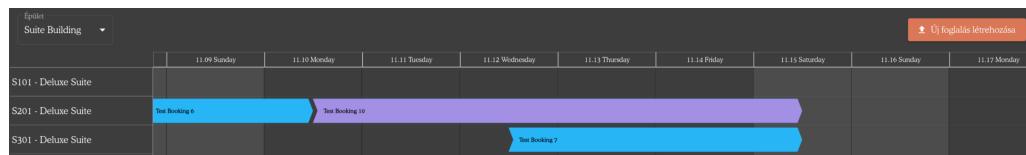
Mire használom? Ezen az oldalon egy vizuális reprezentációt látok az aktív foglalásokról naptár nézetben, szobánként és épületenként. A szobatükör időrendi sorrendben jeleníti meg a szobák foglaltságát, így gyorsan áttekinthetők a szabad kapacitások és az átfedések.

Műveletek A szobatükör felületen elérhető műveletek megegyeznek a Foglalások felületén leírtakkal:

- **Új foglalás létrehozása:** Ugyanaz a többlépcsős folyamat, mint a Foglalások felületen (lásd 2.24. ábra)
- **Foglalás információk megtekintése:** Dupla kattintással megnyílik a részesített információs ablak, ahogy a Foglalások felületen is (lásd 2.5.3)
- **Szűrés:** A naptár felett kiválaszthatom, hogy melyik épület szobáit szeretném megjeleníteni



(a) Szobatükör
elérése a menüből



(b) Szobatükör nézet

2.25. ábra. Szobatükör felület

2.5.4. Számla műveletek kezelése

Számla művelet három felületen keresztül tudom kezelní: Szolgáltatások, Számlák, és ÁFA kulcsok felület. Ezen felületek megtekintéséhez és szerkesztéséhez szükséges megfelelő jogosultság lásd: 6.1. függelék.

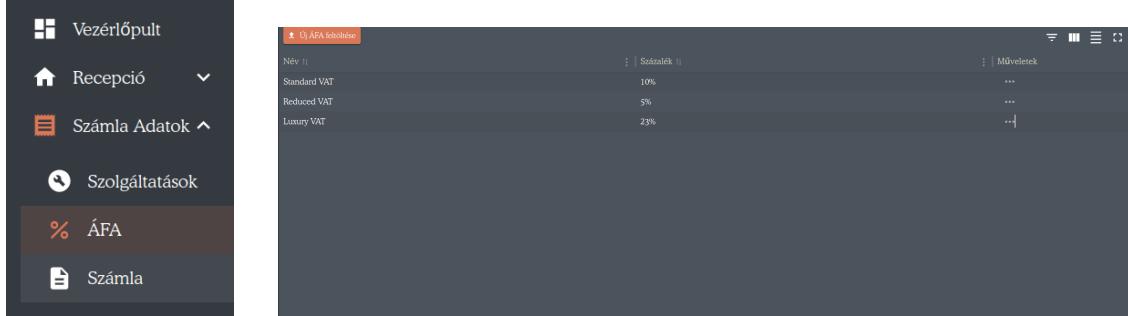
ÁFA kulcsok felület

Mire használom? Ezen az oldalon kezelhetem a rendszerben használt ÁFA kulcsokat, amelyeket később a szolgáltatások árának kiszámításához használok fel.

Táblázat és szűrők Ez a felület megjelenítésben és funkcióiban hasonlít az alapadatokon belül található felületekre. Ugyanúgy egy szűrhető táblázat jelenik meg különböző táblázat műveletekkel (lásd 2.5.2. oldal).

Specifikus mezők:

- ÁFA kulcs neve (pl. Normál, Kedvezményes)
- ÁFA mértéke (%)



The image consists of two screenshots of a software application. The left screenshot, labeled (a), shows a dark-themed sidebar menu with items like 'Vezérlőpult', 'Recepció', 'Számla Adatok', 'Szolgáltatások', '% ÁFA' (which is highlighted in a darker shade), and 'Számla'. The right screenshot, labeled (b), shows a table titled 'Új ÁFA kulcsok' with columns 'Név' and 'Százalék'. It contains three rows: 'Standard VAT' (10%), 'Reduced VAT' (5%), and 'Luxury VAT' (23%). There are also 'Műveletek' (Actions) columns with three dots next to each row.

(a) ÁFA kulcsok elérése a menüből

(b) ÁFA kulcsok táblázat

2.26. ábra. ÁFA kulcsok felület

Műveletek:

- **Új ÁFA kulcs felvétele:** A táblázat feletti Új gombbal hozhatók létre új ÁFA kulcsok.
- **Szerkesztés:** A sorok végén lévő ... gombra kattintva módosíthatom a meglévő ÁFA kulcsok adatait.
- **Törlés:** Szintén a ... gombon keresztül érhető el, de csak akkor engedélyezett, ha az ÁFA kulcshoz nincs hozzárendelve szolgáltatás. Ellenkező esetben a rendszer hibaüzenetet jelenít meg.

Szolgáltatások felület

Mire használom? Ezen az oldalon kezelhetem a rendszerben elérhető szolgáltatásokat, amelyeket később a számlákhoz rendelek hozzá. Ezek a szolgáltatások építik fel a végső számlát (pl. szobák költsége, spa csomag, stb.).

Táblázat és szűrők Ez a felület megjelenítésben és funkciókban hasonlít az alapadatokon belül található felületekre. Ugyanúgy egy szűrhető táblázat jelenik meg különböző táblázat műveletekkel (lásd 2.5.2).

Specifikus mezők:

- Név
- Leírás
- Költség

- ÁFA kulcs



(a) Szolgáltatások elérése a menüből

(b) Szolgáltatások táblázat

2.27. ábra. Szolgáltatások felület

Műveletek:

- **Új szolgáltatás felvétele:** A táblázat feletti Új gombbal hozhatók létre új szolgáltatások.
 - **Szerkesztés:** A sorok végén lévő ... gombra kattintva módosíthatom a meglévő szolgáltatások adatait.
 - **Törlés:** Szintén a ... gombon keresztül érhető el. Megerősítés szükséges a sikeres törléshez. Nem törölhetek olyan szolgáltatást, amit használ egy számla.

Speciális szolgáltatás: Rooms aggregated cost

A "Rooms aggregated cost" szolgáltatás speciális kezelést igényel:

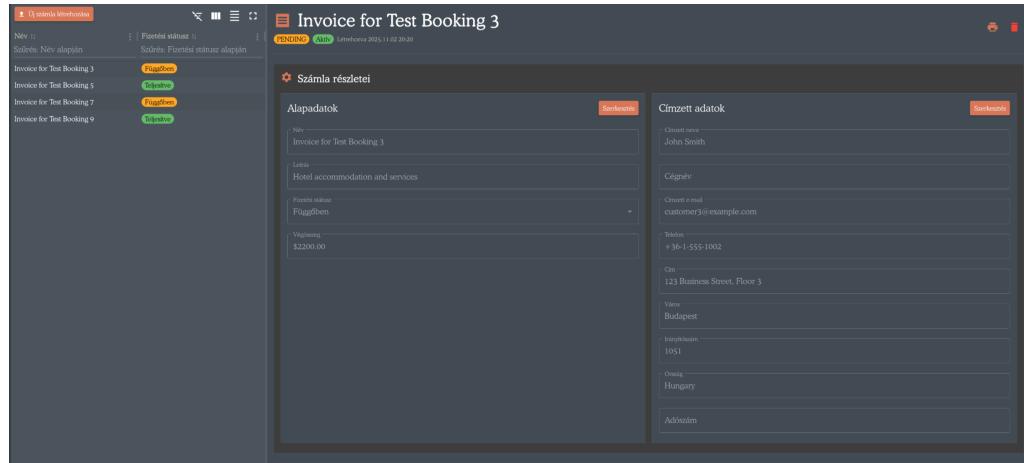
- **Nem törölhető:** Ez a szolgáltatás alapértelmezett, és nem lehet törölni a rendszerből.
 - **Korlátozott szerkesztés:** Csak az ÁFA kulcsa módosítható.
 - **Automatikus árszámítás:** A költsége automatikusan kiszámítódik az adott számlában szereplő szobák alapján, manuálisan nem állítható.

Számlák felület

Mire használom? Ezen az oldalon kezelhetem a rendszerben lévő számlákat. Megtekinthetek, létrehozhatok, szerkeszthetek, nyomtathatok és törölhetek számlákat. A számlák a foglalásokhoz kapcsolódó szobák és egyéb szolgáltatások költségeit tartalmazzák az alap számlainformációk mellett.

A Számlák felület 2 fő részből áll: bal oldalt egy vékony szűrhető táblázat (lásd 2.5.2. oldal szűrési és beállítási lehetőségeit), amely listázza a számlákat, jobb oldalt pedig a munkafelület. Ha a bal oldali táblázatban kattintással kiválasztok egy

számlát, az megnyílik minden adatával a jobb oldalon. A részek mérete állítható az elválasztó húzásával.



(a) Számlák táblázat - felső rész

This screenshot shows the lower part of the invoice interface. It includes a table for 'Szolgáltatások' (Services) with columns for 'Név', 'Leírás', 'Költség', and 'ÁFA'. The table lists Room Service (\$45.00), Rooms aggregated cost (\$450.00), and Rooms aggregated cost (\$1750.00), totaling \$1400.00. The VAT column shows Standard VAT (10%) for each row. Below the table, it says 'Összesen ÁFA nélkül: \$3645.00' and 'Összesen ÁFA-val: \$4009.50'. At the bottom, there is a pagination message 'Sorok oldalanként 10 ▾ 1-4 összesen 4 < >' and a dashed box labeled 'Foglalás hozzáadása' (Add booking).

(b) Számlák táblázat - alsó rész

2.28. ábra. Számlák felület teljes nézet

Specifikus mezők:

- Név
- Fizetési státusz: Függőben, Teljesített, Törölt
- Leírás
- Végösszeg
- Címzett adatok: számlázott neve vagy cégnév, email, telefon, cím, adószám, ország
- Szolgáltatások lista
- Foglalások lista

Műveletek

- Új számla létrehozása
- Meglévő számla alapadatainak szerkesztése
- Meglévő számla címzett adatainak szerkesztése
- Meglévő számla nyomtatása
- Meglévő számla szolgáltatásainak szerkesztése
- Meglévő számla foglalásainak szerkesztése
- Törlés

Új számla létrehozása Rákattintok a bal részen lévő táblázat feletti **Új számla létrehozása** gombra, amivel megnyílik egy egyetlen mezővel rendelkező űrlap, ahol a számla nevét megadhatom. Mentés után létrejön egy új üres számla.

Meglévő számla alapadatainak szerkesztése A jobb oldali munkafelületen az Alapadatok felett rákattintok a **Szerkesztés** gombra, és így már átírhatom a számla alapadatait. Új nevet adhatok neki, leírást, vagy akár a státuszát is módosíthatom. Miután befejeztem a szerkesztést, elmenthetem a változtatásaimat a **Mentés** gombbal, ami a szerkesztés gomb helyett jelent meg, vagy visszaállíthatom az eredeti számla adatait a mellette lévő **Visszaállítás** gombbal.

Meglévő számla címzett adatainak szerkesztése A jobb oldali munkafelületen a Címzett adatai felett rákattintok a **Szerkesztés** gombra, és így módosíthatom a számla címzett adatait (név, cégnév, email, telefon, cím, adószám, ország). Miután befejeztem a szerkesztést, elmenthetem a változtatásaimat a **Mentés** gombbal, vagy visszaállíthatom az eredeti adatakat a **Visszaállítás** gombbal.

Meglévő számla nyomtatása A számla neve mellett lévő **Nyomtatás** gombra kattintva kinyomtathatom a számlát



2.29. ábra. Számla nyomtatás gomb

Meglévő számla szolgáltatásainak szerkesztése A jobb oldali munkafelületen a Szolgáltatások táblázatban (lásd 2.28. (b) ábra) a + gombbal új szolgáltatást adhatok hozzá, amelyet kiválaszthatok a rendszerben lévő szolgáltatások közül. A táblázat sorok végén lévő X gombbal törölhetek szolgáltatást.

Fontos: A szoba ár szolgáltatást ("Rooms aggregated cost") nem törölhetem, mert az automatikusan a foglalásokkal jön és megy.

Meglévő számla foglalásainak szerkesztése A jobb oldali munkafelületen a Foglalások táblázatban (lásd 2.28. (b) ábra) a + gombbal új foglalást adhatok a számlához. Kiválaszthatok a rendszerben lévő foglalások közül olyan foglalást, ami-nek még nincs számlája. Egy számlának lehet 0, 1 vagy több foglalása is.

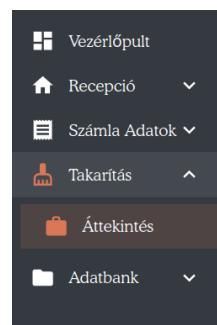
Egy adott foglalásra ha ráviszem az egeret, tudom eltávolítani a számláról az X gombbal. A foglalások módosítása után a teljes ár és a szolgáltatások automatikusan újraszámolódnak.

Törlés A számla neve mellett lévő kuka gombbal, megerősítés után törölhetem a számlát.



2.30. ábra. Számla törlése gomb

2.5.5. Takarítások kezelése



(a) Takarítások elérése a menüből

Feladat létrehozása		Feladataim megjelenítése					
Szoba	Hozzárendelte	Státusz	Prioritás	Megjegyzés	Hozzárendelés dátuma	Befejezés dátuma	Műveletek
Szürés: Szoba ala...	Szürés: Hozzáren...	Szürés: Státusz ale...	Szürés: Prioritás a...	Szürés: Megjegyz...	Szürés: Hozzárendelés dátum...	Szürés: Befejezés ...	
Main Building 201	Admin User	Teendő	Közepes		2025.10.30		...
Main Building 202	Admin User	Folyamatban	Magas		2025.10.30		...
Annex Building A102	Admin User	Kész	Magas		2025.10.30	2025.10.30	...

(b) Takarítások áttekintése ablak

2.31. ábra. Takarítások felület

Ezen a felületen vehetek fel egyes szobákra takarítási kérést, rendelhetem hozzá egyes munkatársakhoz az adott kérést, vagy egyes munkatársaim jelezhetik, hogy a

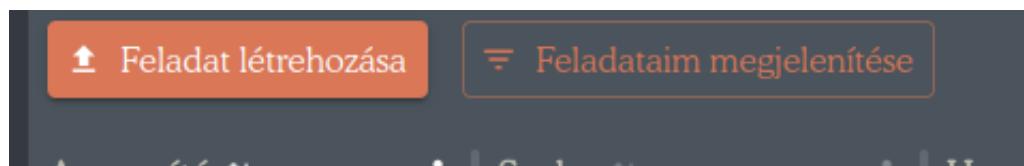
kérés végbement és a szoba kitakarítva (ebben az esetben a szoba státusza automatikusan tiszta lesz).

Kérésenként megjelenített mezők a következők:

1. Szoba: Szoba neve és épület
2. Hozzárendelve: Melyik munkatársnak kell a feladatot elvégeznie
3. Státusz: hogy halad a takarítás? Kész, teendő vagy folyamatban
4. Prioritás: alacsony, közepes, magas vagy sürgős
5. Megjegyzés
6. Hozzárendelés időpontja: mikor osztottam ki a feladatot
7. Befejezés dátuma: mikor lett a feladat befejezve

Takarítás felület műveletei A felületen 7 műveletet hajthatok végre:

1. Feladat létrehozása (2.32 (a))
2. Feladataim megjelenítése (2.32 (a))
3. Szerkesztés (2.32 (b))
4. Törlés (2.32 (b))
5. Kész (2.32 (b))
6. Folyamatban (2.32 (b))
7. Teendő (2.32 (b))



(a) Táblázati műveletek

Annex Building A102	Main Building 202	Admin User	Nincs hozzárendelve	Kész	Folyamatban	Teendő	Alacsony	2025.10.30	2025.10.30	...
										...

(b) Soronkénti műveletek

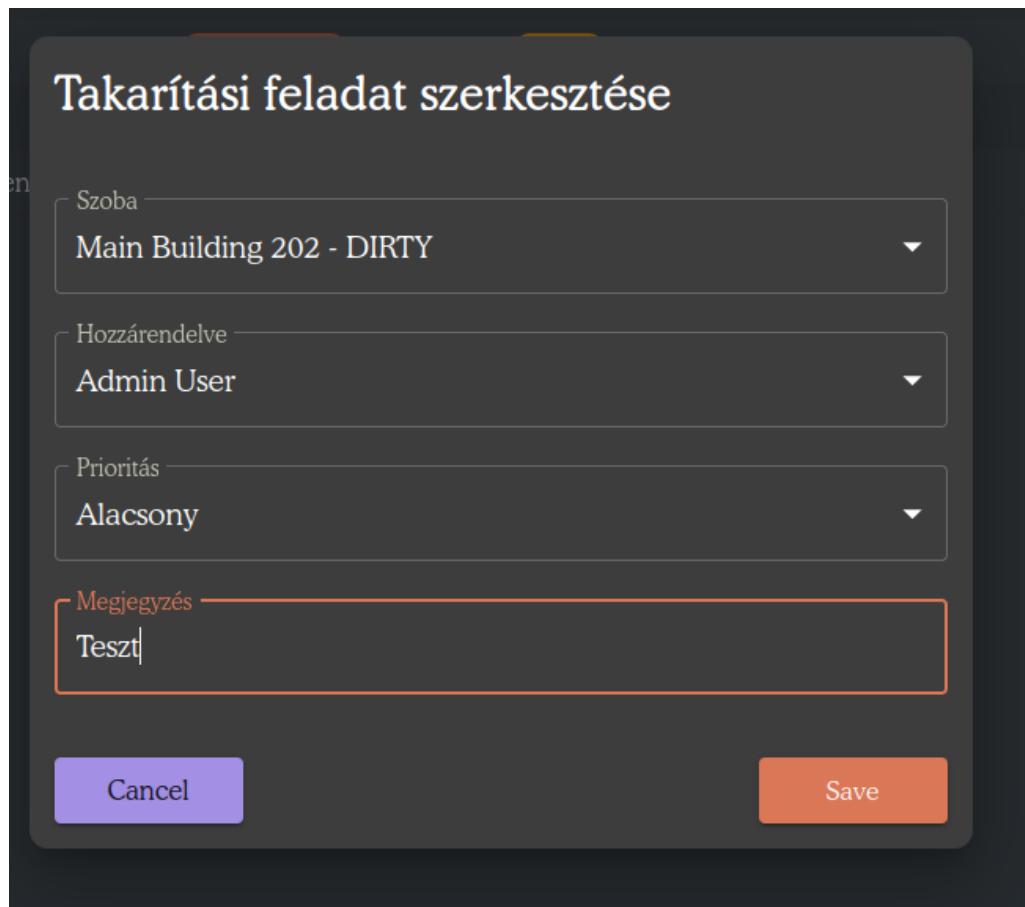
2.32. ábra. Takarítás felület műveletei

Kész, folyamatban, teendő Kattintásra megerősítés nélkül átállítódik a feladat státusza a kiválasztott értékre. Amennyiben ez az érték kész, akkor a szoba státusza is tisztára módosul, és a feladat kap egy befejezés dátumot.

Törlés Megerősítés után a feladat törlődik.

Feladataim megjelenítése Leszűri a táblázatot csak azon feladatokra, amelyek hozzám tartoznak. Másodszori rákattintásra a szűrést megszünteti.

Szerkesztés és új feladat felvétele A kiválasztott gomb megnyomása után csak címében és kezdeti adataiban eltérő ablakok fogadnak, ahol meg tudom adni a takarítandó szobát, sürgősséget, hozzárendelt munkatársamat és akár opcionális megjegyzést.



2.33. ábra. Feladat szerkesztése

Szerkeszteni, törlni vagy új feladatot felvenni csak a megfelelő jogosultágal lehet lásd: 6.1. függelék.

2.6. Hibaüzenetek, figyelmeztetések és visszajelzések

A rendszer különböző helyzetekben hibaüzenetekkel, figyelmeztetésekkel és visszajelzésekkel tájékoztat használata során. Ezek segítenek az űrlapok helyes ki-

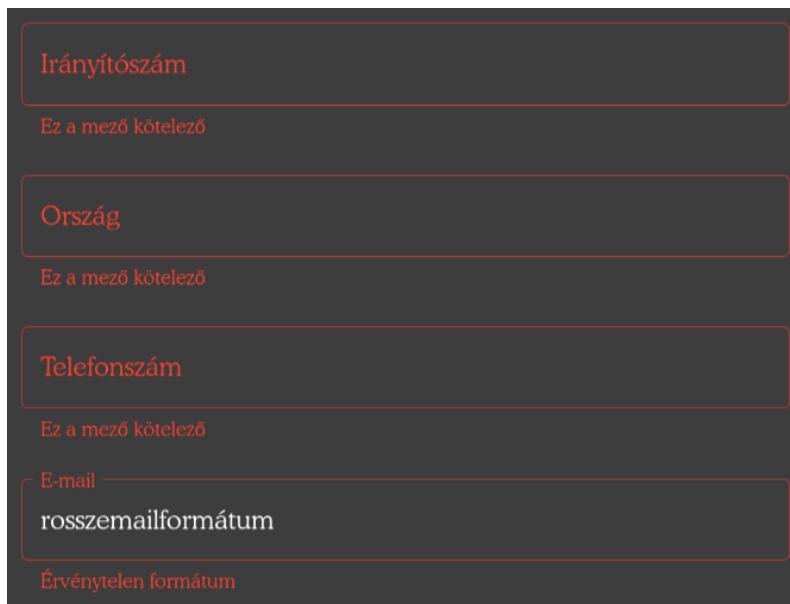
töltésében, műveletek eredményének megjelenítésébe.

2.6.1. Úrlap validációs hibák

Ezek az üzenetek akkor jelennek meg, amikor hibásan vagy hiányosan töltöm ki az ūrlapot. A rendszer nem engedi a mentést, amíg a hibákat nem javítom.

Gyakori példák:

- **Kötelező mező hiánya:** "Ez a mező kötelező"
- **Hibás email vagy telefonszám formátum:** "Érvénytelen formátum"
- **Hibás dátum:** "A kijelentkezés dátumának a bejelentkezés dátuma után kell lennie"



2.34. ábra. Példa ūrlap validációs hibákra

2.6.2. Műveletek végrehajtási üzenetek

A rendszer minden művelet után visszajelzést ad arról, hogy a művelet sikeres volt-e vagy hiba történt. A sikeres műveletek zöld háttérrel, a hibák piros háttérrel jelennek meg.



2.35. ábra. Művelet végrehajtási üzenetek

Gyakori hibaüzenetek:

• Általános hibák:

- "Váratlan hiba történt"
- "Hálózati hiba"
- "Erőforrás nem található"
- "Az erőforrás már létezik"
- "Hozzáférés megtagadva"

• Vendégek kezelésekor:

- "Vendég nem található"
- "Ezzel az e-mail címmel már létezik vendég"
- "Ezzel a telefonszámmal már létezik vendég"

• Foglalások kezelésekor:

- "Foglalás nem található"
- "Érvénytelen foglalási státuszváltás"

• Számlák kezelésekor:

- "Számla nem található"
- "Nem sikerült szinkronizálni a számlát a foglalásokkal"
- "EZ a foglalás egy másik számlához tartozik"
- "Hiba a számla összegének kiszámításakor"

• Szobák kezelésekor:

- "Szoba nem található"
- "A szoba nem elérhető"
- "Nem törölhető foglalásokkal rendelkező szoba"

• Alapadatok törlésekor:

- "Nem törölhető szobákkal rendelkező épület"
- "Nem törölhető szobákkal rendelkező szobatípus"
- "Nem törölhető szolgáltatás modellekkel rendelkező ÁFA"
- "Nem törölhető vendégekkel rendelkező vendég címke"
- "Nem törölhető szobatípusokkal rendelkező ágytípus"

• Bejelentkezéskor és jogosultságok:

- "Érvénytelen bejelentkezási adatok"
- "Elégtelen jogosultságok"
- "A fiók le van tiltva"

3. fejezet

Fejlesztői dokumentáció

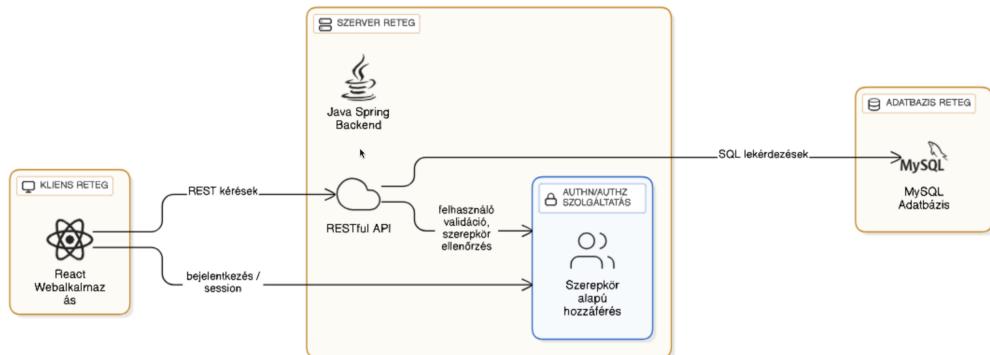
3.1. Bevezetés

A fejlesztői dokumentációban bemutatom a webalapú hotelmrendszer és foglalási rendszer tervezési folyamatát, architektúráját, implementációs döntéseit. A dokumentáció alapján a rendszer továbbfejleszthető és karbantartható.

3.2. Rendszer architektúra

A rendszer háromrétegű architektúrát használ:

- **Kliens réteg (Frontend):** React alapú webalkalmazás
- **Szerver réteg (Backend):** Java Spring és restful api
- **Adatbázis réteg:** MySQL relációs adatbázis



3.1. ábra. Rendszer háromrétegű architektúrája

Rétegek közötti kommunikáció

- **Frontend → Backend:** HTTP kérések (GET, POST, PUT, DELETE, PATCH) Axios kliens segítségével frontenden
- **Backend → Adatbázis:** JPA/Hibernate ORM
- **Autentikáció:** JWT (JSON Web Token) alapú hitelesítés
- **API prefix:** /api - minden backend endpoint ezzel kezdődik

Backend felépítése

A backend háromrétegű (Controller-Service-Repository) architektúrát követ:

- **Controller réteg:** HTTP kérések fogadása, válaszok küldése
- **Service réteg:** Üzleti logika megvalósítása
- **Repository réteg:** Adatbázis elérés JPA-n keresztül
- **Model réteg:** JPA entitások az adatbázis táblák reprezentálására
- **DTO réteg:** Adatátviteli rétegek között
- **Security réteg:** JWT szűrők és Spring Security konfiguráció

Frontend felépítése

A frontend moduláris felépítésű:

- **Pages (Oldalak):** 23 oldal-szintű komponens
- **Components (Komponensek):** Újrafelhasználható UI elemek
- **API modulok:** 19 API kliens modul (amenityApi, bookingApi, stb.)
- **State (Állapot):** Jotai atom-ok globális állapotkezelésre
- **Router:** React Router külön auth és fő router-rel
- **Theme:** Material-UI téma testreszabás
- **i18n:** Többnyelvű támogatás (magyar, angol)

Konténerizáció és deployment

A rendszer Docker konténerekben fut:

- **Frontend konténer:** Node.js 22 Alpine, Vite build, port 3000
- **Backend konténer:** OpenJDK 17, Maven build, port 8080
- **Adatbázis konténer:** MySQL 8.0, port 3306, perzisztens volume

3.3. Technológiai stack

3.3.1. Frontend technológiák

Alap keretrendszer és nyelv

- **React 18.3.1:** Komponens alapú UI könyvtár
- **TypeScript 5.8.3:** Típusbiztos JavaScript
- **Vite 7.1.7:** Modern build eszköz SWC-vel

UI keretrendszer és stílusok

- **Material-UI (MUI) 7.3.4:** Átfogó komponens könyvtár
 - @mui/material: Alap komponensek (gombok, táblázatok, dialógusok)
 - @mui/icons-material: Ikon könyvtár
 - @mui/x-date-pickers: Dátum/idő választók
 - @mui/x-charts: Grafikonok (Dashboard használja)
- **Emotion 11.14:** CSS-in-JS stílusrendszer

Állapotkezelés és routing

- **Jotai 2.15.0:** Atomi állapotkezelés (könnyűsúlyú alternative a Reduxnak)
- **React Router 7.9.3:** Kliens oldali navigáció

API kommunikáció és adatkezelés

- **Axios 1.12.2:** HTTP kliens JWT interceptor-okkal
- **date-fns 4.1.0:** Dátum és idő kezelés
- **jwt-decode 4.0.0:** JWT token dekódolás

Táblázatok és vizualizáció

- **material-react-table 3.2.1:** Haladó táblázat komponens szűrésekkel és rendezésekkel
- **react-calendar-timeline 0.28.0:** Szobatükör naptár nézet

Többnyelvűség és kódminőség

- **i18next 25.5.2 + react-i18next 15.7.3:** Többnyelvű támogatás (magyar, angol)
- **Biome 2.2.5:** Kód formázó és linter (ESLint + Prettier egyben)

3.3.2. Backend technológiák

Keretrendszer és nyelv

- **Spring Boot 3.4.3:** Java alkalmazás keretrendszer
- **Java 17:** Programozási nyelv
- **Maven:** Dependency és build management

Spring modulok

- spring-boot-starter-web
- spring-boot-starter-data-jpa
- spring-boot-starter-validation
- spring-boot-starter-security

Biztonság

- **Spring Security 6.4.4:** Hitelesítés és jogosultságkezelés
- **JJWT 0.11.5:** JWT token generálás és validáció

Adatbázis elérés

- **Spring Data JPA**
- **MySQL Connector/J**

Segédkönyvtárak

- **Lombok 1.18+:** Boilerplate kód csökkentés (@Getter, @Setter stb.)
- **ModelMapper 3.0.0:** DTO és entitás közötti mapping
- **OpenPDF 2.0.0:** PDF generálás számlákhoz

API dokumentáció

- **springdoc-openapi 2.2.0:** Swagger/OpenAPI UI a /swagger végponton

3.3.3. Adatbázis technológia

- **MySQL 8.0:** Relációs adatbázis
- **Adatbázis név:** hotelpms
- **Hibernate DDL mód:** update (automatikus séma frissítés)

3.3.4. Fejlesztői eszközök

- **Git:** Verziókezelés
- **Node.js:** Frontend fejlesztési környezet
- **Java JDK 17:** Backend fejlesztési környezet
- **Docker Desktop:** Konténer futtatás

3.3.5. Technológiai döntések indoklása

Miért React + TypeScript?

- **React:** Komponens alapú UI fejlesztés, elterjett és népszerűbb
- **TypeScript:** Típusbiztonság, jobb fejlesztői élmény
- Széleskörű támogatottság és dokumentáció

Miért Spring Boot?

- Production-ready funkcionálisok beépítve (security, validation stb.)
- Széleskörű támogatottság és dokumentáció
- JPA/Hibernate egyszerűsíti az adatbázis műveletek kezelését

Miért MySQL?

- Relációs adatbázis ACID tulajdonságokkal
- Komplex kapcsolatok kezelésére volt szükség (számlák, foglalások, vendégek)
- Széleskörű támogatottság és dokumentáció

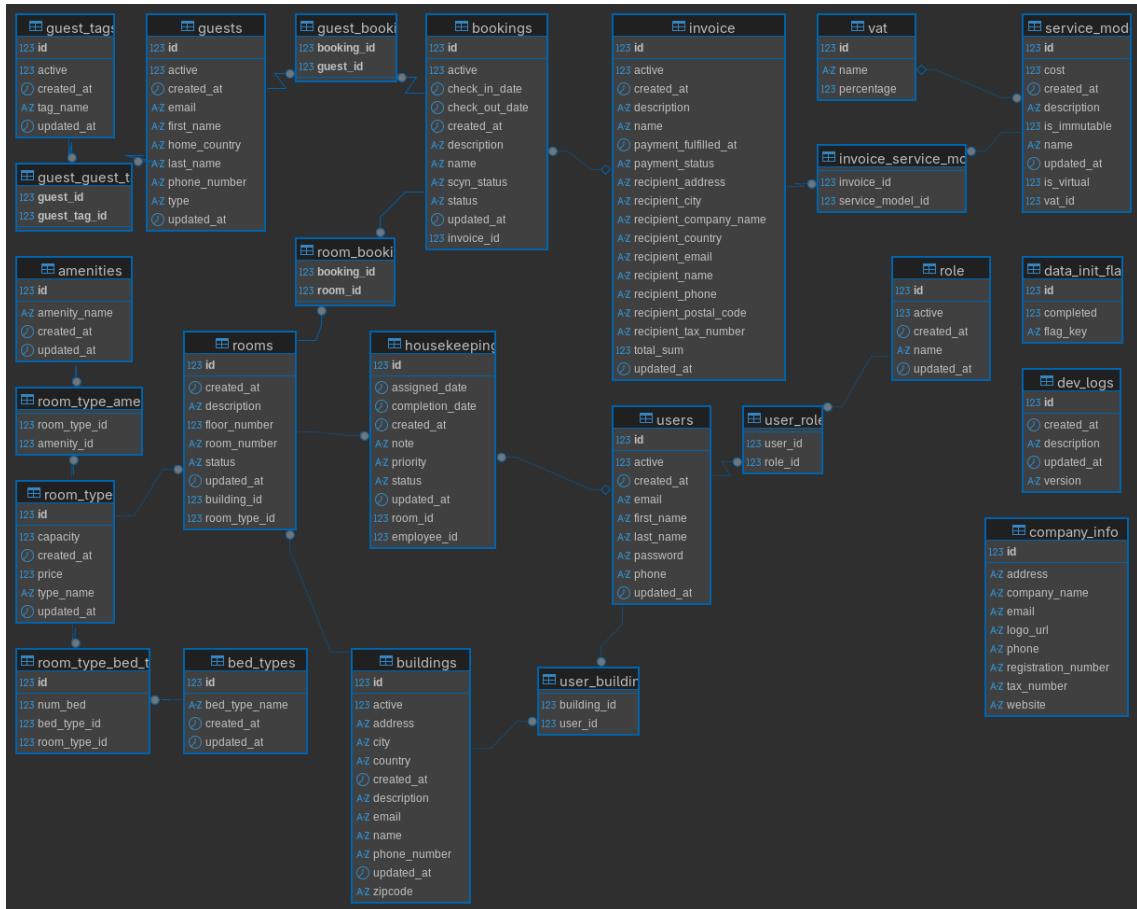
3.4. Adatbázis felépitése

A rendszer relációs adatbázist használ a szálloda adatainak tárolására. Az adatbázis MySQL 8.0-t használ, és 18 fő entitásból áll.

3.4.1. Adatbázis konfiguráció

- **Adatbázis neve:** hotelpms
- **Séma kezelés:** Hibernate auto-update (fejlesztés), manual (produkció)
- **Karakter kódolás:** UTF-8
- **Port:** 3306 (alapértelmezett MySQL port)

3.4.2. Entitás-kapcsolat diagram (ERD)



3.2. ábra. Adatbázis Entitás-Kapcsolat Diagram

3.4.3. Entitások áttekintése

A rendszer 18 entitást tartalmaz, amelyeket a következő csoportokba sorlom:

Felhasználói és biztonsági entitások

- **User** - Rendszer felhasználók (recepciósok, adminok, stb.)
- **Role** - Szerepkörök (N:M kapcsolat User-rel)

Vendég kezelés

- **Guest** - Vendégek adatai
- **GuestTag** - Vendég címkék (törzsvendég, VIP, stb.)

Szálloda struktúra

- **Building** - Épületek

- **Room** - Szobák
- **RoomType** - Szobatípusok (egyágyas, kétágyas, apartman)
- **BedType** - Ágytípusok (egyszemélyes, franciaágy)
- **Amenity** - Szoba szolgáltatások (WiFi, klíma, minibar)

Foglalás kezelés

- **Booking** - Foglalások
- **Booking kapcsolótáblák:** BookingGuest (N:M), BookingRoom (N:M)

Számlázás

- **Invoice** - Számlák
- **ServiceModel** - Szolgáltatások (szoba ár, spa, minibar)
- **Vat** - ÁFA kulcsok
- **Invoice kapcsolótáblák:** InvoiceBooking (N:M), InvoiceService (N:M)

Takarítás

- **Housekeeping** - Takarítási feladatok

Rendszer konfiguráció

- **CompanyInfo** - Cég adatok (egyetlen rekord)
- **DataInitFlag** - Adat inicializálási flag: minta adatokkal fel kell-e tölteni az adatbázist?
- **DevLog** - Változásnapló (verzió történet)

3.4.4. Részletes entitás leírások

Itt bemutatom a rendszer legfontosabb entitásait részletesen, csak a kritikus táblákra térek ki. **Automatizmusok:** A legtöbb entitásban szerepelnek `created_at` és `updated_at` mezők. Ezek mindenhol ugyanúgy működnek:

- @PrePersist: `createdAt` és `updatedAt` beállítása aktuális időre
- @PreUpdate: `updatedAt` frissítése módosításkor

User tábla

A `users` tábla tárolja a rendszer felhasználóit (recepziósok, adminisztrátorok, stb.).

Mező	Típus	Megszorítás	Leírás
<code>id</code>	BIGINT	PK, AUTO_INCREMENT	Elsődleges kulcs
<code>email</code>	VARCHAR(255)	NOT NULL, UNIQUE	Email cím (természetes kulcs)
<code>first_name</code>	VARCHAR(255)	NULL	Keresztnév
<code>last_name</code>	VARCHAR(255)	NULL	Vezetéknév
<code>phone</code>	VARCHAR(255)	NOT NULL, UNIQUE	Telefonszám
<code>password</code>	VARCHAR(255)	NOT NULL	BCrypt hash jelszó
<code>active</code>	BOOLEAN	DEFAULT TRUE	Aktív-e a felhasználó?
<code>created_at</code>	TIMESTAMP	NULL	Létrehozás időpontja
<code>updated_at</code>	TIMESTAMP	NULL	Utolsó módosítás

3.1. táblázat. User tábla szerkezete

Kapcsolatok:

- N:M kapcsolat `Role` táblával: Milyen jogosultásgokal rendelkezik a felhasználó (`user_roles` kapcsolótábla)
- N:M kapcsolat `Building` táblával: Melyik épülethez tartozik a felhasználó (`mappedBy: users`)

Guest tábla

A `guests` tábla a vendégek személyes adatait tárolja.

Mező	Típus	Megszorítás	Leírás
<code>id</code>	BIGINT	PK, AUTO_INCREMENT	Elsődleges kulcs
<code>first_name</code>	VARCHAR(255)	NOT NULL	Keresztnév
<code>last_name</code>	VARCHAR(255)	NOT NULL	Vezetéknév
<code>email</code>	VARCHAR(255)	NOT NULL, UNIQUE	Email cím
<code>phone_number</code>	VARCHAR(255)	NOT NULL, UNIQUE	Telefonszám
<code>home_country</code>	VARCHAR(255)	NULL	Származási ország
<code>type</code>	VARCHAR(20)	NOT NULL	ENUM: GuestType
<code>active</code>	BOOLEAN	DEFAULT TRUE	Aktív-e a vendég?
<code>created_at</code>	TIMESTAMP	NULL	Létrehozás időpontja
<code>updated_at</code>	TIMESTAMP	NULL	Utolsó módosítás

3.2. táblázat. Guest tábla szerkezete

Kapcsolatok:

- N:M kapcsolat Booking táblával (`guest_booking`)
- N:M kapcsolat GuestTag táblával (`guest_guest_tags`)

Enumok:

- `type`: GuestTypeEnum értékei: Adult (Felnött), Child (Gyerek)

Booking tábla

A bookings tábla kezeli a szállodai foglalásokat.

Mező	Típus	Megszorítás	Leírás
id	BIGINT	PK, AUTO_INCREMENT	Elsődleges kulcs
active	BOOLEAN	NOT NULL	Aktív-e a foglalás?
status	VARCHAR(20)	NULL	ENUM: BookingStatus
scyn_status	VARCHAR(20)	NULL	ENUM: BookingInvoice
check_in_date	DATE	NOT NULL	Bejelentkezés dátuma
check_out_date	DATE	NOT NULL	Kijelentkezés dátuma
name	VARCHAR(255)	NOT NULL	Foglalás neve
description	TEXT	NULL	Leírás, megjegyzések
invoice_id	BIGINT	FK(Invoice.id)	Kapcsolódó számla
created_at	TIMESTAMP	NULL	Létrehozás időpontja
updated_at	TIMESTAMP	NULL	Utolsó módosítás

3.3. táblázat. Booking tábla szerkezete

Kapcsolatok:

- N:M kapcsolat Guest táblával (guest_booking)
- N:M kapcsolat Room táblával (room_booking)
- N:1 kapcsolat Invoice táblával (invoice_id FK)

Enumok:

- status: BookingStatusEnum (pl. RESERVED, CHECKED_IN, CHECKED_OUT, CANCELLED, NO_SHOW, WAITLISTED, BLOCKED)
- scynStatus: BookingInvoiceEnum (NOT_SYNCED, NO_INVOICE, SYNCED)

Room tábla

A `rooms` tábla a szálloda szobáit kezeli.

Mező	Típus	Megszorítás	Leírás
id	BIGINT	PK, AUTO_INCREMENT	Elsődleges kulcs
room_number	VARCHAR(255)	NOT NULL	Szobaszám
floor_number	INT	NULL	Emelet száma
status	VARCHAR(20)	NULL	ENUM: RoomStatus
description	TEXT	NULL	Szoba leírása
room_type_id	BIGINT	FK(RoomType.id), NOT NULL	Szobatípus
building_id	BIGINT	FK(Building.id), NOT NULL	Épület
created_at	TIMESTAMP	NULL	Létrehozás időpontja
updated_at	TIMESTAMP	NULL	Utolsó módosítás

3.4. táblázat. Room tábla szerkezete

Kapcsolatok:

- N:1 kapcsolat RoomType táblával (room_type_id FK)
- N:1 kapcsolat Building táblával (building_id FK)
- 1:N kapcsolat Housekeeping táblával
- N:M kapcsolat Booking táblával (`room_booking`)

Enumok:

- `status`: RoomStatusEnum (OUT_OF_SERVICE, CLEAN, DIRTY)

Invoice tábla

Az invoice tábla a számlák adatait tárolja.

3.5. táblázat. Invoice tábla szerkezete

Mező	Típus	Megszorítás	Leírás
id	BIGINT	PK, AUTO_INCREMENT	Elsődleges kulcs
name	VARCHAR(255)	NOT NULL	Számla megnevezése
description	VARCHAR(255)	NULL	Leírás
payment_status	VARCHAR(20)	NULL	ENUM: PaymentStatus
payment_fulfilled_at	TIMESTAMP	NULL	Fizetés teljesítésének időpontja
total_sum	DOUBLE	NULL	Végösszeg
recipient_name	VARCHAR(255)	NULL	Címzett neve
recipient_company_name	VARCHAR(255)	NULL	Címzett cégnéve
recipient_address	VARCHAR(255)	NULL	Címzett címe
recipient_city	VARCHAR(255)	NULL	Címzett városa
recipient_postal_code	VARCHAR(255)	NULL	Címzett irányítószáma
recipient_country	VARCHAR(255)	NULL	Címzett országa
recipient_tax_number	VARCHAR(255)	NULL	Címzett adószáma
recipient_email	VARCHAR(255)	NULL	Címzett emailje
recipient_phone	VARCHAR(255)	NULL	Címzett telefonszáma
active	BOOLEAN	DEFAULT TRUE	Aktív-e számla?
created_at	TIMESTAMP	NULL	Létrehozás időpontja
updated_at	TIMESTAMP	NULL	Utolsó módosítás

Kapcsolatok:

- 1:N kapcsolat Booking táblával (mappedBy: invoice)
- N:M kapcsolat ServiceModel táblával (invoice_serviceModels)

Enumok:

- paymentStatus: PaymentStatusEnum (PENDING, FULFILLED, CANCELLED)

Kapcsolótáblák

A rendszer több N:M kapcsolatot kezel külön kapcsolótáblákon keresztül. Ezek közül kettőt emelek ki példaként: egy általánosat és az egyetlen speciális kapcsolótáblát.

guest_booking kapcsolótábla (általános kapcsolótábla példa) Összeköti a foglalásokat a vendégekkel. Egy foglaláshoz több vendég is tartozhat, és egy vendég több foglalással is rendelkezhet.

3.6. táblázat. guest_booking kapcsolótábla

Mező	Típus	Megszorítás	Leírás
booking_id	BIGINT	PK, FK(Booking.id)	Foglalás azonosító
guest_id	BIGINT	PK, FK(Guest.id)	Vendég azonosító

room_type_bed_type kapcsolótábla Ez a tábla speciális, mert nem csak összeköti a szobatípusokat az ágytípusokkal, hanem tárolja az adott ágytípusból hány darab található az adott szobatípusban. Pl. egy "Családi szoba" lehet 1 db franciaágy + 2 db egyszemélyes ágy.

3.7. táblázat. room_type_bed_type kapcsolótábla szerkezete

Mező	Típus	Megszorítás	Leírás
id	BIGINT	PK, AUTO_INC	Elsődleges kulcs
room_type_id	BIGINT	FK(RoomType.id), NOT NULL	Szobatípus azonosító
bed_type_id	BIGINT	FK(BedType.id), NOT NULL	Ágytípus azonosító
num_bed	INT	NOT NULL	Ágyak száma (mennyiség)

Kapcsolatok:

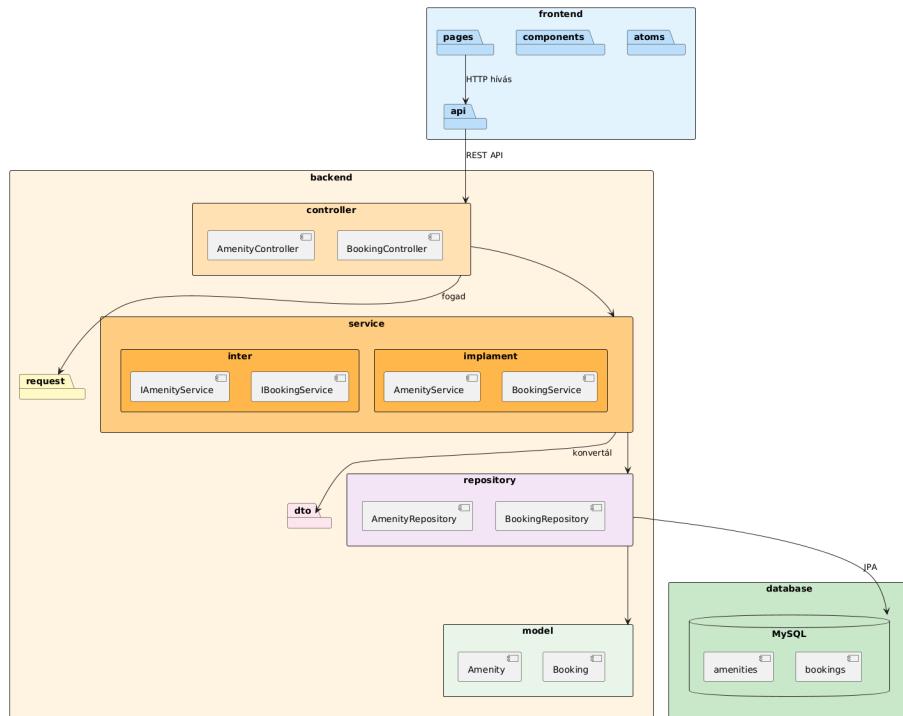
- N:1 kapcsolat RoomType táblával (room_type_id FK)
- N:1 kapcsolat BedType táblával (bed_type_id FK)

Megjegyzés: Ez a tábla saját elsődleges kulccsal rendelkezik (id), szemben az általános kapcsolótáblákkal. Ez a megoldást rugalmasabbnak találtam, mivel lehetővé teszi extra attribútumok (num_bed) tárolását a kapcsolaton.

3.5. Modul- és osztályszerkezet

3.5.1. Architektúra áttekintő diagram

A rendszer három fő rétegből épül fel, amelyek Docker konténerekben futnak:



3.3. ábra. Rendszer architektúra áttekintése

- **Frontend konténer:** React 18 + TypeScript, Vite build eszköz
- **Backend konténer:** Spring Boot 3.4, Java 17, REST API
- **Adatbázis konténer:** MySQL 8.0, perzisztens volume

3.5.2. Főbb osztályok/modulok

Backend osztályok

Controller réteg A REST API végpontokat kezelő osztályok (@RestController annotációval). Strukturailag `controller`-ben találhatóak.

- **AmenityController:** Szoba szolgáltatásokhoz kapcsolódó CRUD műveletek végpontjai
- **BookingController:** Foglalásokhoz kapcsolódó CRUD + státusz műveletek végpontjai
- **AuthController:** Felhasználó azonosítási műveletek: bejelentkezés, kijelentkezés, regisztráció végpontjai
- ... (összesen 19 controller)

Service réteg Üzleti logikát megvalósító osztályok interfésekkel. Strukturailag `service/implament/-ben` és a `service/inter/-ben` találhatóak.

- `IAmenityService / AmenityService`: Szolgáltatások üzleti logikája
- `IBookingService / BookingService`: Foglalási validáció, ütközéskezelés és egyéb üzleti logika
- `IIInvoiceService / InvoiceService`: Számlázási logika
- ... (összesen 17 service interfész + implementáció)

Repository réteg JPA Repository interfészek Spring Data JPA-val. Strukturailag `repository`/ -ben találhatóak.

- `AmenityRepository extends JpaRepository`
- `BookingRepository extends JpaRepository`
- `UserRepository extends JpaRepository`
- ... (összesen 18 repository - minden entitáshoz 1)

Model réteg (Entitások) JPA entitások az adatbázis táblák reprezentálására (lásd 3.4 fejezet). Strukturailag `model`/ -ben találhatóak.

- User, Role, Guest, GuestTag
- Booking, Room, RoomType, BedType, Amenity
- Invoice, ServiceModel, Vat
- Building, Housekeeping
- CompanyInfo, DataInitFlag, DevLog

DTO és Request/Response objektumok Adatátvitel formája a frontend-backend között. Strukturailag `dto`/ -ben, `request`/ -ben és `response`/ -ben találhatóak.

- `AmenityDto, BookingDto, GuestDto, InvoiceDto` stb.
- `CreateAmenityRequest, UpdateAmenityRequest` stb.
- `ApiResponse`: Egységes válasz formátum

Security réteg JWT autentikáció, Spring Security konfiguráció. Strukturailag `security`/ -ben találhatóak.

Frontend modulok

Pages Oldal-szintű komponensek. Strukturailag `pages`/ -ben találhatóak.

- `DashboardPage`: Főoldal statisztikákkal

- `BookingPage`: Foglalások listája és kezelése
- `RoomMirrorPage`: Szobatükör naptár nézettel
- `InvoicePage`: Számlák kezelése
- `GuestPage`: Vendégek adminisztrációja
- ... (összesen 23 oldal)

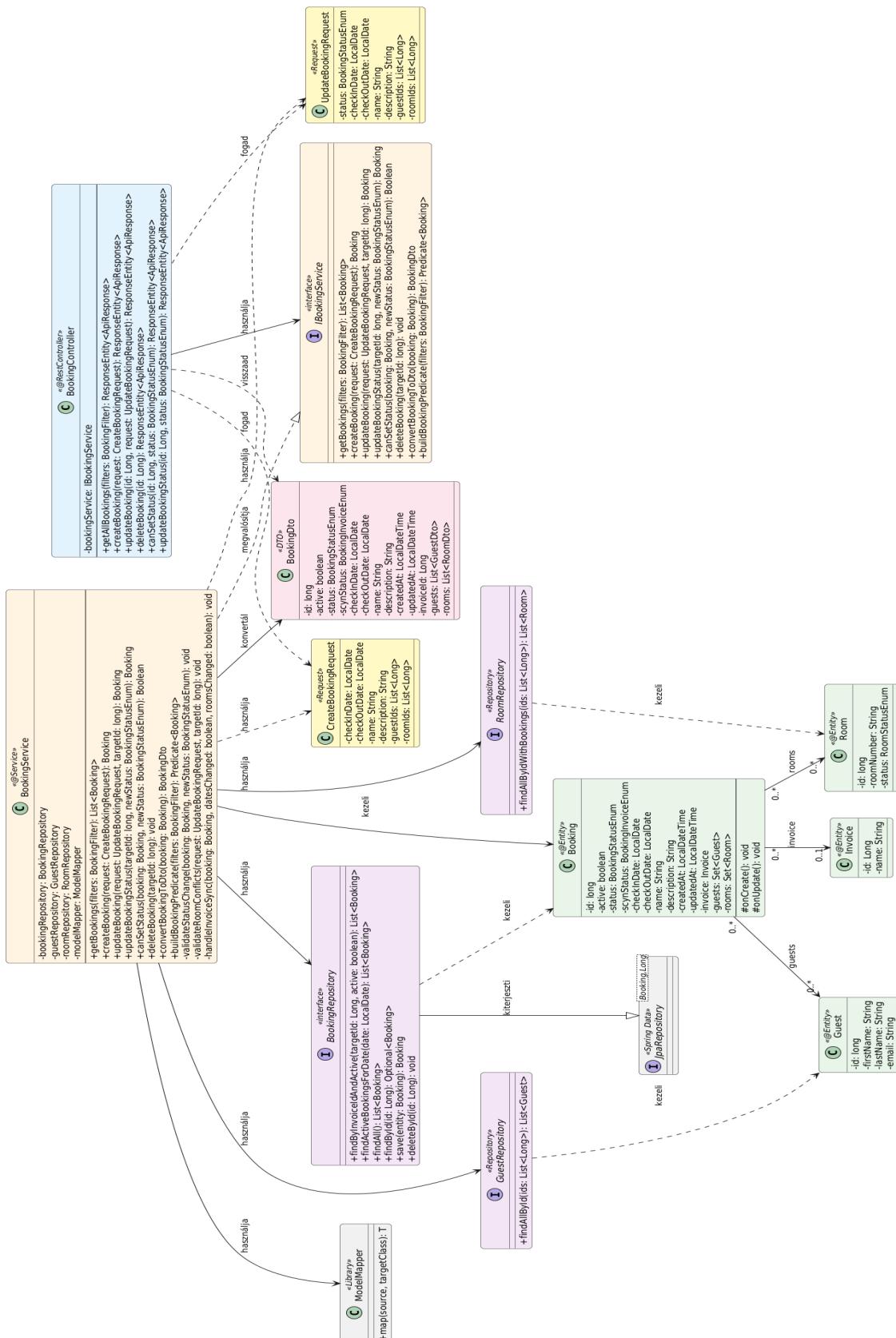
API kliens modulok Axios-alapú API kommunikáció (összesen 19 modul). Strukturailag `api/-ben` találhatóak.

- `amenityApi.ts`: Szolgáltatások API hívásai
- `bookingApi.ts`: Foglalások API hívásai
- `authApi.ts`: Autentikáció API hívásai
- ... (összesen 19 API modul)

State management Globális állapotkezelés Jotai atomokkal megoldva (összesen 30 fájl). Strukturailag `state/-ben` találhatóak.

- `user.ts`: Bejelentkezett felhasználó adatai
- `vat.ts`: Backendtől megkapott ÁFA adatok tárolása
- ... (összesen 30 állapotkezelő fájl)

3.5.3. Osztályok közötti kapcsolatok példa



3.4. ábra. Booking modul osztálydiagramja

BookingService főbb metódusai

createBooking

- **Bemenő adat:** CreateBookingRequest (checkInDate, checkOutDate, name, guestIds, roomIds)
- **Kimenő adat:** Booking entitás
- **Tevékenység:**
 - Szobák lekérdezése az adatbázisból
 - BookingConflictUtil segítségével ütközések ellenőrzése
 - Foglalás létrehozása, vendégek és szobák hozzárendelése
 - Mentés az adatbázisba

updateBookingStatus

- **Bemenő adat:** Long targetId, BookingStatusEnum newStatus
- **Kimenő adat:** Booking entitás
- **Tevékenység:**
 - canSetStatus() hívása - státusz átmenet validálása
 - CHECKED_OUT státusz esetén: szobák állapotának "DIRTY"-re állítása
 - Foglalás státuszának frissítése
 - Mentés az adatbázisba

Példa: Új foglalás létrehozása

Egy konkrét használati esetet bemutatok a rétegek közötti adatáramlásról:

1. **Frontend:** A BookingPage komponensen a felhasználó kitölti a foglalási űrlapot **MultiStepBookingModal.tsx**
2. **API hívás:** bookingApi.createBooking(data) meghívása (POST /api/bookings)
3. **Backend Controller:** BookingController.createBooking(@RequestBody CreateBookingRequest)
 - Validáció (@Valid annotáció)
 - Service réteg hívása
4. **Service réteg:** BookingService.createBooking(request)

- Üzleti logika: szobák elérhetőségének ellenőrzése
- Foglalás adatok beállítása
- Vendégek beállítása
- Repository hívás

5. **Repository:** `BookingRepository.save(booking)`

- JPA entitás mentése MySQL-be

6. **Válasz:** Entity → DTO konverzió, visszaküldés JSON-ként

7. **Frontend:** frissül a frontend oldal és visszajelz a felhasználó felé

3.5.4. UI-tervezek és navigáció

Képernyőtervezés

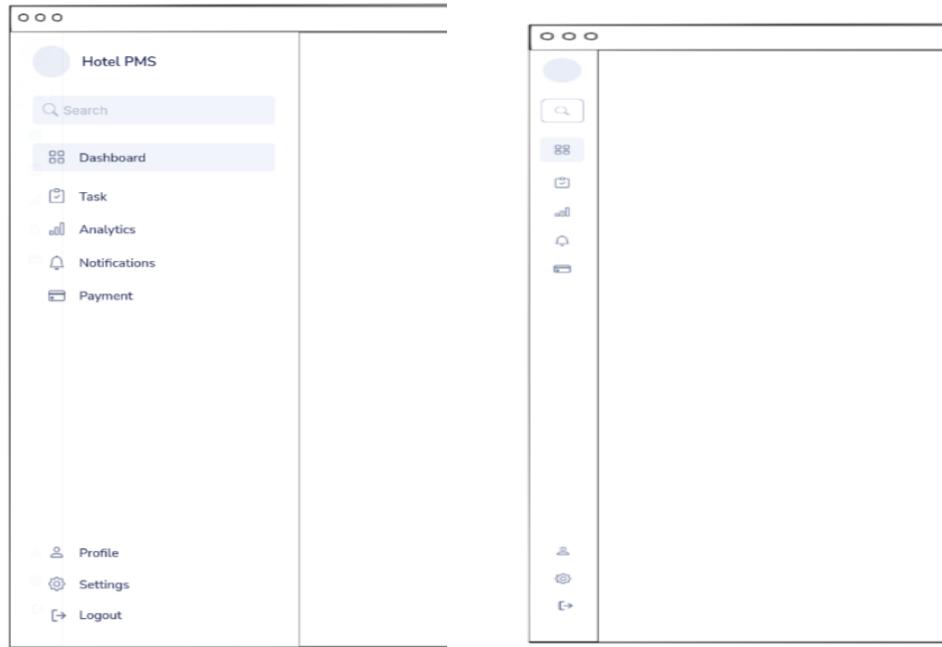
3.5.5. Felhasználói felület tervei

A rendszer webes felületet használ, Material-UI komponens könyvtárral megvalósítva. A tervezés során wireframe-eket készítettem de fontos megjegyezni, hogy ezek a kezdeti tervezek voltak és a fejlesztés során tovább fejlődtek és finomításra kerültek, de jól szemléltetik a rendszer alapvető UI architektúráját.

Navigációs struktúra

A rendszer navigációja egy összecsukható oldalsó menüsorból (sidebar) áll, amely két állapotban jelenhet meg:

- **Nyitott állapot:** teljes szöveges menüpontokkal (lásd: 3.5. (a) ábra)
- **Csukott állapot:** csak ikonokkal a képernyőterület maximalizálása érdekében (lásd: 3.5. (b) ábra)



(a) Nyitott navigációs menü

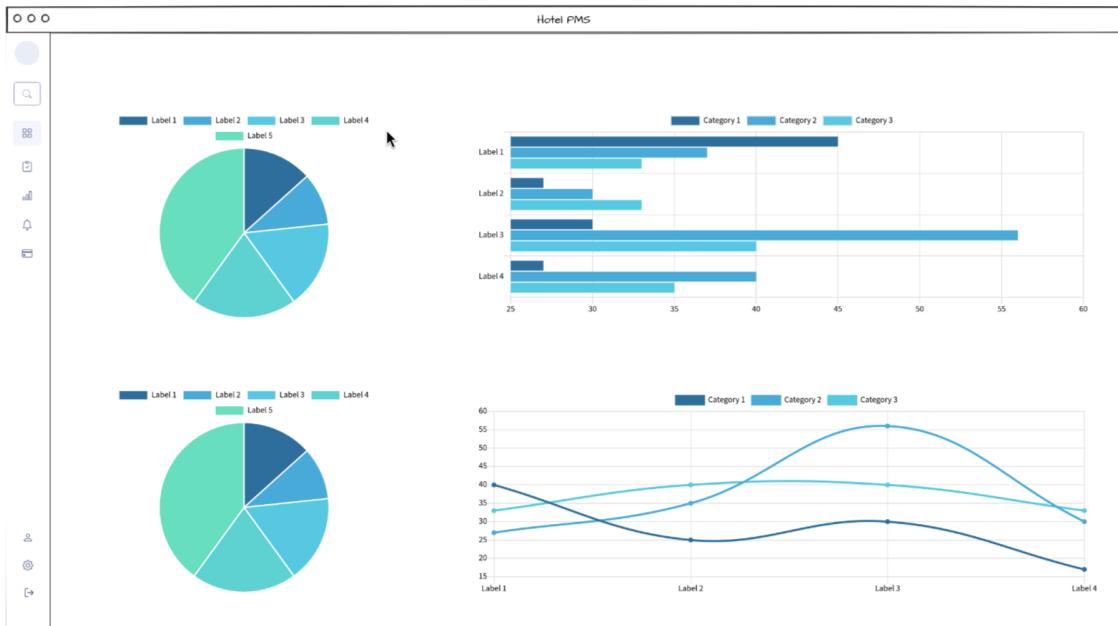
(b) Csukott navigációs menü

3.5. ábra. Navigációs menü

Kulcsfontosságú nézetek

A wireframe diagram of a login form titled "Login". The form includes fields for "Username" and "Password", a "Remember me" checkbox with a checked state, and a "Sign in" button. The entire form is contained within a large rectangular frame. At the top center of the frame, the text "Hotel PMS" is visible.

3.6. ábra. Bejelentkezési felület



3.7. ábra. Dashboard nézet statisztikákkal

Táblázatos listák Az adatok megjelenítése táblázatos formában történik (lásd: 3.8. ábra), amely biztosítja:

- Oszlopok szerinti rendezést
- Keresési és szűrési lehetőségeket
- Sor szintű műveleteket (Edit, Delete, Bonus action)
- Új rekord létrehozását ("Create new" gomb)
- Lapozást nagy adatmennyiségek esetén
- Teljes képernyős nézetet

3. Fejlesztői dokumentáció

ID	Data1	Data2	Data	Actions
1	Koch and Sons	Janice_Monahan@yahoo.com	Port Beulah, Iowa 90719, United States of America	<button>Delete</button> <button>Edit</button> <button>Bonus action</button>
2	Steuber LLC	Rollin_Fadel@gmail.com	Lake Matilde, Tennessee 74062, United States of America	<button>Delete</button> <button>Edit</button> <button>Bonus action</button>
3	Konopelski Group	Lera_Stroman3@gmail.com	Vicentaview, Mississippi 47576-9639, United States of America	<button>Delete</button> <button>Edit</button> <button>Bonus action</button>
4	Harber Inc	Adan_Schiller19@yahoo.com	VonRuedenberg, Delaware 99072-4003, United States of America	<button>Delete</button> <button>Edit</button> <button>Bonus action</button>

3.8. ábra. Táblázatos lista nézet műveleti gombokkal

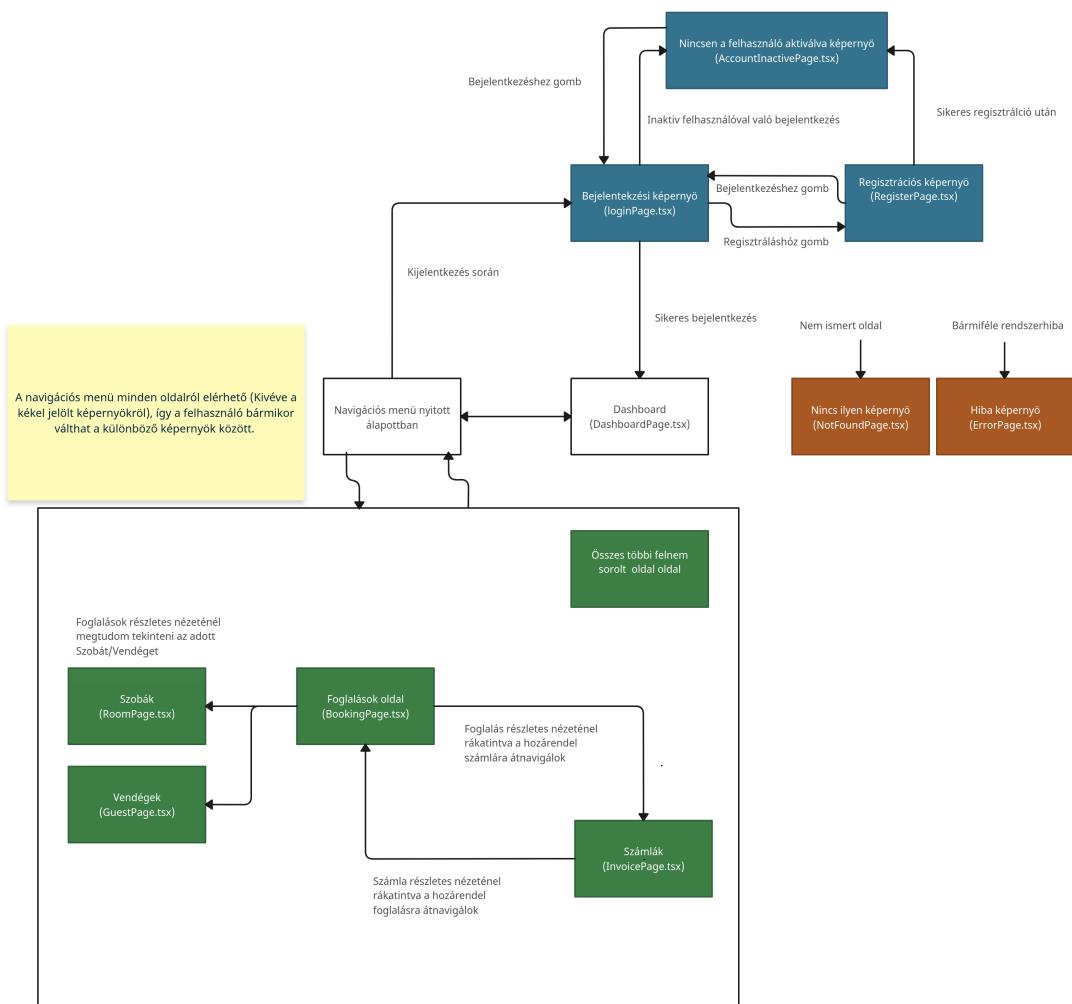
ID	Data1	Data2	Data	Actions
1	Koch and Sons	Email address	Port Beulah, Iowa 90719, United States of America	<button>Delete</button> <button>Edit</button> <button>Bonus action</button>
2	Steuber LLC	Phone	Lake Matilde, Tennessee 74062, United States of America	<button>Delete</button> <button>Edit</button> <button>Bonus action</button>
3	Konopelski Group	Data 3	Vicentaview, Mississippi 47576-9639, United States of America	<button>Delete</button> <button>Edit</button> <button>Bonus action</button>
4	Harber Inc	Adan_Schiller19@yahoo.com	VonRuedenberg, Delaware 99072-4003, United States of America	<button>Delete</button> <button>Edit</button> <button>Bonus action</button>

3.9. ábra. Modal ablak

Select building	01.01	01.02	0.03	01.04	01.05	01.06	01.07	01.08
101			Booking 1			Booking 3		
102								
103								
104						Booking 2		
105			Booking 1					

3.10. ábra. Szobatükör nézet foglalások vizualizálásához

3.5.6. Navigációs diagram



3.11. ábra. Navigációs diagram

3.6. Kiemelndö kód részletek

Itt bemutatom a rendszer implementációjának fontosabb részeit, amelyek a legnagyobb kihívást jelentették a fejlesztés során, vagy amelyek egyedi megoldásokat igényeltek.

A bemutatott kód részletek nem tartalmazzák a teljes forráskódot hanem csak a lényegi logikát emelik ki, hogy a dokumentáció olvasható maradjón.

A fejezet két fő részre tagolódik:

- **Közös algoritmusok:** Olyan hasonló logikai egységek, amelyeket a rendszer több pontján is használtam.
- **Nem triviális megoldások:** Összetett implementációs megoldások, amelyek speciális tervezési döntéseket, külső könyvtárak integrációját vagy nem magától értetődő programozási technikákat igényeltek.

3.6.1. Közös algoritmusok

CRUD műveletek általános mintája

A rendszer backend moduljai következetes szerkezetet követnek a CRUD (Create, Read, Update, Delete) műveletek implementálásánál.

Általános felépítés: minden CRUD modul az alábbi komponensekből áll:

- **Controller:** REST API végpontok (@RestController)
- **Service interfész + implementáció:** Üzleti logika
- **Repository:** JPA adatbázis elérés
- **Request/Response DTO-k:** Adatátviteli objektumok

Példa: AmenityController CRUD műveletek A AmenityController-el szemléltetem az általános CRUD eljárásokat

1. List (Összes elem lekérdezése):

```

1  @GetMapping
2  public ResponseEntity<ApiResponse> getAllAmenities(
3      @ModelAttribute AmenityFilter filters) {
4
5      try {
6
7          List<Amenity> amenities = amenityService.
8              getAmenities(filters);
9
10         List<AmenityDto> amenityDtos = amenities.stream()
11             .map(amenityService::convertAmenityToDto)
12             .toList();
13
14
15     return ResponseEntity
16         .ok(new ApiResponse(FrontEndCodes.SUCCESS.getCode
17             () , amenityDtos));
18
19
20 } catch //error handling...
21 }
22
23 // Service reteg
24
25 @Transactional(readOnly = true)
26 @Override //Hiszen van egy interface
27 public List<Amenity> getAmenities(AmenityFilter filters
28 ) {
29
30     if (filters == null) {
31
32         return amenityRepository.findAll();
33     }
34
35
36     return amenityRepository.findAll().stream()
37         .filter(buildAmenityPredicate(filters))
38         .collect(Collectors.toList());
39
40 }
```

3.1. forráskód. GET /api/amenities - Lista lekérdezés

2. Create (Új elem letrehozása):

```

1  @PostMapping
2  public ResponseEntity<ApiResponse> createAmenity(@Valid
3      @RequestBody CreateAmenityRequest request) {
4
5      try {
6
7          Amenity createdAmenity = amenityService.
8              createAmenity(request);
9
10         AmenityDto amenityDto = amenityService.
11             convertAmenityToDto(createdAmenity);
12
13
14     return ResponseEntity.status(HttpStatus.CREATED)
15         .body(new ApiResponse(FrontEndCodes.SUCCESS.getCode(),
16             amenityDto));
17
18
19 } catch //error handling...
20
21
22 // Service reteg
23
24 @Override
25 public Amenity createAmenity(CreateAmenityRequest
26     request) {
27
28     Optional<Amenity> existingAmenityOpt =
29         amenityRepository.findByAmenityName(request.
30             getAmenityName());
31
32
33     if (existingAmenityOpt.isPresent()) {
34
35         throw new AlreadyExistsException(FrontEndCodes.
36             AMENITY_ALREADY_EXISTS.getCode());
37     }
38
39
40     Amenity amenity = new Amenity();
41     amenity.setAmenityName(request.getAmenityName());
42
43
44     return amenityRepository.save(amenity);

```

25 }

3.2. forráskód. POST /api/amenities - Uj elem letrehozása

3. Update (Elem módosítása):

```

1   @PutMapping("/{id}")
2   public ResponseEntity<ApiResponse> updateAmenity(
3       @PathVariable Long id,
4       @Valid @RequestBody UpdateAmenityRequest request) {
5       try {
6           Amenity updatedAmenity = amenityService.
7               updateAmenity(request, id);
8           AmenityDto amenityDto = amenityService.
9               convertAmenityToDto(updatedAmenity);
10
11          return ResponseEntity.status(HttpStatus.CREATED)
12              .body(new ApiResponse(FrontEndCodes.SUCCESS.getCode(),
13                  amenityDto));
14
15      } catch //error handling...
16  }
17
18 // Service reteg
19
20 @Override
21 public Amenity updateAmenity(UpdateAmenityRequest
22     request, long targetId) {
23     Optional<Amenity> existingAmenityOpt =
24         amenityRepository.findById(targetId);
25     if (existingAmenityOpt.isEmpty()) {
26         throw new ResourceNotFoundException(FrontEndCodes.
27             AMENITY_NOT_FOUND.getCode());
28     }
29     Amenity existingAmenity = existingAmenityOpt.get();

```

```
21     Optional<Amenity> existingName = amenityRepository.  
22         findByAmenityName(request.getAmenityName());  
23     if (existingName.isPresent() && existingName.get().  
24         getId() != targetId)  
25         throw new AlreadyExistsException(FrontEndCodes.  
26             AMENITY_ALREADY_EXISTS.getCode());  
27     existingAmenity.setAmenityName(request.getAmenityName()  
28 );  
29     return amenityRepository.save(existingAmenity);  
30 }
```

3.3. forráskód. PUT /api/amenities/{id} - Elem módosítása

4. Delete (Elem törlése):

```

1   @DeleteMapping("/{id}")
2   public ResponseEntity<ApiResponse> deleteAmenity(
3       @PathVariable Long id) {
4       try {
5           amenityService.deleteAmenity(id);
6           return ResponseEntity.ok(new ApiResponse(
7               FrontEndCodes.SUCCESS.getCode(), null));
8       } catch //error handling...
9   }
10
11
12 // Service reteg
13
14 @Override
15 public void deleteAmenity(long targetId) {
16     amenityRepository.findById(targetId).ifPresentOrElse(
17         amenity -> {
18             if (amenity.getRoomTypes().isEmpty()) {
19                 amenityRepository.deleteById(targetId);
20             } else {
21                 throw new IllegalStateException(FrontEndCodes.
22                     AMENITY_HAS_ROOMS.getCode());
23             }
24         },
25         () -> {
26             throw new ResourceNotFoundException(FrontEndCodes.
27                 AMENITY_NOT_FOUND.getCode());
28         }
29     );
30 }
```

3.4. forráskód. DELETE /api/amenities/{id} - Elem törlése

Eltérések speciális modulokban: Vannak olyan modulok, amelyek **extra üzleti logikát** tartalmaznak a standard CRUD műveleteken túl pl.: BookingService: Ütközéskezelés, státusz validáció, szobák foglalása Ezek a modulok azonban **ugyanazt az alapstruktúrát használják**, csak extra metódusokkal bővítik ki.

Request és DTO

- **Request objektumok:** Bejövő adatok (pl. CreateXxxRequest)
- **DTO objektumok:** Kimenő adatok a frontend felé (XxxDto)

CreateAmenity példa

1. **Frontend:** POST /api/amenities, body: {"name": "WiFi"}
2. **Controller:** CreateAmenityRequest deserializálása JSON-ból
3. **Validáció:** @Valid annotáció ellenőrzi a mezőket
4. **Service:** createAmenity üzleti logika
5. **Repository:** Amenity mentése adatbázisba
6. **Service:** Amenity → AmenityDto (convertAmenityToDto)
7. **Controller:** AmenityDto JSON-ná alakítása
8. **Frontend:** JSON válasz fogadása

Request Request objektumokban Bean Validation annotációk használtam, egyszerű szerver oldali ellenőrzés érdekében (@NotNull, @Email, @Size stb.)

```

1  @Data
2  public class CreateRoomRequest {
3      @NotNull
4      private RoomStatusEnum status;
5
6      private String description;
7
8      @NotBlank
9      private String roomNumber;
10
11     @NotNull
12     @Min(1)
13     private Integer floorNumber;

```

```
14  
15     @NotNull  
16     private Long roomTypeId;  
17  
18     @NotNull  
19     private Long buildingId;  
20 }
```

3.5. forráskód. Request validálció Bean Validation-al

DTO

```
1 @Override  
2 @Transactional(readOnly = true)  
3 public AmenityDto convertAmenityToDto(Amenity amenity)  
4 {  
5     return modelMapper.map(amenity, AmenityDto.class);  
6 }
```

3.6. forráskód. Egyszerü DTO konverzió

```
1 @Override  
2 @Transactional(readOnly = true)  
3 public GuestDto convertGuestToDto(Guest guest) {  
4     GuestDto guestDto = modelMapper.map(guest, GuestDto.  
5                                         class);  
6  
7     List<GuestTagDto> activeGuestTagDtos = guest.  
8         getGuestTags().stream()  
9         .filter(GuestTag::getActive)  
10        .map(guestTag -> modelMapper.map(guestTag,  
11              GuestTagDto.class))  
12        .collect(Collectors.toList());  
13  
14     guestDto.setGuestTags(activeGuestTagDtos);  
15 }
```

```
12     return guestDto;  
13 }
```

3.7. forráskód. Komplex DTO konverzió ha több entitást is vissza kell adni

Backend szűrés és lapozás - alternatív megközelítés

A rendszer fejlesztése során fontolóra vettet a backend oldali szűrés és lapozás implementálását. Végül azonban ezt a megoldást nem alkalmaztam, mert az adatmennyiség nem indokolta.

Miért NEM lett implementálva?

- **Adatmennyiség:** A szálloda mérete kisebb vagy közepes max, így a teljes adathalmaz betöltése nem vezett teljesítményproblémához
- **Frontend képességek:** A Material React Table komponens és gyors kliens oldali szűrést és rendezést biztosít

Előkészített backend szűrési infrastruktúra: Ennek ellenére a rendszer **továbbfejleszthetőség** céljából tartalmazza a backend oldali szűrés kódját. A legtöbb Service osztályban megtalálható a szűrési logika, amely azonnal használható, ha a frontend implementálja a szűrési paraméterek küldését.

```
1 @Data  
2 public class BuildingFilter {  
3     private Long id;  
4     private String name;  
5     private String address;  
6     private String city;  
7     private String zipcode;  
8     private String country;  
9     private Boolean active;  
10    private List<Long> userIds;  
11 }
```

3.8. forráskód. Épületek szűrés kérés

```
1  @Transactional(readOnly = true)
2  @Override
3  public List<Building> getBuildings(BuildingFilter filters
4      ) {
5      if (filters == null) {
6          return buildingRepository.findAll();
7      }
8
9      return buildingRepository.findAll().stream()
10     .filter(buildBuildingPredicate(filters))
11     .collect(Collectors.toList());
12
13
14  @Override
15  public Predicate<Building> buildBuildingPredicate(
16      BuildingFilter filters) {
17      Predicate<Building> predicate = building -> true;
18
19      if (filters.getId() != null) {
20          predicate = predicate.and(building -> building.getId()
21              .equals(filters.getId()));
22      }
23
24      if (filters.getName() != null && !filters.getName().
25          isEmpty()) {
26          predicate = predicate.and(building ->
27              building.getName().toLowerCase().contains(filters.
28                  getName().toLowerCase()));
29      }
30
31      ...
32
33      if (filters.getUserIds() != null && !filters.getUserIds()
34          .isEmpty()) {
35          predicate = predicate.and(building -> building.
36              getUsers().stream()
```

```

27     .anyMatch(user -> filters.getUserIds().contains(user.
28         getId())));
29
30     return predicate;
31 }

```

3.9. forráskód. Szűrés függvény a BuildingService-ben

3.6.2. Nem triviális megoldások

Dinamikus űrlap léterhozás - FormFactory

A frontend fejlesztése során kihívást okozott a sok hasonló űrlap (modal ablakok) kezelése. Ahelyett, hogy minden entitáshoz (Amenity, Guest, Room, stb.) külön komponenst írtam volna, egy általános űrlap generátort hoztam létre, amely egy alap konfigurációból dinamikusan építi fel az űrlapokat.

Probléma: A rendszerben 15+ entitás van, mindegyiknek van Create és Update művelete, ami 30+ különböző űrlapot jelentene. Ez rengeteg kód duplikációhoz vezetne:

- Validációs logika ismétlődik
- Hibakezelés minden űrlapban azonos
- Stílus és elrendezés következetlensége
- Nehéz karbantarthatóság

Megoldás: FormFactory + FormConfig A FormFactory komponens egy konfigurációból generálja az űrlap mezőket automatikusan.

FormConfig típusdefiníció:

```

1  export interface FieldConfig {
2      label: string;                      // Mezo cimke
3      type: FieldType;                    // Mezo tipusa
4      defaultValue?: FieldValue;          // Alapertelmezett ertekek
5      validation?: ValidationRules;       // Validacios szabalyok
6      options?: FieldOption[];           // Select/Autocomplete
7          opciok
8      placeholder?: string;
9      helperText?: string;
10     disabled?: boolean;
11     multiple?: boolean;                // Tobbszelekciós field
12 }
13
14
15
16
17 export type FieldType =
18 | "text"
19 | "number"
20 | "autocomplete"
21 | "select"
22 | "checkbox"
23 | "datepicker"
24 | "bedTypeQuantity";

```

3.10. forráskód. FormConfig TypeScript típusok

ConverterFactory működése:

Először létrehozok egy `FormConfig` típusú űrlap konfigurációt, amely leírja az összes mezőt és tulajdonságait. A `ConverterFactory` végigmegy ezen a konfigurációs objektumon, minden `FieldConfig`-hoz generál egy megfelelő input mezőt, majd ezeket betölti egy egységes stílusú `FormModal` komponensbe.

```

1 export const FormFactory = ({...}: FormFactoryProps) => {
2   return (
3     <>
4       {Object.keys(config).map((fieldName) => {
5         const fieldConfig = config[fieldName];
6         const initialValue =
7           initialValues[fieldName] ??
8             fieldConfig.defaultValue ??
9               getDefaultValueForType(fieldConfig);
10
11         return (
12           <FieldWrapper
13             key={fieldName}
14             fieldName={fieldName}
15             fieldConfig={fieldConfig}
16             initialValue={initialValue}
17             valuesRef={valuesRef}
18             error={errors[fieldName]}
19           />
20         );
21       })} }
22     </>
23   );
24 };

```

3.11. forráskód. FormFactory komponens

A FieldWrapper komponens a type alapján dönti el, hogy melyik konkrét beviteli mező komponenst használja:

```

1  const FieldWrapper = ({ fieldName, fieldConfig, ... }) {
2      => {
3          const [value, setValue] = useState(initialValue);
4          const handleChange = (newValue) => {
5              setValue(newValue);
6              valuesRef.current[fieldName] = newValue;
7          };
8          switch (fieldConfig.type) {
9              case "text":
10                 case "number": return <TextField {...fieldProps}>;
11                 case "autocomplete": return <AutocompleteField {...fieldProps}>;
12                 case "select": return <SelectField {...fieldProps}>;
13                 case "checkbox": return <CheckboxField {...fieldProps}>;
14                 case "datepicker": return <DatePickerField {...fieldProps}>;
15                 case "bedTypeQuantity": return <BedTypeQuantityField {...fieldProps}>;
16                 default: return null;
17             }
18         };

```

3.12. forráskód. FieldWrapper - dinamikus beviteli mező választás

Példa: Vendég címke létrehozása modal:

```
1  export const guestTagFormConfig: FormConfig = {
2
3    tagName: {
4
5      label: "forms.labels.tagName",
6
7      type: "text",
8
9      defaultValue: "",
10
11     validation: {
12
13       required: true,
14
15       minLength: 2,
16
17     },
18
19   },
20
21   active: {
22
23     label: "forms.labels.active",
24
25     type: "checkbox",
26
27     defaultValue: true,
28
29     validation: {
30
31       required: false,
32
33     },
34
35   },
36
37 };
38
39
40
41  const handleClick = () => {
42
43   setTitle(t("guestTag.uploadTitle"));
44
45   setConfig(guestTagFormConfig);
46
47   setInitialValues({});
48
49   setOnSubmit(() => handleSubmit as (values: FormValues
50
51     ) => Promise<void>);
52
53   setFormModalOpen(true);
54
55 };

56
57 
```

3.13. forráskód. Vendég címke űrlap konfiguráció

FormModal komponens: A FormModal egy egységes modal ablak, amely a FormFactory-t használja és reaktívan kezeli állapotokat. Jotai atomokon keresztül

```

29         valuesRef.current = { ...initialValues } as
30             FormValues;
31         setErrors({}) ;
32     }
33   }, [open, initialValues]);
34
35
36   if (!config) return null;
37
38   return (
39     <Dialog open={open}>
40       <Typography sx={{ fontWeight: "bold", padding: 2 }}>
41         variant="h5">
42           {title}
43         </Typography>
44       <Box sx={{ padding: 2 }}>
45         <FormFactory
46           config={config}
47           initialValues={initialValues}
48           valuesRef={valuesRef}
49           errors={errors}
50         />
51       </Box>
52       <DialogActions>
53         <Button onClick={handleClose} label="Cancel" />
54         <ButtonUI label="Save" onClick={handleSave}/>
55       </DialogActions>
56     </Dialog>
57   );
58 };

```

3.14. forráskód. FormModal komponens (Egyszerűsített)

Automatikus validáció: A validateForm függvény a konfiguráció alapján automatikusan ellenőrzi a mezőket ha menteni szeretném az űrlapot:

```

1 export const validateForm = (
2   values: FormValues,
3   config: { [fieldName: string]: FieldConfig },
4 ): FormErrors => {
5   const errors: FormErrors = {};
6
7   Object.keys(config).forEach((fieldName) => {
8     const fieldConfig = config[fieldName];
9     const value = values[fieldName];
10    const error = validateField(value, fieldConfig.
11      validation);
12
13    if (error) {
14      errors[fieldName] = error;
15    }
16
17  return errors;
18};

```

3.15. forráskód. Validációs rendszer

Előnyök:

- **Típusbiztonság:** TypeScript típusok garantálják a helyes konfigurációt
- **Következetesség:** minden űrlap azonos megjelenésű és működésű
- **Gyors fejlesztés:** új entitás űrlap 10-15 sor JSON konfigurációval elkészül
- **Könnyű módosítás:** Egy helyen kell módosítani a validációs/styling logikát

Speciális mezőtípus: BedTypeQuantity Ez egy egyedi field típus a bedTypeQuantity, amely a szobatípus-ágytípus kapcsolatot kezeli mennyiséggel:

```

1 bedTypes: {
2   label: "Bed Configuration",
3   type: "bedTypeQuantity",
4   defaultValue: [],

```

```

5     validation: { required: true }
6   }
7   // [{ bedTypeId: 1, quantity: 1 }, { bedTypeId: 2,
8     quantity: 2 }]

```

3.16. forráskód. BedTypeQuantity field konfiguráció

Ez a megoldás jól szemlélteti a FormFactory rugalmasságát: könnyen bővíthető új field típusokkal anélkül, hogy a központi logikát sérteném meg.

Dinamikus konfiguráció - backend adatokkal: A FormFactory másik előnye, hogy a konfigurációt futásidőben is módosíthatom backend adatok alapján. Például a `ServiceModel` űrlapnál a ÁFA (VAT) kulcsok listáját nem beleégetem a konfigurációba (Hiszen ezek változhatnak), hanem futásidőben lekérem a backendről és dinamikusan illesztem be.

```

1
2   export const serviceFormConfig: FormConfig = {
3     ...
4     vatId: {
5       label: "forms.labels.vat",
6       type: "select",
7       defaultValue: "",
8       validation: {
9         required: true,
10      },
11       options: [] ,
12     },
13   };
14
15   const [vats, setVats] = useState<{ value: number; label
16     : string }[]>([]);
17
18   useEffect(() => {
19     const loadVats = async () => {

```

```

20     setVats(
21       vatsData.map((vat) => ({
22         value: vat.id,
23         label: `${vat.name} (${vat.percentage}%)`,
24       })),
25     );
26   };
27   loadVats();
28 }, []);
29
30
31 const handleClick = () => {
32   setTitle(t("service.uploadTitle"));
33   const configWithVats = {
34     ...serviceFormConfig,
35     vatId: {
36       ...serviceFormConfig.vatId,
37       options: vats,
38     },
39   };
40   setConfig(configWithVats);
41   setInitialValues({ cost: 0 });
42   setOnSubmit(() => handleSubmit as (values: FormValues
43     ) => Promise<void>);
44   setFormModalOpen(true);
45 };

```

3.17. forráskód. Dinamikus form konfiguráció VAT opciókkal

Így az űrlap minden az aktuális adatbázisban tárolt ÁFA kulcsokat jeleníti meg, anélkül hogy a kódot módosítani kellene. Ugyanez a megoldás használható más entitásoknál is pl. a Room űrlapnál és RoomType űrlapnál

Virtuális szolgáltatások - szobák összesített költsége

A számlázási rendszer egyik speciális megoldása a **virtuális szolgáltatások** kezelése, amelyek csak a számlákon belül létező szolgálatások, és a szolgáltatások képernyön nem jelennek meg.

Probléma: Amikor egy számlához foglalást kapcsolunk, a szobák árát napok szerint kell kiszámítani és a számlához rendelni. Azonban nem szerencsés minden egyes foglalás szobáit külön-külön szolgáltatásként kezelní, mert:

- A szolgáltatások listája tele lenne ugyanolyan nevű "Szoba költség" tételekkel
- Nehéz lenne megkülönböztetni őket
- Zavaró lenne a frontend számára

Megoldás: Virtuális ServiceModel A rendszer egy **base ServiceModel-t** használ ("Rooms aggregated cost", id=1), amely sablonként szolgál. Ebből virtuális példányokat hoz létre minden számlához (foglalásonként):

```

1  @Override
2  public ServiceModel createVirtualServiceModel(
3      List<Room> rooms,
4      LocalDate checkInDate,
5      LocalDate checkOutDate
6  ) {
7      ServiceModel baseServiceModel =
8          getOrCreateRoomsCostServiceModel();
9
10     double totalCost = 0.0;
11
12     for (Room room : rooms) {
13         long days = checkOutDate.toEpochDay() - checkInDate
14             .toEpochDay();
15         totalCost += room.getRoomType().getPrice() * days;
16     }
17
18     ServiceModel virtualServiceModel = new ServiceModel()
19         ;

```

```

16     virtualServiceModel.setName(baseServiceModel.getName()
17         ());
18     virtualServiceModel.setDescription(baseServiceModel.
19         getDescription());
20     virtualServiceModel.setCost(totalCost);
21     virtualServiceModel.setVirtual(true);
22     virtualServiceModel.setImmutable(false);
23     virtualServiceModel.setVat(baseServiceModel.getVat())
24         ;
25
26     return serviceModelRepository.save(
27         virtualServiceModel);
28 }
```

3.18. forráskód. Virtuális ServiceModel létrehozása

Base ServiceModel garantálása: A `getOrCreateRoomsCostServiceModel()` metódus biztosítja, hogy mindenkor létezzen a sablon ServiceModel:

```

1  @Override
2  public ServiceModel getOrCreateRoomsCostServiceModel()
3  {
4      Optional<ServiceModel> existingServiceModel =
5          serviceModelRepository.findById(1L);
6
7      if (existingServiceModel.isPresent() &&
8          "Rooms aggregated cost".equals(existingServiceModel.
9              get().getName())) {
10         ServiceModel serviceModel = existingServiceModel.
11             get();
12
13         if (serviceModel.getImmutable() == null ||
14             !serviceModel.getImmutable()) {
15             serviceModel.setImmutable(true);
16         }
17     }
18
19     return serviceModel;
20 }
```

```
13         serviceModel = serviceModelRepository.save(
14             serviceModel);
15
16     return serviceModel;
17 }
18
19 ServiceModel roomsCostServiceModel = new ServiceModel
20     ();
21 roomsCostServiceModel.setName("Rooms aggregated cost"
22     );
23 roomsCostServiceModel.setDescription(
24     "Base service model for room costs in invoices"
25     );
26 roomsCostServiceModel.setCost(0.0);
27 roomsCostServiceModel.setVirtual(false);
28 roomsCostServiceModel.setImmutable(true);
29
30 Vat defaultVat = vatRepository.findAll().stream().
31     findFirst()
32     .orElseThrow(() -> new IllegalStateException(
33         "No VAT records found in database"
34     ));
35 roomsCostServiceModel.setVat(defaultVat);
36
37 return serviceModelRepository.save(
38     roomsCostServiceModel);
39 }
```

3.19. forráskód. Base ServiceModel létrehozása/lekérése

Virtuális szolgáltatások szűrése: A `getServiceModels()` metódus automatikusan kiszűri a virtuális elemeket, így a frontend szolgálatások képernyőn nem jelennek meg.

```

1  @Transactional(readOnly = true)
2  @Override
3  public List<ServiceModel> getServiceModels(
4      ServiceModelFilter filters) {
5
6      if (filters == null) {
7
8          return serviceModelRepository.findAll().stream()
9              .filter(serviceModel -> !serviceModel.getVirtual())
10             .collect(Collectors.toList());
11
12     }
13
14 }
```

3.20. forráskód. Virtuális szolgáltatások szűrése

Miért fontos ez a megoldás?

- **Frontend számára:** A ServiceModel lista tiszta, csak a valódi szolgáltatásokat tartalmazza (SPA, minibar, parkolás stb.)
- **Számlán belül:** A szobák költsége egy összesített téteként jelenik meg ("Rooms aggregated cost: 45000 HUF")
- **Immutable flag:** A base ServiceModel nem törlhető.

Virtuális szolgáltatás módosítása: A virtuális szolgáltatások költsége és ÁFA kulcsa utólag is módosítható:

```

1  @Override
2  public ServiceModel updateVirtualServiceModel(
3      long targetId,
4      Double cost,
5      Long vatId
6  ) {
7      Optional<ServiceModel> existingServiceModelOpt =
8 }
```

```
8     serviceModelRepository.findById(targetId);
9
10    if (existingServiceModelOpt.isEmpty()) {
11
12        throw new ResourceNotFoundException(
13            FrontEndCodes.SERVICE_MODEL_NOT_FOUND.getCode()
14        );
15
16    }
17
18    ServiceModel existingServiceModel =
19        existingServiceModelOpt.get();
20
21    if (!existingServiceModel.getVirtual()) {
22
23        throw new IllegalStateException(
24            FrontEndCodes.SERVICE_MODEL_NOT_FOUND.getCode()
25        );
26
27    }
28
29    if (cost != null) {
30
31        existingServiceModel.setCost(cost);
32
33    }
34
35    if (vatId != null) {
36
37        Optional<Vat> vatOpt = vatRepository.findById(vatId
38
39        );
40
41        if (vatOpt.isEmpty()) {
42
43            throw new ResourceNotFoundException(
44                FrontEndCodes.SERVICE_MODEL_VAT_NOT_FOUND.getCode()
45            )
46        );
47
48    }
49
50    existingServiceModel.setVat(vatOpt.get());
51
52
53    return serviceModelRepository.save(
54        existingServiceModel);
55}
```

3.21. forráskód. Virtuális ServiceModel módosítása

3.7. Telepítési utmutató

A rendszer Docker konténerekben fut, így egyszerü telepíteni és konzisztens futási környezetet különböző platformokon is. A telepítéshez csak a Docker és Docker Compose szükséges.

3.7.1. Rendszerkövetelmények

Szoftver követelmények:

- **Docker Engine:** 20.10 vagy újabb
- **Docker Compose:** 2.0 vagy újabb
- **Minimum 4 GB RAM** a konténerek futtatásához
- **Minimum 5 GB szabad lemezterület**

Támogatott operációs rendszerek:

- Linux (Ubuntu 20.04+ volt csak tesztelve)
- macOS 11+ (Intel és Apple Silicon)
- Windows 10/11 (WSL2-vel)

3.7.2. Docker telepítése

- Linux (Ubuntu): docs.docker.com/engine/install/ubuntu/
- MacOS / Windows: www.docker.com/products/docker-desktop

3.7.3. Projekt letöltése és indítása

1. Repository klónozása:

```
git clone https://github.com/CsBenedek32/HOTEL-PMS
```

3.22. forráskód. Projekt letöltése

2. Konténerek építése és indítása:

```
1 docker compose up --build
```

3.23. forráskód. Docker Compose indítása

Ez a parancs:

- Felépíti a frontend és backend Docker image-eket
- Letölti a MySQL 8.0 image-et
- Elindítja minden konténert
- Létrehozza a `hotelpms` adatbázist

3. Háttérben futtatás (opcionális): Ha a konténereket háttérben szeretném futtatni:

```
1 docker compose up -d
```

3.24. forráskód. Docker Compose háttérben

Elérhetőség ellenőrzése

A konténerek sikeres indítása után a következő portokat elkövethetem:

Szolgáltatás	URL	Port
Frontend	http://localhost:3000	3000
Backend API	http://localhost:8080/api	8080
Swagger UI	http://localhost:8080/swagger	8080
MySQL adatbázis	localhost:3306	3306

3.8. táblázat. Szolgáltatások elérhetősége

3.7.4. Docker konfiguráció

Docker Compose konfiguráció

A `docker-compose.yml` fájl három szolgáltatást definiál:

```
1 services:
2   frontend:
3     build: ./frontend
4     ports:
5       - "3000:3000"
6     depends_on:
7       - backend
8     restart: on-failure
9
```

```
10  backend:
11    build: ./backend
12    ports:
13      - "8080:8080"
14    environment:
15      - SPRING_DATASOURCE_URL=jdbc:mysql://mysql:3306/
16        hotelpms
17      - SPRING_DATASOURCE_USERNAME=root
18      - SPRING_DATASOURCE_PASSWORD=rootpass123
19      - SPRING_JPA_HIBERNATE_DDL_AUTO=update
20
21 depends_on:
22   mysql:
23     condition: service_healthy
24     restart: on-failure
25
26 mysql:
27   image: mysql:8.0
28   environment:
29     - MYSQL_ROOT_PASSWORD=rootpass123
30     - MYSQL_DATABASE=hotelpms
31   command: --authentication-policy=
32     mysql_native_password --host-cache-size=0
33   ports:
34     - "3306:3306"
35   volumes:
36     - mysql_data:/var/lib/mysql
37   healthcheck:
38     test: ["CMD", "mysqladmin", "ping", "-h", "
39       localhost", "-u", "root", "-prootpass123"]
40     timeout: 20s
41     retries: 10
42     interval: 10s
43     start_period: 40s
```

```
41    volumes:  
42      mysql_data:
```

3.25. forráskód. docker-compose.yml struktúra

Frontend Dockerfile

A frontend konténer Node.js 22 Alpine image-et használ:

```
1  FROM node:22-alpine  
2  
3  WORKDIR /app  
4  
5  COPY package.json .  
6  
7  RUN npm install  
8  
9  RUN npm i -g serve  
10  
11 COPY . .  
12  
13 RUN npm run build  
14  
15 EXPOSE 3000  
16  
17 CMD [ "serve", "-s", "dist" ]
```

3.26. forráskód. Frontend Dockerfile

Build lépések:

1. Node.js 22 Alpine base image
2. Függőségek telepítése (`npm install`)
3. Serve telepítése (statikus fájl kiszolgáláshoz)
4. Forráskód másolása
5. Vite production build (`npm run build`)
6. 3000-es port expose-olása

-
7. Serve indítása a `dist` mappából

Backend Dockerfile

A backend konténer Eclipse Temurin JDK 17 image-et használ:

```

1  FROM eclipse-temurin:17-jdk-alpine
2
3  WORKDIR /app
4
5  COPY mvnw .
6  COPY mvnw.cmd .
7  COPY .mvn .mvn
8  COPY pom.xml .
9
10 RUN chmod +x ./mvnw
11
12 RUN ./mvnw dependency:go-offline -B
13
14 COPY src src
15
16 RUN ./mvnw clean package -DskipTests
17
18 EXPOSE 8080
19
20 CMD ["java", "-jar", "target/backend-0.0.1-SNAPSHOT.jar"]

```

3.27. forráskód. Backend Dockerfile

Build lépések:

1. Eclipse Temurin JDK 17 Alpine base image
2. Maven wrapper fájlok másolása
3. Függőségek előzetes letöltése (build gyorsítás)
4. Forráskód másolása
5. Maven clean package (tesztek kihagyásával)

6. 8080-as port expose-olása
7. Spring Boot JAR indítása

Adatbázis inicializálás

Az adatbázis adatai a `mysql_data` Docker volume-ban tárolódnak, így a konténer újraindítása után is megmaradnak. A MySQL konténer első indításkor:

- Létrehozza a `hotelpms` adatbázist
- A backend Hibernate `ddl-auto=update` miatt automatikusan létrehozza a táblákat
- A `DataInitFlag` kapcsóla alapján minta adatok töltődnek be, ha ez az első indítás

Környezeti változók

A backend környezeti változói a `docker-compose.yml`-ben módosíthatók:

- `SPRING_DATASOURCE_URL`: Adatbázis kapcsolat URL
- `SPRING_DATASOURCE_USERNAME`: Adatbázis felhasználónév
- `SPRING_DATASOURCE_PASSWORD`: Adatbázis jelszó
- `SPRING_JPA_HIBERNATE_DDL_AUTO`: Séma kezelés módja

Fejlesztői mód

Production telepítéshez a fenti Docker Compose elegendő. Fejlesztési környezetben azonban érdemes a frontend-et és backend-et lokálisan futtatni hot-reloadal

3.7.5. Gyakori hibák és megoldásuk

Port már foglalt (Address already in use): Ha a 3000, 8080 vagy 3306 port már használatban van:

- Állítsd le a konfliktusban lévő alkalmazást
- Vagy módosítsd a portokat a `docker-compose.yml`-ben

MySQL healthcheck timeout: Ha a MySQL konténer nem indul el időben, növeld a `healthcheck` értékeit:

Backend nem kapcsolódik az adatbázishoz: Ellenőrizd a backend logokat

3.8. Tesztek

Többszintű tesztelési stratégiát alkalmaztam, amely lefedi az üzleti logika kritikus részeit. A tesztelés során hangsúlyt fektettem a rendszer központi funkcióinak ellenőrzésére, különös tekintettel a szobafoglalások konfliktuskezelésére és a számlázási folyamatokra.

3.8.1. Tesztelési módszertan

A teszteket három fő kategóriába tudom sorolni:

Unit tesztek

Az unit tesztek célja az egyes komponensek izolált tesztelése külső függőségek nélkül. Ezek a tesztek tisztán üzleti logikai számításokat és validációkat ellenőriznek, gyors futási idővel és magas kódlefedettséggel. A három fő utility osztályt teszteltem unit tesztekkel:

- **BookingConflictUtil:** A szobafoglalások időbeli konfliktusának detektálása
- **InvoiceCalculationUtil:** Számlák végösszegének kiszámítása ÁFÁ-val
- **RoomPriceCalculationUtil:** Szobák árának kalkulációja foglalási időszak alapján

com.hmps.backend.util

Element	Missed Instructions	Cov.	Missed Branches	Cov.
BookingConflictUtil		97%		83%
RoomPriceCalculationUtil		94%		100%
InvoiceCalculationUtil		93%		100%
Total	9 of 199	95%	3 of 40	92%

3.12. ábra. Kódlefedettség: Unit tesztek

Mock-alapú service tesztek

A service réteg tesztelése során a Mockito keretrendszer használtam az adatbázis és egyéb függőségek mockolására. Így tudtam tesztelni egyszerűen az üzleti logika komplex folyamatait anélkül, hogy valódi adatbázis-műveletek történnének.

Tesztelt service osztályok:

- **BookingService:** Foglalások életciklusa, státuszváltások, konfliktuskezelés

- **InvoiceService:** Számlák és foglalások szinkronizációja
- **ServiceModelService:** Service modellek kezelése

com.hpmms.backend.service.implament

Element	Missed Instructions	Cov.	Missed Branches	Cov.
C InvoiceService		39%		19%
C BookingService		54%		37%
C ServiceModelService		59%		38%

3.13. ábra. Kódlefedettség: Mock-alapú service tesztek

Megjegyzés: Csak a kritikusan fontos műveletek letek itt tesztelve

Manuális tesztek

Az automatizált tesztek mellett manuális teszteket is végeztem, amelyek a felhasználói felület és az end-to-end folyamatok helyes működését vizsgálják.

A manuális tesztelés célja:

- Felhasználói élmény (UX) ellenőrzése
- Böngészőkompatibilitás tesztelése
- Komplex üzleti folyamatok end-to-end tesztelése
- Vizuális megjelenés és esztétika ellenőrzése
- Olyan edge case-ek felfedezése, amelyeket automatizált tesztek nem fednek le

Tesztelési eszközök

A teszteléshez az alábbi eszközöket használtam:

- **JUnit 5:** A tesztek futtatási keretrendszer
- **Mockito:** Mock objektumok létrehozása és viselkedésük definiálása
- **JaCoCo:** Kódlefedettség mérése
- **Google Chrome**
- **Mozilla Firefox**

3.8.2. Tesztesetek

Az alábbiakban a Unit- és Mock-alapú tesztesetek táblázatai kerülnek megjelentésre, míg a manuális tesztesetek részletes ismertetése a 4. fejezetben található.

Kategória	Tesztek száma
Unit tesztek	
BookingConflictUtil	9
InvoiceCalculationUtil	7
RoomPriceCalculationUtil	11
Unit tesztek összesen	27
Mock-alapú service tesztek	
BookingService	9
InvoiceService	8
ServiceModelService	10
Mock-alapú tesztek összesen	27
Összes automatizált teszt	54
Automatizált tesztek sikerességi rátája	100%

3.9. táblázat. Tesztelési eredmények összefoglalása

3.10. táblázat. Unit tesztesetek

Teszt neve	Leírás
BookingConflictUtil	
Overlapping dates	Átfedő dátumok esetén konfliktus detektálása
No overlap	Nem átfedő foglalások ne okozzanak konfliktust
Back to back	Egymás után következő foglalások (checkout = checkin) kezelése
Cancelled booking	CANCELLED státuszú foglalás ne blokkoljon
Checked out booking	CHECKED_OUT státuszú foglalás ne okozzon konfliktust
Exclude booking ID	Saját foglalás módosításakor ne jelezzen önmagával konfliktust
Empty room list	Üres szoba lista esetén minden szabad
All rooms free	Minden szoba szabad esetén true visszaadása
One room has conflict	Egy szoba foglaltsága esetén false visszaadása
InvoiceCalculationUtil	
Null list	Null lista esetén 0.0 visszaadása
Empty list	Üres lista esetén 0.0 visszaadása
Single service without VAT	ÁFA nélküli service model: csak cost
Single service with VAT	100 Ft + 27% ÁFA = 127 Ft kalkuláció
Multiple services different VAT	Több service különböző ÁFA kulcsokkal: helyes összegzés
Service with null cost	Null cost kezelése (0.0-ként)
VAT with null percentage	Null VAT percentage kezelése (0%-ként)
RoomPriceCalculationUtil	
Null room list	Null szoba lista esetén 0.0
Empty room list	Üres szoba lista esetén 0.0
Null check-in date	Null checkIn esetén 0.0
Null check-out date	Null checkOut esetén 0.0
Checkout before check-in	Érvénytelen dátum (checkout < checkin) esetén 0.0
Same day check-in/out	Azonos napi checkin/checkout esetén 0.0
Single room	1 szoba × 100 Ft × 5 nap = 500 Ft
Multiple rooms same price	Több szoba azonos áron: helyes összegzés
Multiple rooms different price	Több szoba különböző árakon: $(100+150+200) \times 3 = 1350$ Ft
Room with null room type	Null roomType esetén szoba kihagyása
One day stay	Egynapi tartózkodás: 1 szoba × 100 Ft × 1 = 100 Ft

3.11. táblázat. BookingService tesztesetek

Teszt neve	Leírás
Create with available rooms	Foglalás létrehozása elérhető szobákkal, RESERVED státusz beállítása, 2 szoba és 1 vendég hozzárendelése
Create without rooms	Szobák nélküli foglalás létrehozása, WAITLISTED státusz beállítása
Create with conflicting rooms	Már foglalt szoba újrafoglalási kísérlete, IllegalStateException kivétel dobása, foglalás nem mentődik
Update status: Reserved to checked-in	Booking státusz váltás RESERVED-ról CHECKED_IN-re
Update status: Checked-in to checked-out	CHECKED_IN-ról CHECKED_OUT-ra váltás, szobák DIRTY státuszba állítása
Update status: Cancelled to reserved	CANCELLED-ról RESERVED-re váltás, szobák elérhetőségének újraellenőrzése
Update booking: Change dates	Foglalás dátumainak (checkIn, checkOut) és alapadatainak (név, leírás) módosítása
Delete booking	Foglalás soft delete: active flag false-ra állítása
Update: Change to waitlisted	Státusz váltás WAITLISTED-re, szobák törlése a foglalásból

3.12. táblázat. InvoiceService tesztesetek

Teszt neve	Leírás
Create with initial booking	Számla létrehozása initial booking-gal, automatikus syncWithBookings() hívás
Create without booking	Üres számla létrehozása initial booking nélkül
Sync with bookings	Booking-hoz tartozó szobákból virtuális service model generálása, scynStatus SYNCED-re állítása
Add invoice bookings	Booking hozzáadása meglévő számlához, invoice referencia beállítása, szinkronizálás
Add already taken booking	Már számlázott booking (scynStatus != NO_INVOICE) hozzáadási kísérlete IllegalArgumentException-t dob
Remove invoice bookings	Booking eltávolítása számlából, invoice referencia nullázása, scynStatus NO_INVOICE-ra állítása
Update invoice services	Service model lista frissítése, totalSum automatikus újraszámolása
Delete invoice	Invoice soft delete (active=false), kapcsolt bookings invoice és scynStatus nullázása

3.13. táblázat. ServiceModelService tesztesetek

Teszt neve	Leírás
Create service model	Új service model létrehozása megadott névvel, leírással, költséggel és VAT-tal
Create virtual service model	Virtuális service model generálás: 2 szoba (100 Ft, 150 Ft) × 5 nap = 1250 Ft
Get or create: First call	Singleton első hívás: "Rooms aggregated cost" létrehozása, immutable=true, cost=0
Get or create: Already exists	Singleton második hívás: meglévő objektum visszaadása, nem hoz létre újat
Get or create: Set immutable	Meglévő objektum immutable flag-jének utólagos beállítása ha false vagy null
Update regular service	Normál service model name, description, cost és VAT frissítése
Update immutable service	Immutable service esetén csak VAT frissíthető, name/description/cost változatlan
Update virtual service	Virtual service model cost és VAT frissítése
Delete service model	Service model törlése (deleteById), ha nincs kapcsolt invoice
Delete immutable service	Immutable service törlési kísérlete IllegalStateException-t dob

4. fejezet

Teszt jegyzőkönyv

4.0.1. Bejelentkezési képernyő

4.1. táblázat. Bejelentkezési képernyő manuális tesztjei

Funkció	Lépések	Elvárt eredmény
Sikeres bejelentkezés helyes adatokkal	1. Beírom az admin@hotel.com címet az email mezőbe 2. Beírom az admin123 jelszót a jelszó mezőbe 3. Rányomok a bejelentkezés gombra	Sikeresen bejelentkeztek a rendszerbe, a vezérlőpult fogad
Sikertelen bejelentkezés rossz jelszóval	1. Beírom az admin@hotel.com címet az email mezőbe 2. Beírom a rossz123 jelszót 3. Rányomok a bejelentkezés gombra	Hibaüzenet: "Érvénytelen bejelentkezési adatok"
Üresen hagyott mezőkkel való bejelentkezés	1. Üresen hagyom az email mezőt 2. Üresen hagyom a jelszó mezőt 3. Megpróbálok rányomni a bejelentkezés gombra	A gomb inaktív marad

Folytatás a következő oldalon

táblázat 4.1 – Folytatás az előző oldalról

Funkció	Lépések	Elvárt eredmény
Sikertelen bejelentkezés inaktív fiók miatt	1. Beírom a teszt@hotel.com címet az email mezőbe 2. Beírom a helyes jelszót 3. Rányomok a bejelentkezés gombra	"A fiók nem aktív" oldal fogad

4.0.2. Regisztrációs képernyő

4.2. táblázat. Regisztrációs képernyő manuális tesztjei

Funkció	Lépések	Elvárt eredmény
Sikeress regisztráció helyes adatokkal	1. Beírom a keresztnévet: Teszt 2. Beírom a vezetéknévet: Béla 3. Beírom az emailt: teszt.bela@hotel.com 4. Beírom a telefont: +36301234567 5. Beírom a jelszót: Jelszo123 6. Beírom a jelszó megerősítést: Jelszo123 7. Rányomok a regisztráció gombra	Sikeresen regisztrálok a rendszerbe, "A fiok még nem aktív" oldal fogad
Sikertelen regisztráció eltérő jelszavakkal	1. Kitöltöm az összes mezőt helyesen 2. Jelszó: Jelszo123 3. Jelszó megerősítés: Jelszo456 4. Rányomok a regisztráció gombra	Hibaüzenet: "A jelszavak nem egyeznek"

Folytatás a következő oldalon

táblázat 4.2 – Folytatás az előző oldalról

Funkció	Lépések	Elvárt eredmény
Sikertelen regisztráció már létező emaillel	1. Kitölöm az összes mezőt 2. Email: admin@hotel.com (létező) 3. Rányomok a regisztráció gombra	Hibaüzenet: "Az erőforrás már létezik"
Üresen hagyott kötelező mezők	1. Néhány mezőt üresen hagyok 2. Megpróbálok rányomni a regisztráció gombra	Validációs üzenetek jelennek meg

4.0.3. Foglalások oldal

4.3. táblázat. Foglalások oldal manuális tesztjei

Funkció	Lépések	Elvárt eredmény
Foglalások listázása	1. Navigálás a Foglalások menüpontra 2. Az oldal betöltődik	Megjelenik a foglalások listája táblázatos formában

Folytatás a következő oldalon

táblázat 4.3 – Folytatás az előző oldalról

Funkció	Lépések	Elvárt eredmény
Új foglalás teljes adatokkal (szobával és vendéggel)	<p>1. lépés: Alapadatok:</p> <p>1. Rányomok az "Új foglalás" gombra</p> <p>2. Beírom a nevet: Teszt Foglalás</p> <p>3. Dátumok: 2025-11-15 - 2025-11-20</p> <p>4. Leírás: Teszt leírás (opcionális)</p> <p>5. "Következő" gomb</p> <p>2. lépés: Vendégek:</p> <p>6. Kiválasztok egy létező vendéget: Teszt Béla</p> <p>7. "Következő" gomb</p> <p>3. lépés: Szobák:</p> <p>8. Kiválasztom a szobát: 101</p> <p>9. "Mentés" gomb</p>	Sikeress mentés üzenet, foglalás megjelenik "Foglalt" státusszal és "Nincs számla" számla státusszal
Új foglalás vendég nélkül	<p>1. lépés: Alapadatok:</p> <p>1. Rányomok az "Új foglalás" gombra</p> <p>2. Kitölöm a kötelező mezőket</p> <p>3. "Következő" gomb</p> <p>2. lépés: Vendégek:</p> <p>4. Nem adok hozzá vendéget</p> <p>5. "Következő" gomb</p> <p>3. lépés: Szobák:</p> <p>6. Kiválasztom a szobát: 102</p> <p>7. "Mentés" gomb</p>	Sikeress mentés, foglalás létrejön vendég nélkül, "Foglalt" státusz, "Nincs számla" számla státusz

Folytatás a következő oldalon

táblázat 4.3 – Folytatás az előző oldalról

Funkció	Lépések	Elvárt eredmény
Új foglalás szoba nélkül	<p>1. lépés: Alapadatok:</p> <ol style="list-style-type: none"> 1. Rányomok az "Új foglalás" gombra 2. Kitöltöm a kötelező mezőket 3. "Következő" gomb <p>2. lépés: Vendégek:</p> <ol style="list-style-type: none"> 4. Hozzáadok vendéget 5. "Következő" gomb <p>3. lépés: Szobák:</p> <ol style="list-style-type: none"> 6. Nem választok szobát 7. "Mentés" gomb 	Sikeres mentés, foglalás létrejön szoba nélkül, "Várólistás" státusz, "Nincs számla" számla státusz
Új foglalás sem vendég, sem szoba nélkül	<p>1. lépés: Alapadatok:</p> <ol style="list-style-type: none"> 1. Rányomok az "Új foglalás" gombra 2. Kitöltöm a kötelező mezőket 3. "Következő" gomb <p>2 és 3. lépés:</p> <ol style="list-style-type: none"> 4. Átlépkedek vendég és szoba hozzáadása nélkül 5. "Mentés" gomb 	Sikeres mentés, foglalás létrejön, "Várólistás" státusz, "Nincs számla" számla státusz

Folytatás a következő oldalon

táblázat 4.3 – Folytatás az előző oldalról

Funkció	Lépések	Elvárt eredmény
Új foglalás új vendég hozzáadásával	<p>1. lépés: Alapadatok:</p> <p>1. Kitöltöm az alapadatokat</p> <p>2. "Következő" gomb</p> <p>2. lépés: Vendégek:</p> <p>3. Elkezdem gépelni egy rendszerben nem létező vendég nevét</p> <p>4. Kiválasztom az "Új vendég hozzáadása" opciót a dropdown-ból</p> <p>5. Kitöltöm: Név, Email, Telefon</p> <p>6. Elmentem az új vendéget</p> <p>7. Az új vendég megjelenik a listában, automatikusan kiválasztva</p> <p>8. "Következő" gomb</p> <p>3. lépés: Szobák:</p> <p>9. Kiválasztok szobát</p> <p>10. "Mentés" gomb</p>	Sikeress mentés, új vendég létrejön és hozzárendelődik a foglaláshoz
Alapadatok lépés hibaellenőrzés: üres név	<p>1. lépés: Alapadatok:</p> <p>1. Rányomok az "Új foglalás" gombra</p> <p>2. Név mezőt üresen hagyom</p> <p>3. Dátumok: 2025-11-15 - 2025-11-20</p> <p>4. Megpróbálok rányomni a "Következő" gombra</p>	Validálciós üzenet: "A név mező kitöltése kötelező", nem lehet továbblépni

Folytatás a következő oldalon

táblázat 4.3 – Folytatás az előző oldalról

Funkció	Lépések	Elvárt eredmény
Alapadatok lépés hibaellenőrzés: üres dátumok	1. lépés: Alapadatok: 1. Rányomok az "Új foglalás" gombra 2. Név: Teszt 3. Dátumokat üresen hagyom 4. Megpróbálok rányomni a "Következő" gombra	Validálciós üzenet: "A dátumok megadása kötelező", nem lehet továbblépni
Alapadatok lépés hibaellenőrzés: érvénytelen dátumok	1. lépés: Alapadatok: 1. Rányomok az "Új foglalás" gombra 2. Név: Teszt 3. Érkezés: 2025-11-20 4. Távozás: 2025-11-15 (korábbi dátum) 5. Megpróbálok rányomni a "Következő" gombra	Validálciós üzenet: "A távozás dátuma nem lehet korábbi, mint az érkezés", nem lehet továbblépni
Szobák lépés: csak elérhető szobák lát-szanak	1 és 2. lépés: 1. Alapadatok: 2025-11-15 - 2025-11-20 2. Vendég hozzáadása, "Következő" 3. lépés: Szobák: 3. Megjelenik a szobák lista	Csak azok a szobák jelennek meg, amelyek a megadott dátumok között szabadok
Foglalás információk megtekintése	1. Meglévő foglalás sorában rányomok a "..." gombra 2. Kiválasztom az "Információ" opciót	Megjelenik a foglalás részletes információi

Folytatás a következő oldalon

táblázat 4.3 – Folytatás az előző oldalról

Funkció	Lépések	Elvárt eredmény
Foglalás szerkesztése: alapadatok módosítása	<p>1. Foglalás sorában "..." gomb 2. "Szerkesztés" opció kiválasztása</p> <p>1. lépés: Alapadatok:</p> <p>3. Név módosítása: Módosított Foglalás 4. Dátumok módosítása: 2025-12-01 - 2025-12-05 5. "Következő" gomb</p> <p>2 és 3. lépés:</p> <p>6. Végigmegyek a lépéseken 7. "Mentés" gomb</p>	Sikeres módosítás, foglalás frissül az új adatokkal. A számla státusz átállítódik "Nincs szinkronizálva"-ra
Foglalás szerkesztése: szoba nélkül mentés	<p>1. Foglalás sorában "..." gomb 2. "Szerkesztés" opció</p> <p>1 és 2. lépés:</p> <p>3. Alapadatok és vendégek lépések</p> <p>3. lépés: Szobák:</p> <p>4. Nem választok szobát (vagy eltávolítom az összeset) 5. "Mentés" gomb</p>	Sikeres mentés, foglalás státusza automatikusan "Várólistás"-ra vált
Foglalás szerkesztése: vendég hozzáadása	<p>1. Foglalás sorában "..." gomb 2. "Szerkesztés" opció</p> <p>2. lépés: Vendégek:</p> <p>3. További vendég hozzáadása: Kiss Anna</p> <p>4. "Következő", "Mentés"</p>	Sikeres módosítás, új vendég hozzárendelődik a foglaláshoz

Folytatás a következő oldalon

táblázat 4.3 – Folytatás az előző oldalról

Funkció	Lépések	Elvárt eredmény
Foglalás szerkesztése: vendég eltávolítása	1. Foglalás sorában "..." gomb 2. "Szerkesztés" opció 2. lépés: Vendégek: 3. Meglévő vendégnél "X" gomb 4. "Következő", "Mentés"	Sikeres módosítás, vendég eltávolítódik a foglalásból
Foglalás szerkesztése: szoba hozzáadása	1. Foglalás sorában "..." gomb 2. "Szerkesztés" opció 3. lépés: Szobák: 3. További szoba kiválasztása: 103 4. "Mentés"	Sikeres módosítás, új szoba hozzárendelődik, státusz "Foglalt"-ra vált ha korábban "Várólistás" volt
Foglalás szerkesztése: szoba eltávolítása	1. Foglalás sorában "..." gomb 2. "Szerkesztés" opció 3. lépés: Szobák: 3. Meglévő szobánál "Eltávolítás" gomb 4. "Mentés"	Sikeres módosítás, szoba eltávolítódik, ha nincs több szoba akkor státusz "Várólistás"-ra vált
Foglalás státusz módosítása: Foglalt → Bejelentkezett	1. Foglalás sorában "..." gomb 2. "Státusz módosítása" opció 3. Kiválasztom: "Bejelentkezett" 4. "Mentés" gomb	Sikeres módosítás, foglalás státusza "Bejelentkezett"-re vált
Foglalás státusz módosítása: Bejelentkezett → Kijelentkezett	1. Foglalás sorában "..." gomb 2. "Státusz módosítása" opció 3. Kiválasztom: "Kijelentkezett" 4. "Mentés" gomb	Sikeres módosítás, foglalás státusza "Kijelentkezett"-re vált, szobák státusza "Piszkoz"-ra vált

Folytatás a következő oldalon

táblázat 4.3 – Folytatás az előző oldalról

Funkció	Lépések	Elvárt eredmény
Foglalás státusz módosítása: Törölt → Foglalt	1. Törölt foglalás sorában "... gomb 2. "Státusz módosítása" opció 3. Kiválasztom: "Foglalt" 4. "Mentés" gomb	Sikeres módosítás (Ha más aktiv foglalás nem használja már foglalásban lévő szobákat), foglalás státusza "Foglalt"-ra vált, szobák elérhetőségének újraellenőrzése történik
Foglalás státusz módosítása: Várólistás-ra váltás	1. Foglalás sorában "... gomb 2. "Státusz módosítása" opció 3. Kiválasztom: "Várólistás" 4. "Mentés" gomb	Sikeres módosítás, foglalás státusza "Várólistás"-ra vált, szobák törlődnek a foglalásból
Számla szinkronizálása (ha van számla)	1. Foglalás sorában ..." gomb (foglalásnak van számlája) 2. "Számla szinkronizálása" opció	Számla szinkronizálódik a foglalással, megjelenik dialóg: "Szeretné megtekinteni a számlát?"
Számla megtekintése (ha van számla)	1. Foglalás sorában ..." gomb (foglalásnak van számlája) 2. "Számla megtekintése" opció	Átirányít a számla részletes nézetéhez
Számla létrehozása (ha nincs számla)	1. Foglalás sorában ..." gomb (foglalásnak nincs számlája) 2. "Számla létrehozása" opció	Új számla létrejön, megjelenik dialóg: "Szeretné megtekinteni a számlát?"
Számla létrehozása után megtekintés	1. Számla létrehozása 2. Dialógban "Igen" gomb	Átirányít az újonnan létrehozott számla részletes nézetéhez
Számla létrehozása után megtekintés elutasítása	1. Számla létrehozása 2. Dialógban "Nem" gomb	Maradok a foglalások listázó oldalon, dialóg bezárul

Folytatás a következő oldalon

táblázat 4.3 – Folytatás az előző oldalról

Funkció	Lépések	Elvárt eredmény
Foglalás törlése megerősítéssel	1. Foglalás sorában "... gomb 2. "Törlés" opció 3. Megerősítő dialógban "Igen" gomb	Sikeress törlés üzenet, foglalás active flag false-ra áll (soft delete)
Foglalás törlése megerősítés nélkül	1. Foglalás sorában "... gomb 2. "Törlés" opció 3. Megerősítő dialógban "Nem" gomb	Törlés megszakad, dialóg bezárul, foglalás változatlan marad

4.0.4. Vendégek oldal

4.4. táblázat. Vendégek oldal manuális tesztjei

Funkció	Lépések	Elvárt eredmény
Vendégek listázása	1. Navigálás a Vendégek menüpontra 2. Az oldal betöltődik	Megjelenik a vendégek listája táblázatos formában
Új vendég létrehozása teljes adatokkal	1. Rányomok az "Új vendég" gombra 2. Keresztnév: Teszt 3. Vezetéknév: Béla 4. Email: teszt.bela2@email.com 5. Telefon: +36301234267 6. Típus: Felnőtt 7. Aktív: Igen 8. Ország: Magyarország 9. Címkek: VIP, Törzsvásárló 10. "Mentés" gomb	Sikeress mentés üzenet, új vendég megjelenik a listában az összes megadott adattal

Folytatás a következő oldalon

táblázat 4.4 – Folytatás az előző oldalról

Funkció	Lépések	Elvárt eredmény
Új vendég típus: Gyerek	1. Rányomok az "Új vendég" gombra 2. Kitöltöm a kötelező mezőket 3. Típus: Gyerek 4. "Mentés" gomb	Sikeres mentés, vendég típusa "Gyerek"
Új vendég hibaellenőrzés: Üres keresztnév	1. Rányomok az "Új vendég" gombra 2. Keresztnév mezőt üresen hagyom 3. Kitöltöm a többi kötelező mezőt 4. "Mentés" gomb	Validálciós üzenet: "Ez a mező kötelező"
Új vendég hibaellenőrzés: Üres vezetéknév	1. Rányomok az "Új vendég" gombra 2. Vezetéknév mezőt üresen hagyom 3. Kitöltöm a többi kötelező mezőt 4. "Mentés" gomb	Validálciós üzenet: "Ez a mező kötelező"
Új vendég hibaellenőrzés: Érvénytelen email	1. Rányomok az "Új vendég" gombra 2. Email: tesztEmail (érvénytelen formátum) 3. Kitöltöm a többi kötelező mezőt 4. "Mentés" gomb	Validálciós üzenet: "Érvénytelen formátum"

Folytatás a következő oldalon

táblázat 4.4 – Folytatás az előző oldalról

Funkció	Lépések	Elvárt eredmény
Új vendég hibaellenőrzés: Már létező email	1. Rányomok az "Új vendég" gombra 2. Email: teszt.bela@email.com (már létezik) 3. Kitöltöm a többi kötelező mezőt 4. "Mentés" gomb	Rendszer üzenet: "Ezzel az e-mail címmel már létezik vendég"
Új vendég hibaellenőrzés: Érvénytelen telefonszám	1. Rányomok az "Új vendég" gombra 2. Telefon: 123abc (érvénytelen formátum) 3. Kitöltöm a többi kötelező mezőt 4. "Mentés" gomb	Validálciós üzenet: "Érvénytelen formátum"
Vendég szerkesztése	1. Vendég sorában "..." gomb 2. "Szerkesztés" opció 3. Módosítom az adatokat (pl. telefon, ország, címek) 4. "Mentés" gomb	Sikeres módosítás, vendég adatai frissülnek
Vendég aktív státusz váltása	1. Vendég sorában "..." gomb 2. "Szerkesztés" opció 3. Aktív: Nem 4. "Mentés" gomb	Sikeres módosítás, vendég inaktív státuszra vált
Vendég törlése:	1. Vendég sorában "..." gomb (vendégnek nincs foglalása) 2. "Törlés" opció 3. Megerősítő dialógban "Igen"	Sikeres törlés, vendég eltávolítódik a listából (Inaktiv lesz)

Folytatás a következő oldalon

táblázat 4.4 – Folytatás az előző oldalról

Funkció	Lépések	Elvárt eredmény
Vendég törlése megerősítés nélkül	1. Vendég sorában "... gomb (nincs foglalása) 2. "Törlés" opció 3. Megerősítő dialógban "Nem"	Törlés megszakad, dialóg bezárul, vendég változatlan marad
Vendég címkék hozzáadása	1. Vendég sorában "... gomb 2. "Szerkesztés" opció 3. Címkék mezőben hozzáadok: VIP, Törzsvendég 4. "Mentés" gomb	Sikeress mentés, címkék megjelennek a vendég adatainál
Vendég címkék eltávolítása	1. Vendég sorában "... gomb 2. "Szerkesztés" opció 3. Meglévő címkéknél "X" gomb (törölés) 4. "Mentés" gomb	Sikeress mentés, címkék eltávolítódnak a vendég adataiból

4.0.5. Számlák oldal

4.5. táblázat. Számlák oldal manuális tesztjei

Funkció	Lépések	Elvárt eredmény
Számlák listázása	1. Navigálás a Számlák menüpontra 2. Az oldal betöltődik	Megjelenik a számlák lista táblázatos formában bal oldalon, jobb oldal üres
Számla részleteinek megtekintése	1. Számlák listájában rágattintok egy számla sorára	A számla részletei megjelennek a jobb oldali panelen

Folytatás a következő oldalon

táblázat 4.5 – Folytatás az előző oldalról

Funkció	Lépések	Elvárt eredmény
Másik számla kiválasztása	1. Egy számla már meg van nyitva a jobb oldalon 2. Rákattintok egy másik számla sorára	A korábban megnyitott számla bezárul, az új számla részletei jelennek meg a jobb oldalon
Új számla létrehozása	1. Rányomok az "Új számla" gombra 2. Név: Teszt Számla 3. "Mentés" gomb	Sikeressé mentés, üres számla létrejön "Függőben" fizetési státusszal, megjelenik a listában
Új számla hibaellenőrzés: Üres név	1. Rányomok az "Új számla" gombra 2. Név mezőt üresen hagyom 3. "Mentés" gomb	Validálciós üzenet: "Ez a mező kötelező"
Számla alapadatok szerkesztése	1. Számla kiválasztása 2. Jobb oldali panelen "Szerkesztés" gomb 3. Név módosítása: Módosított Számla 4. Leírás: Új leírás 5. Státusz: Fizetve 6. "Mentés" gomb	Sikeressé módosítás, számla alapadatai frissülnek
Számla címzett adatok szerkesztése	1. Számla kiválasztása 2. "Szerkesztés" gomb 3. Címzett adatok szerkesztése: Név, Cím, Adószám... 4. "Mentés" gomb	Sikeressé módosítás, címzett adatok frissülnek
Számla alapadatok és címzett változtatások eldobása	1. Számla kiválasztása 2. "Szerkesztés" gomb 3. Módosítom az alapadatokat és címzett adatokat 4. "Visszaállítás" gomb	Változtatások eldobódnak, eredeti adatok visszaállnak

Folytatás a következő oldalon

táblázat 4.5 – Folytatás az előző oldalról

Funkció	Lépések	Elvárt eredmény
Új szolgáltatás hozzáadása számlához	1. Számla kiválasztása 2. Szolgáltatások szekciójánál "+" gomb 3. Új sor jelenik meg a szolgáltatások táblázat alján 4. Kiválasztok egy szolgáltatást a legördülőből 5. Sorvégi "Mentés" ikon	Sikeressé mentés, új szolgáltatás hozzáadódik a számlához, összeg újraszámolódik
Szolgáltatás hozzáadásának eldobása	1. Számla kiválasztása 2. Szolgáltatások szekciójában "+" gomb 3. Új sor jelenik meg 4. Kiválasztok egy szolgáltatást 5. Sorvégi "X" gomb	Új szolgáltatás hozzáadása megszakad, sor eltűnik, változatlan marad a számla
Szolgáltatás eltávolítása számlából	1. Számla kiválasztása 2. Meglévő szolgáltatás sorában "X" gomb 3. Megerősítő dialógban "Igen"	Sikeressé törlés, szolgáltatás eltávolítódik, összeg újraszámolódik
Új foglalás hozzáadása számlához	1. Számla kiválasztása 2. Foglalások szekciójában "+" kártyára kattintás 3. Legördülőből kiválasztok egy foglalást (csak olyan foglalások látszanak, amik nem tartoznak más számlához) 4. "Hozzáadás" gomb	Sikeressé hozzáadás, foglalás hozzárendelődik a számlához, megjelenik kártyaként
Foglalás eltávolítása számlából	1. Számla kiválasztása 2. Foglalás kártya sarkában "X" gomb 3. Megerősítő dialógban "Igen"	Sikeressé eltávolítás

Folytatás a következő oldalon

táblázat 4.5 – Folytatás az előző oldalról

Funkció	Lépések	Elvárt eredmény
Foglalás eltávolítása megerősítés nélkül	1. Számla kiválasztása 2. Foglalás kártya sarkában "X" gomb 3. Megerősítő dialógban "Nem"	Eltávolítás megszakad, dialóg bezárul, foglalás változatlan marad
Foglalás hozzáadása: Csak szabad foglalások	1. Számla kiválasztása 2. "+" kártya 3. Legördülő megnyílik	Csak azok a foglalások jelennek meg, amelyek nem tartoznak más számlához
Számla összeg automatikus számítása	1. Számla kiválasztása 2. Szolgáltatás vagy foglalás hozzáadása/eltávolítása	A számla végösszege automatikusan újraszámolódik
Számla nyomtatása (PDF letöltés)	1. Számla kiválasztása 2. A számla neve mellett "Nyomtatás" ikon gomb	PDF fájl letöltődik a számlával
Számla törlése	1. Számla kiválasztása 2. A számla neve mellett "Törlés" gomb 3. Megerősítő dialógban "Igen"	Sikeres törlés, számla active flag false-ra áll (inaktiv), soft delete
Számla törlése megerősítés nélkül	1. Számla kiválasztása 2. A számla neve mellett "Törlés" gomb 3. Megerősítő dialógban "Nem"	Törlés megszakad, dialóg bezárul, számla változatlan marad

4.0.6. Takarítás oldal

4.6. táblázat. Takarítás oldal manuális tesztjei

Funkció	Lépések	Elvárt eredmény
Takarítási feladatak listázása	1. Navigálás a Takarítás menüpontra 2. Az oldal betöltődik	Megjelenik a takarítási feladatok listája
Új takarítási feladat létrehozása dolgozó nélkül	1. Rányomok az "Új takarítás" gombra 2. Szoba: 101 3. Prioritás: Magas 4. Megjegyzés: Alapos takarítás szükséges 5. Hozzárendelt dolgozó: üresen hagyom 6. "Mentés" gomb	Sikeressé mentés, új feladat létrejön dolgozó és hozzárendelés dátuma nélkül, státusz: "Teendő"
Új takarítási feladat létrehozása dolgozóval	1. Rányomok az "Új takarítás" gombra 2. Szoba: 102 3. Prioritás: Normál 4. Megjegyzés: Gyors takarítás 5. Hozzárendelt dolgozó: Kiss Anna 6. "Mentés" gomb	Sikeressé mentés, új feladat létrejön hozzárendelt dolgozóval, aktuális dátum hozzárendelve, státusz: "Teendő"
Új takarítási feladat hibaellenőrzés: Üres szoba	1. Rányomok az "Új takarítás" gombra 2. Szoba mezőt üresen hagyom 3. Kitölöm a többi mezőt 4. "Mentés" gomb	Validálciós üzenet: "Ez a mező kötelező"
Feladat szerkesztése: Dolgozó hozzárendelése	1. Feladat sorában "..." gomb 2. "Szerkesztés" opción 3. Hozzárendelt dolgozó: Kovács Béla 4. "Mentés" gomb	Sikeressé módosítás, dolgozó hozzárendelődik, aktuális dátum hozzárendelődik a feladathoz

Folytatás a következő oldalon

táblázat 4.6 – Folytatás az előző oldalról

Funkció	Lépések	Elvárt eredmény
Gyors státusz váltás: Teendő → Folyamatban	1. Feladat sorában státusz dropdown 2. Kiválasztom: "Folyamatban"	Státusz azonnal frissül "Folyamatban"-ra, befejezési dátum nincs
Gyors státusz váltás: Folyamatban → Kész	1. Feladat sorában státusz dropdown 2. Kiválasztom: "Kész"	Státusz azonnal frissül "Kész"-re, aktuális dátum hozzárendelődik befejezési dátumként
Gyors státusz váltás: Kész → Folyamatban	1. Befejezett feladat sorában státusz dropdown 2. Kiválasztom: "Folyamatban"	Státusz frissül "Folyamatban"-ra, befejezési dátum törlődik
Gyors státusz váltás: Kész → Teendő	1. Befejezett feladat sorában státusz dropdown 2. Kiválasztom: "Teendő"	Státusz frissül "Teendő"-re, befejezési dátum törlődik
Gyors státusz váltás: Folyamatban → Teendő	1. Folyamatban lévő feladat sorában státusz dropdown 2. Kiválasztom: "Teendő"	Státusz azonnal frissül "Teendő"-re
Feladataim megjelenítése: Kapcsoló bekapcsolása	1. "Feladataim megjelenítése" kapcsoló bekapcsolása	Csak azok a feladatok jelennek meg, amelyekhez az aktuális felhasználó van hozzárendelve dolgozóként
Feladataim megjelenítése: Kapcsoló kikapcsolása	1. "Feladataim megjelenítése" kapcsoló kikapcsolása	Minden takarítási feladat megjelenik a listában
Feladataim szűrés: Nincs hozzárendelt feladat	1. "Feladataim megjelenítése" toggle bekapcsolása (felhasználóhoz nincs feladat hozzárendelve)	Üres lista jelenik meg és üzenet: "Nincs találat"

Folytatás a következő oldalon

táblázat 4.6 – Folytatás az előző oldalról

Funkció	Lépések	Elvárt eredmény
Prioritás módosítása	1. Feladat sorában "... " gomb 2. "Szerkesztés" opció 3. Prioritás: Alacsony → Magas 4. "Mentés" gomb	Sikeres módosítás, prioritás frissül
Megjegyzés módosítása	1. Feladat sorában "... " gomb 2. "Szerkesztés" opció 3. Megjegyzés módosítása: "Sürgős takarítás" 4. "Mentés" gomb	Sikeres módosítás, megjegyzés frissül
Feladat törlése	1. Feladat sorában "... " gomb 2. "Törlés" opció 3. Megerősítő dialógban "Igen"	Sikeres törlés, feladat eltávolítódik a listából
Feladat törlése megerősítés nélkül	1. Feladat sorában "... " gomb 2. "Törlés" opció 3. Megerősítő dialógban "Nem"	Törlés megszakad, dialóg bezárul, feladat változatlan marad

5. fejezet

Összegzés

Szakdolgozatom célja egy modern, egyszerű, de hatékony webalapú hotelmenedzsment és foglalási rendszer készítése volt. Elkészítettem egy átfogó hotelmenedzsment rendszert, amelyben lehet kezelní a foglalásokat, szobákat, számlákat, takarításokat és a hotel alapvető adatait.

Technológiailag felhasználtam az egyetemen elsajátított React és TypeScript tudásomat a frontend megvalósításához és ugyancsak az egyetemtől megszerzett MySQL tudást az adatbázis felépítéséhez. Újdonságként megtanultam a projekthez a Java Spring Boot keretrendszer a backend elkészítéséhez.

Ezek mellett sok apróbb új technológiát használtam a szakdolgozat során, például Jotai state management-et Redux helyett, vagy OpenPDF-et nyomtatványok generálásához.

Úgy gondolom, a rendszer mostani állapota megfelelően használható kisebb száloldák igényei számára, de sajnos nem lett olyan komplex, mint szerettem volna. Ez leginkább abban látszik meg, hogy nincs beépített channel manager. (Ez tenné lehetővé az automatikus kommunikálást a rendszer és más foglalási platformok között, például Booking.com.) Azonban a rendszer moduláris mivolta miatt továbbfejlesztési lehetőségekkel nem kellene akkora kihívást okoznia egy channel manager kiépítése.

Összességében sikerült egy működőképes, modern szállodamenedzsment rendszert létrehoznom, amely valós problémára nyújt megoldást.

6. fejezet

Függelék

6.1. Szerepkörök

6.1. táblázat. Felhasználói szerepkörök és jogosultságaik

Szerepkör neve	Jogosultságok
Admin	Teljes rendszer hozzáférés: felhasználók kezelése, szerepkörök módosítása, minden adat megtekintése és szerkesztése
Receptionist	Recepciós műveletek: foglalások megtekintése/létrehozása/módosítása, számlák megtekintése, vendégek megtekintése/létrehozása/módosítása és nyomtatása , Számlák szinkronizálása foglalásokkal, szobatükör megtekintése
Housekeeper	Takarítási feladatok: housekeeping feladatok megtekintése/létrehozása/módosítása
Invoice Manager	Pénzügyi műveletek: számlák megtekintése/létrehozása/módosítása és nyomtatása, Számlák szinkronizálása foglalásokkal. ÁFA kulcsok megtekintése/létrehozása/módosítása és Szolgáltatások megtekintése/létrehozása/módosítása
Data Maintainer	Adatbázisban található adatok létrehozása/szerkesztése (kivéve a felhasználói adatok és szerepkörök)
Alap Felhasználó	Vezérlőpult megtekintése, alapadatok megtekintése. minden felhasználó rendelkezik ezekkel a jogosultságokkal az egyéni szerepkörétől függetlenül

Ábrák jegyzéke

2.1.	Alapértelmezetten megjelenő bejelentkezési képernyő	8
2.2.	Új felhasználó létrehozását dokumentáló képek	9
2.3.	Alapértelmezett admin fiók törlésé	10
2.4.	Vezérlőpult megjelenése	11
2.5.	Bevételi gráf időszak beállítása	12
2.6.	Táblaműveletek	13
2.7.	Oszlopszűrés és rendezés felület	13
2.8.	Példa: Új épület hozzáadásának lépései	14
2.9.	Példa: Épület szerkesztésének lépései	15
2.10.	Példa: Épület törlése gomb	16
2.11.	Épületek felület	17
2.12.	Szoba szolgáltatások felület	17
2.13.	Vendég címkek felület	18
2.14.	Ágytípusok felület	19
2.15.	Szerepkörök felület	20
2.16.	Felhasználók felület	21
2.17.	Szobatípusok felület	22
2.18.	Szobák felület	23
2.19.	Szoba műveletek	23
2.20.	Változásnapló felület	24
2.21.	Cég információ felület	25
2.22.	Vendégek felület	26
2.23.	Foglalások felület	27
2.24.	Új foglalás létrehozásának lépései	29
2.25.	Szobatükör felület	31
2.26.	ÁFA kulcsok felület	32
2.27.	Szolgáltatások felület	33

2.28. Számlák felület teljes nézet	34
2.29. Számla nyomtatás gomb	35
2.30. Számla törlése gomb	36
2.31. Takarítások felület	36
2.32. Takarítás felület műveletei	37
2.33. Feladat szerkesztése	38
2.34. Példa űrlap validációs hibákra	39
2.35. Művelet végrehajtási üzenetek	39
 3.1. Rendszer háromrétegű architektúrája	41
3.2. Adatbázis Entitás-Kapcsolat Diagram	46
3.3. Rendszer architektúra áttekintése	54
3.4. Booking modul osztálydiagramja	57
3.5. Navigációs menü	60
3.6. Bejelentkezési felület	60
3.7. Dashboard nézet statisztikákkal	61
3.8. Táblázatos lista nézet műveleti gombokkal	62
3.9. Modal ablak	62
3.10. Szobatükör nézet foglalások vizualizálásához	63
3.11. Navigációs diagram	63
3.12. Kódlefedettség: Unit tesztek	95
3.13. Kódlefedettség: Mock-alapú service tesztek	96

Táblázatok jegyzéke

2.1. Alapértelmezett adminisztrátori fiók	8
3.1. User tábla szerkezete	48
3.2. Guest tábla szerkezete	49
3.3. Booking tábla szerkezete	50
3.4. Room tábla szerkezete	51
3.5. Invoice tábla szerkezete	52
3.6. guest_booking kapcsolótábla	53
3.7. room_type_bed_type kapcsolótábla szerkezete	53
3.8. Szolgáltatások elérhetősége	90
3.9. Tesztelési eredmények összefoglalása	97
3.10. Unit tesztesetek	98
3.11. BookingService tesztesetek	99
3.12. InvoiceService tesztesetek	99
3.13. ServiceModelService tesztesetek	100
4.1. Bejelentkezési képernyő manuális tesztjei	101
4.2. Regisztrációs képernyő manuális tesztjei	102
4.3. Foglalások oldal manuális tesztjei	103
4.4. Vendégek oldal manuális tesztjei	111
4.5. Számlák oldal manuális tesztjei	114
4.6. Takarítás oldal manuális tesztjei	118
6.1. Felhasználói szerepkörök és jogosultságai	122

Forráskódjegyzék

3.1.	GET /api/amenities - Lista lekérdezés	65
3.2.	POST /api/amenities - Uj elem letrehozása	66
3.3.	PUT /api/amenities/{id} - Elem módosítása	67
3.4.	DELETE /api/amenities/{id} - Elem törlése	69
3.5.	Request validálció Bean Validation-al	70
3.6.	Egyszerü DTO konverzió	71
3.7.	Komplex DTO konverzió ha több entitást is vissza kell adni	71
3.8.	Épületek szűrés kérés	72
3.9.	Szűrés függvény a BuildingService-ben	73
3.10.	FormConfig TypeScript típusok	75
3.11.	FormFactory komponens	76
3.12.	FieldWrapper - dinamikus beviteli mező választás	77
3.13.	Vendég címke űrlap konfiguráció	78
3.14.	FormModal komponens (Egyszerűsített)	79
3.15.	Validációs rendszer	80
3.16.	BedTypeQuantity field konfiguráció	81
3.17.	Dinamikus form konfiguráció VAT opciókkal	82
3.18.	Virtuális ServiceModel létrehozása	84
3.19.	Base ServiceModel létrehozása/lekérése	85
3.20.	Virtuális szolgáltatások szűrése	86
3.21.	Virtuális ServiceModel módosítása	87
3.22.	Projekt letöltése	89
3.23.	Docker Compose indítása	89
3.24.	Docker Compose háttérben	90
3.25.	docker-compose.yml struktúra	90
3.26.	Frontend Dockerfile	92

3.27. Backend Dockerfile 93