**Orvosi időpontfoglaló rendszer dokumentáció**

***1. Projektleírás / Bevezető:***

Ez a projekt egy teljesen működőképes és biztonságos rendszert mutat be orvosi időpontfoglaláshoz, amely WPF-et, React Native-et, Node.js-t és MySQL-t használ. A rendszer célja, hogy modern és hatékony alternatívát biztosítson a hagyományos, papíralapú időpontfoglalással szemben, amely gyakran pontatlansághoz, átláthatatlansághoz és adminisztrációs nehézségekhez vezethet.

Az alkalmazás lehetővé teszi a gördülékeny interakciót az adminisztrátorok, az orvosok és a páciensek között. A páciensek egyszerűen tudnak időpontot foglalni, módosítani vagy lemondani; az orvosok áttekinthetik a saját rendelési idejüket és időbeosztásukat; az adminisztrátorok pedig teljeskörűen menedzselhetik a felhasználói adatokat és foglalásokat.

A projekt megvalósítása során a modern szoftverfejlesztési gyakorlatokat követtem: a biztonságos adatkezelés érdekében MySQL alapú adatbázist és titkosított jelszókezelést alkalmaztam, a valós idejű frissítéseket Node.js biztosítja, míg a felhasználói élményt a WPF és a React Native adja. Az alkalmazás a világos UI/UX elveknek megfelelően készült, így egyszerre esztétikus és könnyen használható.

A projekt során gyakorlati tapasztalatot szereztem a full-stack fejlesztésben, a backend integrációban és az asztali alkalmazások tervezésében, amely hozzájárult szakmai fejlődésemhez is.

***2. Célkitűzés:***

A projekt elsődleges célja egy olyan orvosi időpontfoglaló rendszer kialakítása, amely egyszerűsíti és digitalizálja a rendelőkben zajló adminisztratív folyamatokat. A hagyományos, papíralapú vagy telefonos időpontfoglalási módszerek gyakran pontatlanok, nehezen kezelhetők és időigényesek mind a páciensek, mind az egészségügyi dolgozók számára. A rendszer célja, hogy ezekre a problémákra modern, biztonságos és felhasználóbarát megoldást nyújtson.

A fejlesztés során kiemelt cél volt:

* a páciensek számára egy átlátható és könnyen használható felület biztosítása időpontfoglalásra, módosításra és lemondásra,
* az orvosok számára a saját rendelési idejük és foglalásaik egyszerű áttekintése és kezelése,
* az adminisztrátorok számára a teljes rendszer felügyelete, a felhasználói adatok és foglalások menedzselése,
* a biztonság megteremtése az érzékeny egészségügyi és személyes adatok kezelése során,
* valamint a valós idejű szinkronizáció, amely lehetővé teszi, hogy minden felhasználó mindig a legfrissebb adatokat lássa.

A projekt hosszú távú célja, hogy egy olyan skálázható rendszert hozzon létre, amely nemcsak egyetlen rendelőben, hanem akár egy teljes egészségügyi intézmény hálózatában is alkalmazható, bővíthető további funkciókkal, például e-mail vagy SMS értesítésekkel, többnyelvű felülettel és integrált statisztikai elemzésekkel.

***3. Rendszerkövetelmények:***

Hardverkövetelmények:

Szerver oldal: legalább 1 magos processzor, 1 GB RAM és 2 GB szabad tárhely, stabil internetkapcsolattal.

Asztali kliens (WPF alkalmazás): Windows 10 vagy újabb operációs rendszerrel rendelkező számítógép, 2 GB RAM és minimum 1 GB szabad tárhely.

Mobil kliens (React Native alkalmazás): Android 9.0 vagy újabb, illetve iOS 13 vagy újabb operációs rendszert futtató okostelefon vagy tablet, internetkapcsolattal.

Szoftverkövetelmények:

* Szerver oldalon: Node.js futtatókörnyezet, MySQL adatbázis.
* Asztali kliens oldalon: .NET keretrendszer a WPF alkalmazás futtatásához.
* Mobil kliens oldalon: React Native környezet, Expo CLI.
* Hálózati követelmények
* Stabil internetkapcsolat a valós idejű szinkronizációhoz.
* Lokális hálózat opcionálisan, ha a rendszer egyetlen rendelőben működik.

***4. Felhasználói dokumentáció:***

Ez a fejezet közvetlenül Önnek szól, aki a rendszer felhasználója. Az alábbiakban részletesen megtalálja a teendőit attól függően, hogy páciensként, orvosként vagy adminisztrátorként jelentkezik be.

***Pácienseknek:***

Amennyiben Ön páciensként használja a rendszert:

1. Nyissa meg az alkalmazást, és kattintson a **Regisztráció** gombra. Adja meg adatait (név, e-mail cím, jelszó).
2. Bejelentkezés után ki tudja választani a számára legmegfelelőbb időpontot.
3. Válassza ki a kívánt orvost és a kezelés típusát.
4. Kattintson a **Foglalás** gombra. A rendszer azonnal jelzi egy üzenetben, hogy sikeres, vagy nem sikeres az időpont foglalása.
5. A **Saját időpontjaim**menüpontban bármikor megtekintheti, módosíthatja vagy törölheti foglalásait.

Példa: Ha kedden 10:00-ra szeretne időpontot foglalni dr. Kovácshoz, a rendszer a foglalás után azonnal visszaigazolást küld.

Ha a választott időpont már foglalt, a rendszer hibaüzenetet ad.

***Orvosoknak:***

Amennyiben Ön orvosként lép be a rendszerbe:

1. Jelentkezzen be a saját orvosi fiókjával.
2. Válassza a **Naptár** menüpontot, ahol áttekintheti a napi, heti vagy havi foglalásokat.
3. A foglalásoknál láthatja a páciens nevét és a kezelés típusát.
4. Ha új rendelési időszakot kíván megnyitni, válassza a **Rendelési idő módosítása** menüpontot.

***Adminisztrátoroknak:***

Amennyiben Ön adminisztrátorként használja a rendszert:

1. Jelentkezzen be az adminisztrátori fiókjával.
2. A **Felhasználók kezelése** menüpontban létrehozhat, módosíthat vagy törölhet pácienseket és orvosokat.
3. A **Foglalások áttekintése** menüpontban ellenőrizheti a bejegyzéseket, hibás foglalásokat törölhet.

***2. Fejlesztői Dokumentáció:***

A fejlesztői dokumentáció azoknak készül, akik a programban hibát keresnek, javítanak, hatékonyabbá tesznek, más környezetbe átültetnek, más nyelvre átírnak vagy továbbfejlesztenek. Ez a dokumentáció tartalmaz minden olyan információt, amely a program működésének pontos megértéséhez szükséges.

**Specifikáció és követelmények:**  
A rendszer célja egy digitális orvosi időpontfoglaló megoldás, amely egyszerre szolgálja ki a pácienseket, orvosokat és adminisztrátorokat. Biztonságos adatkezelés, jogosultsági szintek, egyszerű kezelhetőség, valós idejű frissítés, többplatformos működés (Windows, Android, iOS).

**Futási környezet:**

* Szerver: Node.js futtatókörnyezet, MySQL adatbázis.
* Asztali kliens: Windows 10 vagy újabb, .NET keretrendszerrel.
* Mobil kliens: Android 9+ vagy iOS 13+.
* Internetkapcsolat a kliens és szerver között.

**Fejlesztői környezet:**

* Programnyelvek: JavaScript (Node.js, React Native), C# (WPF).
* Könyvtárak: express, cors, mysql2, sequelize, dotenv, expo-cli.
* Adatbázis: MySQL, Sequelize ORM-mel kezelve.

**Algoritmusok és adatok:**

* Adatbázistáblák: users (felhasználók), doctors (orvosok), appointments (foglalások).
* Kapcsolatok: egy páciens több időpontot foglalhat, egy orvoshoz több páciens tartozhat.
* Algoritmusok biztosítják, hogy az időpontok ne ütközzenek.

**Kódolási szabványok:**

* REST API endpointok egységes JSON választ adnak.
* Sequelize modellek és változók angol elnevezése.
* Jelszavak titkosítása.

**Tesztesetek:**

* Páciens foglal → az időpont bekerül az adatbázisba.
* Admin töröl hibás foglalást → a páciens listájából is eltűnik.

**Hatékonysági szempontok:**

* MySQL indexek a gyors kereséshez.
* Minimális válaszidő (<1s normál terhelés mellett).

**Fejlesztési lehetőségek:**

* Értesítések (SMS, e-mail).
* Online fizetés integrálása.
* Több rendelő összehangolt támogatása.

**Koncepcióterv:**

A cél egy modern, valós idejű időpontfoglaló rendszer kialakítása, amely egyszerre kényelmes a pácienseknek és hatékony az orvosoknak.

***Rendszerterv:***

A rendszer kliens–szerver architektúrára épül:

* **Frontend:** WPF (Windows kliens), React Native (mobil).
* **Backend:** Node.js, Express.
* **Adatbázis:** MySQL, Sequelize ORM.

***5. Fejlesztői dokumentáció:***

A fejlesztői dokumentáció célja, hogy segítséget nyújtson a program karbantartásához és továbbfejlesztéséhez.

Fejlesztői környezet és használt eszközök:

* Programozási nyelvek: JavaScript (Node.js, React Native), C# (WPF)
* Adatbázis-kezelő: MySQL
* Backend könyvtárak: express, cors, mysql2, sequelize, dotenv
* Frontend eszközök: Expo CLI, React Native
* Fejlesztői környezet: Visual Studio, Visual Studio Code
* Követelmények: Node.js 18+, .NET 6+, MySQL 8.0+, XAMPP

Adatszerkezetek és táblák példák:

* Users (id, név, email, jelszó)
* Doctors (id, név, szakirány, rendelési idő)
* Appointments (id, páciens\_id, orvos\_id, időpont, státusz)

Algoritmusok rövid leírása:

* Foglaláskor a rendszer ellenőrzi, hogy az adott orvos adott időpontban szabad-e.
* Ha foglalt, hibaüzenetet ad vissza.
* Ha szabad, létrehoz egy új bejegyzést az Appointments táblában.

Továbbfejlesztési lehetőségek:

* E-mail vagy SMS értesítések küldése foglalás után.
* Több rendelő és intézmény támogatása.
* Statisztikai riportok exportálása PDF/Excel formátumban.

***6. Installálási kézikönyv és operátori kézikönyv***

*Backend telepítése:*

1. Töltsük le a projekt backend mappáját.

2. Az App.js-ben át kell írni a IP címeket, a helyi IP címre.

3. Nyissuk meg a mappát, majd telepítsük a szükséges könyvtárakat (express, cors, mysql2, sequelize, dotenv).

* npm i express cors mysql2 sequelize dotenv

4. A szerver futtatásához szükséges a XAMPP Control Panel használata. Ebben el kell indítani a **MySQL** és **Apache** szolgáltatásokat, hogy az adatbázis és a szerver megfelelően működjön.

5. A szerver indítása a következő paranccsal történik:

* node server.js

6.Adatbázis létrehozása phpMyAdmin-ban:

1. Indítsa el a **XAMPP CONTROL PANEL** -t, majd a MySQL sorban kattintson az **Admin** gombra.
2. Ekkor a böngészőben automatikusan megnyílik a phpMyAdmin felület.
3. A bal oldali listában több rendszeradatbázist fog látni, viszont ezekhez **nem szükséges hozzányúlni.**
4. A bal felső sarokban kattintson az **Új** gombra.
5. A „Adatbázis neve” mezőbe írja be:

patientproject

majd kattintson a **Létrehozás** gombra.

1. A bal oldali listában most meg fog jelenni a patientproject nevű adatbázis.
2. Ebben a pillanatban még nincsenek benne táblák, de ez teljesen rendben van.
3. Ha elindítja a szervert a node server.js paranccsal, akkor a Sequelize automatikusan létrehozza a szükséges táblákat a patientproject adatbázisban.
4. Amint a táblák megjelennek (például users, doctors, appointments), lehetőség van kézzel adatokat hozzáadni.
   * Ehhez kattintson a bal oldali listában a kívánt táblára (például users).
   * A felső menüben válassza a **Beszúrás** fület.
   * Töltse ki a megjelenő űrlapot a kívánt adatokkal (például hozzon létre egy admin felhasználót vagy egy orvost).
   * A kitöltés után kattintson a **Mehet** gombra, így az adat elmentésre kerül az adatbázisban.

***Mobil kliens (React Native) telepítése:***

Telepítsük a **React Native** környezetet és az Expo CLI-t:

* npm i -g expo-cli
* npx expo init (Ez a parancs létrehoz egy új mappát a projektnek ott, ahol futtatjuk.)

*Projekt indítása:*

npx expo start

Ezután az alkalmazás Androidon vagy iOS-en futtatható emulátorral vagy valódi eszközön.

*Asztali kliens (WPF) telepítése:*

* Szükséges a Windows 10 vagy újabb rendszer és a .NET keretrendszer.
* A projekt futtatása Visual Studio segítségével történik.
* Az alkalmazás automatikusan csatlakozik a szerverhez.

Operátori kézikönyv:

(A szerver indítása: node server.js.)

* A XAMPP Control Panel-ben a MySQL és Apache modulokat minden indítás előtt futtatni kell.
* Rendszeres ellenőrzés, hogy a szerver és az adatbázis fut-e.
* Hibák esetén újraindítás vagy konfigurációs fájlok ellenőrzése.
* Rendszeres biztonsági mentések készítése az adatbázisról.
* Felhasználói fiókok kezelése (létrehozás, módosítás, törlés).

***7. Hibakezelés és hibaüzenetek***

A rendszerben előforduló leggyakoribb hibák és azok kezelése:

* „Adatbázis kapcsolat sikertelen”
* Oka: nincs elindítva a MySQL a XAMPP-ban, vagy hibás a konfiguráció.
* Megoldás: indítsa el a XAMPP-ban a MySQL-t, és ellenőrizze a .env fájlban a beállításokat.
* „Az időpont már foglalt”
* Oka: egy másik páciens ugyanarra az időpontra már lefoglalt időpontot.
* Megoldás: válasszon másik szabad időpontot.
* „Érvénytelen felhasználónév vagy jelszó”
* Oka: hibás bejelentkezési adatok.
* Megoldás: próbálja újra, vagy kérjen jelszó-visszaállítást az adminisztrátortól.

***8. Mintaalkalmazás / Példafutás***

* Egy tipikus használati folyamat a rendszerben:
* Páciens belép a rendszerbe.
* Megnyitja az „Időpontfoglalás” menüpontot.
* Kiválasztja az „Dr. Kovács Péter – Fogorvos” szakrendelést.
* Látja, hogy a jövő hét szerdán 14:00 szabad.
* Lefoglalja az időpontot → a rendszer visszajelzést ad: „Sikeres foglalás”.
* Az orvos belép a saját fiókjával, és a naptárban látja, hogy szerdára 14:00-óra van foglalás.
* Az adminisztrátor később lekéri a havi statisztikát, amely megmutatja, hogy az adott hónapban 120 foglalás történt.

***9. Programismertető***

Az orvosi időpontfoglaló rendszer célja, hogy modern, digitális megoldást nyújtson a rendelők időpontkezelésére. A rendszer biztonságos adatkezelést, valós idejű szinkronizációt és könnyen használható grafikus felületet biztosít a páciensek, az orvosok és az adminisztrátorok számára.

Fő előnyök:

* Gyors és egyszerű időpontfoglalás, lemondás és módosítás.
* Orvosoknak átlátható rendelési naptár.
* Adminisztrátoroknak teljes felügyelet és statisztikai lehetőségek.
* Több platformon elérhető: Windows asztali kliens, Android és iOS mobilalkalmazás.

***10. Dokumentációs tulajdonságok***

* Világos szerkezet: a fejezetek logikus sorrendben épülnek fel.
* Érthetőség: a szöveg közérthető.
* Formai követelmények: tagolás, számozás, címsorok, felsorolások.
* Használhatóság: a felhasználói és operátori kézikönyv gyakorlati példákat ad.