Barbershop Időpontfoglaló Rendszer Dokumentációja

Projekt neve: Barbershop Appointment
Fejlesztő: Polgár Ádám, Strung Dávid,
Dátum: 2025.04.15

Tartalomjegyzék

---

1. Bevezetés

1.1. Projekt Áttekintése

A Barbershop Időpontfoglaló Rendszer célja, hogy egyetlen, konkrét borbélyüzlet ügyfelei számára egyszerű és hatékony online időpontfoglalási szolgáltatást biztosítson. A rendszer lehetővé teszi:

- Az ügyfelek számára a szabad időpontok megtekintését, foglalását és esetleges módosítását.
- Az adminisztrátor számára a foglalások kezelését (megtekintés, törlés).
- A felhasználók és adminisztrátorok számára dinamikusan generált navigációs menük révén történő bejelentkezést, jogosultságok kezelését.

1.2. Célok és Felhasználási Terület

- **Fő cél:** Az időpontfoglalás teljes folyamatának digitalizálása és automatizálása egyetlen barbershop számára.
- **Felhasználási terület:**
    - Ügyfelek: Időpontfoglalás, profiladatok módosítása.
    - Adminisztrátor: Foglalási adatok kezelése, hibakezelés, rendszerkarbantartás.
- **Elvárt eredmények:**
    - Átlátható, gyors és biztonságos foglalási rendszer.
    - Egyszerű adminisztráció és hatékony ügyfélkezelés.

1.3. A Dokumentáció Felépítése

A dokumentáció modulokra bontva ismerteti a rendszer fő komponenseit. Minden modul esetében részletes kódelemzés, funkcionális magyarázat, valamint a kapcsolódó hibakezelési és biztonsági megfontolások kerülnek tárgyalásra.

---

2. Technológiai Követelmények és Környezet

2.1. Használt Programozási Nyelv és Könyvtárak

- **PHP:** A backend logikához, különösen a session kezeléshez, a JWT alapú autentikációhoz, és az adatbázis műveletekhez.
- **JavaScript:** A frontend dinamikus viselkedéséhez, pl. időpontválasztó frissítése, AJAX kérések.
- **Bootstrap:** Reszponzív felhasználói felület kialakításához.
- **Composer:** Függőségkezeléshez, melynek segítségével a következő csomagok kerültek telepítésre:
    - **firebase/php-jwt:** JSON Web Token (JWT) kezeléshez.
    - **phpmailer/phpmailer:** E-mail küldési funkciók megvalósításához.

2.2. Szerver- és Adatbázis Konfiguráció

- **Web Szerver:** Apache futtatja a PHP alkalmazást.
- **Adatbázis:** MySQL, melyben a rendszer a felhasználói és foglalási adatokat tárolja.
  A config.php fájl felel az adatbázis-kapcsolat létrehozásáért, melyben a karakterkészlet (utf8) miatt fontos a magyar ékezetek támogatása.

2.3. Composer és Függőségkezelés

A projekt két fő csomagja:

- **firebase/php-jwt (v6.11.1):** A JWT alapú autentikáció megvalósításához.
- **phpmailer/phpmailer (v6.9.3):** Regisztrációs folyamat során a megerősítő e-mail küldéséhez.

---

3. Rendszer Architektúra

3.1. Áttekintés

A rendszer rétegezett architektúrával készült:

- **Frontend réteg:** HTML, CSS, JavaScript alapú felhasználói felület, mely dinamikusan állítja össze a navigációs menüt a felhasználói jogosultságok alapján.
- **Backend réteg:** PHP kód, mely kezeli a regisztrációt, bejelentkezést, időpontfoglalást, adminisztrációs műveleteket, valamint a JWT tokenek kezelését.
- **Adatbázis réteg:** Az adatbázis-kapcsolatot és CRUD műveleteket megvalósító modul, mely a config.php fájlban kerül definiálásra.

3.2. Rétegek: Frontend, Backend, Adatbázis

- **Frontend:**
    - Dinamikus navigációs menü, amely a felhasználói token alapján változtatja a menü elemeit (pl. bejelentkezés, kijelentkezés, admin panel, időpontfoglalás).
    - Időpontfoglalási űrlap, ahol a felhasználók választanak dátumot és időpontot.
- **Backend:**
    - Autentikáció és jogosultság-kezelés JWT token segítségével.
    - Bejelentkezési és regisztrációs folyamat, mely során a jelszavak biztonságos hash-elése és ellenőrzése történik.
    - Adminisztrációs felület, mely lehetővé teszi a foglalások listázását, törlését.
    - REST API végpontok (például get_appointments.php, save_appointment.php), melyek JSON formátumú válaszokat adnak.
- **Adatbázis:**
    - Táblák: users (felhasználói adatok), appointments (foglalási adatok).
    - Kapcsolódás és SQL lekérdezések a MySQLi interfészen keresztül.

3.3. Moduláris Felépítés és Adatáramlás

A rendszer moduláris felépítése lehetővé teszi:

- Egyszerű kódkarbantartást és bővítést.
- A felhasználói űrlapokon beadott adatok validálását, továbbítását JSON formában a backend felé, ahol az adatbázis műveletek és értesítések történnek.
- Dinamikus menügenerálást a token ellenőrzése alapján, így az admin és a normál felhasználók számára különböző lehetőségek jelennek meg.

---

4. Fő Funkcionális Modulok

4.1. Autentikáció és Jogosultságkezelés (JWT)

- **JWT token létrehozása és érvényesítése:**
  A bejelentkezési folyamatban a felhasználó adatai (id, email, role) alapján generálódik a JWT token, melyet a cookie-ban tárol a rendszer. Kódrészlet (login.php részlet):

```php
$payload = [
    'id' => $id,
    'email' => $email,
    'role' => $role,
    'exp' => time() + 3600
];
$jwt = JWT::encode($payload, $key, 'HS256');
setcookie('token', $jwt, time() + 3600, "/");
```

- **Token ellenőrzés minden oldalon:**
  A különböző oldalakon (például admin.php, appointment.php) a cookie-ból beolvasott token alapján kerül validálásra a felhasználó jogosultsága.

4.2. Felhasználói Regisztráció és Bejelentkezés

- **Regisztrációs folyamat:**
  A register.php fájlban megvalósított folyamat ellenőrzi a megadott jelszavak egyezését, a jelszó erősségét (reguláris kifejezés segítségével), majd hash-eli a jelszót, és beszúrja az adatbázisba. Emellett PHPMailer segítségével megerősítő e-mailt küld.
- **Bejelentkezési folyamat:**
  A login.php feldolgozza az űrlap adatait, ellenőrzi a felhasználói létezést, és a jelszó ellenőrzése után a felhasználói adatok alapján generálja a JWT token-t.

4.3. Időpontfoglalás Kezelése

- **Űrlap a foglaláshoz:**
  Az appointment.php oldal kliensoldali űrlapja lehetővé teszi a felhasználó számára, hogy ne csak nevet, hanem a foglalni kívánt dátumot és időpontot is megadja.
- **Időpontválasztó logika:**
  JavaScript alapú megoldás, amely a megadott dátum alapján AJAX segítséggel lekéri az adott napra már foglalt időpontokat (a get_appointments.php végpontról), majd ennek alapján generálja a félórás bontású, elérhető időpontokat.
- **Foglalási adatok mentése:**
  A save_appointment.php PHP szkript ellenőrzi a POST kérés adatait, validálja az időpontot (csak hétköznap, 09:00 és 16:00 között, félórás bontással), majd ellenőrzi az ütközéseket az adatbázisban. Amennyiben az időpont szabad, beszúrja a foglalást, és JSON válaszban visszaigazolást ad.

4.4. Adminisztrációs Felület – Foglalások Kezelése

- **Admin jogosultság ellenőrzése:**
  Az admin.php és a hozzá tartozó törlési szkript (delete_appointment.php) ellenőrzik a JWT tokenben szereplő szerepet, és csak az admin jogosultság esetén engedélyezik a hozzáférést.
- **Foglalások listázása és törlése:**
  Az admin panel egy táblázatos formában jeleníti meg a foglalásokat (név, dátum, időpont), valamint biztosít egy törlés gombot, amely POST kéréssel hívja a törlési műveletet.

4.5. E-mail Értesítési Rendszer (PHPMailer)

- **Megerősítő e-mail küldés regisztrációkor:**
  A regisztráció feldolgozása során a PHPMailer küld egy egyszerű HTML alapú e-mailt a felhasználónak, amely tájékoztatja a sikeres regisztrációról.
- **SMTP Konfiguráció:**
  A kódban példaként a Gmail SMTP szerver paraméterei szerepelnek, melyek segítségével a rendszer biztonságosan küldi az e-mail értesítéseket.

---

5. Frontend Felület és Kliensek

5.1. Navigációs Menü Dinamikus Összeállítása

- Az egyes oldalak elején futó PHP kód a JWT token alapján határozza meg, hogy mely menüpontok jelenjenek meg:
  - Ha nincs token, az alapértelmezett „Bejelentkezés/Regisztráció” link jelenik meg.
  - Token esetén, a felhasználó szerepétől függően megjelenik az „Időpontfoglalás”, illetve az „Admin” menüpont, illetve a „Kijelentkezés”.

```php
if (isset($_COOKIE['token'])) {
    try {
        $decoded = JWT::decode($_COOKIE['token'], new Key($key, 'HS256'));
        $appointmentMenu = '<li class="nav-item"><a class="nav-link" href="appointment.php">Időpontfoglalás</a></li>';
        if ($decoded->role === 'admin') {
            $adminMenu = '<li class="nav-item"><a class="nav-link" href="admin.php">Admin</a></li>';
        }
        $authMenu = '<li class="nav-item"><a class="nav-link" href="logout.php">Kijelentkezés</a></li>';
    } catch (Exception $e) {
        // Hibás token esetén marad a bejelentkezés link
    }
}
```

5.2. Időpontválasztó és Foglalási Űrlap

- **Űrlap elemek:**
  A foglalási űrlapon szerepel név, dátum, valamint időpont kiválasztása. A dátum változás eseményére JavaScript kód frissíti az elérhető időpontokat.
- **Validáció és AJAX:**
  A JavaScript kód AJAX segítsével lekéri a már foglalt időpontokat, és frissíti az időpontválasztó opcióit.

5.3. Responsive Design és Bootstrap Használata

- A rendszer felhasználói felülete Bootstrap segítségével készült, így mobilbarát, reszponzív megjelenést biztosít.
- Egyéni CSS szabályokkal (pl. árnyékolás, színek, háttér transzparencia) finomítottuk a megjelenést.

---

6. Adatbázis Modell és Konfiguráció

6.1. Kapcsolódás és Karakterkészlet Beállítása

- A config.php felelős az adatbázis kapcsolódásért:

```php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "barbershop";

$conn = new mysqli($servername, $username, $password, $dbname);
$conn->set_charset("utf8"); // FONTOS: magyar ékezetek miatt
if ($conn->connect_error) {
    die("Kapcsolati hiba: " . $conn->connect_error);
}
```

6.2. Táblák: users és appointments

- **users:**
  Tartalmazza a regisztrált felhasználók adatait (id, email, password, role stb.).
- **appointments:**
  Tartalmazza az időpontfoglalási adatokat (customer_name, appointment_date, appointment_time, user_id stb.).

6.3. CRUD Műveletek Implementációja

- Az adatbázis műveletek a MySQLi interfészen keresztül történnek, előkészített utasításokkal (prepared statements) a biztonság érdekében.
- Példák:
  - **Foglalás beszúrása:** INSERT utasítás a save_appointment.php fájlban.
  - **Foglalások lekérdezése:** SELECT utasítás a get_appointments.php végpontban.
  - **Foglalás törlése:** DELETE utasítás az adminisztrációs törlésnél.

---

7. Biztonsági Intézkedések és Hibakezelés

7.1. JWT Token Ellenőrzés és Érvényesítés

- Minden olyan oldal, amely jogosultság-ellenőrzést igényel (admin panel, foglalási űrlap, API végpontok) ellenőrzi a cookie-ban tárolt token érvényességét.
- Hibás vagy hiányzó token esetén a rendszer a bejelentkezési oldalra irányít.

7.2. SQL Injection Elleni Védelem

- Minden adatbázis művelet előkészített utasításokkal történik, így elkerülhető az SQL Injection támadás.

7.3. Hibakezelési Stratégiák és Logolás

- A kódban minden kritikus pont try/catch blokkokat alkalmaz, így az érvénytelen tokenek, adatbázis hiba vagy egyéb problémák megfelelően lekezelhetők.
- A felhasználók számára érthető hibaüzenetek jelennek meg, míg a részletes hibaleírásokat a fejlesztők naplózhatják.

---

8. Tesztelési Forgatókönyvek és Validáció

8.1. Manuális tesztek

- Kézi funkcionális teszteket végeztünk minden egyes rendszerfunkción (pl. jelszóellenőrzés, foglalási logika, admin jogosultságok) a helyes működés igazolására.
- Kézzel ellenőriztük az AJAX-hívások, adatbázis-műveletek és felhasználói űrlapok végső működését a tényleges felhasználói élmény biztosításához.

8.2. Felhasználói Visszajelzések és Hibajelentések

- A rendszer a felhasználókat hibás adatbevitel vagy nem elérhető időpont esetén értesíti, így biztosítva a felhasználói élmény folyamatos javítását.

---

9. Fejlesztési és Karbantartási Irányok

9.1. Verziókövetés és Dokumentáció Frissítése

- A forráskód Git verziókövetést használ, részletes commit üzenetekkel.
- A dokumentáció folyamatosan frissül a kód változásainak megfelelően.

9.2. Jövőbeni Bővítési Lehetőségek

- Felhasználói profilkezelés bővítése.
- Mobilalkalmazás integráció.
- További biztonsági intézkedések (például kétfaktoros autentikáció).

---

10. Összegzés és Függelék

10.1. Kódkimenetek Listázata

A melléklet tartalmazza az összes elkészített PHP fájl tartalmát, többek között:

- **config.php:** Adatbázis kapcsolat és charset beállítás.
- **JWT alapú autentikációs modulok:** login.php, register.php, logout.php.
- **Foglalás kezelési modulok:** appointment.php, save_appointment.php, get_appointments.php, admin.php, delete_appointment.php.
- **Frontend fájlok:** HTML sablonok, CSS és JavaScript kódrészletek.

10.2. Konfigurációs Fájlok

- **composer.json, composer.lock:** A függőségkezelés részletes leírása.
- **config.yaml vagy config.php:** Az adatbázis és API paraméterek beállítása.

10.3. Külső Források, Referenciák

- Firebase PHP-JWT GitHub
- PHPMailer GitHub