

JEGYZŐKÖNYV

Szakmai Gyakorlat I.

Frontend – Backend integrációs szakasz

Készítette: Orosz Kristóf

Neptunkód: EYZWG9

Dátum: 2025. június 24.

.....
Orosz Kristóf

Hallgató

.....
Dédesi Péter

Szakmai konzulens

Tartalomjegyzék:

Bevezetés.....	3
Fájlok tartalmának leírása.....	3
1. Feladatrészek	4
1.1 Regisztrációs funkció működése	4
1.2 Bejelentkezési funkció működése	9
1.3 Szolgáltatók funkció működése.....	13
1.4 Időpontok funkció működése	31
1.5 Értékelések funkció működése	50
1.6 Fiókom funkció működése	62

Bevezetés

A szakmai gyakorlat Frontend–Backend integrációs szakaszában összehangoltam a projektem frontend és backend oldalát. A dokumentum első részében bemutatom a projekt felépítését, valamint azokat a fájlokat, melyek az egyes funkciók megvalósításához készültek. A következő alfejezetekben sorra veszem a regisztráció, bejelentkezés, szolgáltatók kezelés, időpontok rögzítése, értékelések kezelése és a “Fiókom” felület működését. Minden feladatrésznél részletesen dokumentálom a backend PHP forráskód logikáját, az adatbázisműveleteket és a frontend űrlapok szerkezetét. A leírás tartalmazza az adatvalidálás és hibakezelés lépéseit, valamint a visszaküldött státuszüzenetek formátumát. A felhasználói élményt minimalista, középre igazított, reszponzív űrlapok segítik, melyek egyszerű CSS-sel és meta-beállításokkal készültek.

Fájlok tartalmának leírása

- | | |
|---|-------------------------------------|
| - Bevezetés.pdf | [A mappa tartalmának leírása.] |
| - Regisztrációs funkció működése.pdf | [PDF fájl az alábbi feladatrésről.] |
| - Bejelentkezési funkció működése.pdf | [PDF fájl az alábbi feladatrésről.] |
| - Szolgáltatók funkció működése.pdf | [PDF fájl az alábbi feladatrésről.] |
| - Időpontok funkció működése.pdf | [PDF fájl az alábbi feladatrésről.] |
| - Értékelések funkció működése.pdf | [PDF fájl az alábbi feladatrésről.] |
| - Fiókom funkció működése.pdf | [PDF fájl az alábbi feladatrésről.] |
| - Jk-Frontend – Backend integrációs szakasz | [Jegyzőkönyv a szakasról.] |

Feladatrészek

Az alábbi pontokban röviden összefoglalom, hogy a Frontend-Backend integrációs szakasz során az egyes feladatrészekhez milyen konkrét tevékenységeket végeztem.

1.1 Regisztrációs funkció működése

A feladatrészben bemutatom, hogy a regisztrációs funkció a kapcsolat.php betöltésével hoz létre adatbázis-kapcsolatot, és kizárólag POST kéréseket fogad el, JSON formátumú válasszal. A beérkező nyers JSON-t dekódolom, majd ellenőrzöm, hogy a nev, email, jelszo és szerepkor mezők jelen vannak-e, és hogy a szerepkor csak „felhasznalo” vagy „admin” lehet. Megtisztítom az értékeket, és validálom, hogy egyik mező sem üres, különben 400-as hibakódot küldök vissza. PDO segítségével lekérdezem az adatbázist az email cím létezése miatt, és ha már regisztrálva van, 409-es hibával jelzem a konfliktust. Amennyiben az email cím még nem szerepel, beszúrom az új felhasználót, és sikeres beszúrás esetén 201 Created státusszal JSON-ban küldöm vissza a „Sikeres regisztracio” üzenetet és az új felhasználó azonosítóját. A feladathoz tartozó űrlap középre igazított, fehér hátterű, minimális CSS-sel és reszponzív meta-beállításokkal készült, és a felhasználótól kéri be a szükséges adatokat. A beépített JavaScript eseménykezelő AJAX hívással továbbítja az űrlapadatokat a regisztracio.php végpontra, majd a kapott eredménytől függően értesítem a felhasználót, és siker esetén átirányítom a bejelentkezési oldalra.

PHP forrásfájl

```
<?php
// Betöltöm az adatbázis-kapcsolatot létrehozó fájlt.
require_once 'kapcsolat.php';

// Beállítom, hogy JSON-t adjak vissza.
header('Content-Type: application/json');

// Csak a POST kéréseket fogadom el.
if ($_SERVER['REQUEST_METHOD'] !== 'POST') {
```

```

    http_response_code(405);
    echo json_encode(['error' => 'Csak POST metódussal érhető el ez a végpont.']);
    exit;
}

// A bejövő nyers JSON-t dekódolom tömbbé.
$input = json_decode(file_get_contents('php://input'), true);

// Ellenőrzöm, hogy minden szükséges mező létezik-e, és a szerepkor érvényes-e.
if (
    !isset($input['nev'], $input['email'], $input['jelszo'], $input['szerepkor'])
    ||
    !in_array($input['szerepkor'], ['felhasznalo', 'admin'], true)
) {
    http_response_code(400);
    echo json_encode(['error' => 'Nem sikerült értelmezni a JSON bemenetet, vagy
hiányzó/érvénytelen mező.']);
    exit;
}

// A bemenetből kiolvasom és megtisztítom a stringeket.
$nev = trim($input['nev']);
$email = trim($input['email']);
$jelszo = trim($input['jelszo']);
$szerepkor = $input['szerepkor'];

// Ellenőrzöm, hogy a név, email és jelszó nem üres.
if ($nev === '' || $email === '' || $jelszo === '') {
    http_response_code(400);
    echo json_encode(['error' => 'A név, email és jelszó mezők nem lehetnek
üresek.']);
    exit;
}

try {
    // Lekérdezem, hogy létezik-e már felhasználó ezzel az email címmel.
    $stmt = $pdo->prepare('SELECT COUNT(*) FROM felhasznalok WHERE email =
:email');
    $stmt->bindParam(':email', $email, PDO::PARAM_STR);
    $stmt->execute();
    if ($stmt->fetchColumn() > 0) {
        http_response_code(409);
        echo json_encode(['error' => 'Ez az email már regisztrálva van.']);
        exit;
    }
} catch (PDOException $e) {
    // Ha adatbázis-hiba történik.
    http_response_code(500);
}

```

HTML forrásfájl

6

```
}

.container {
  display: flex;
  justify-content: center;
  align-items: center;
  height: calc(100vh - 70px);
}

.form-box {
  background-color: white;
  padding: 30px;
  border-radius: 8px;
  width: 100%;
  max-width: 400px;
  box-shadow: 0 0 10px rgba(0,0,0,0.1);
}

.form-box h2 {
  text-align: center;
  margin-top: 0;
}

.form-box input,
.form-box select {
  width: 100%;
  padding: 10px;
  margin: 10px 0;
  border-radius: 4px;
  border: 1px solid #ccc;
}

.form-box button {
  width: 100%;
  padding: 10px;
  background-color: #007BFF;
  color: white;
  border: none;
  border-radius: 4px;
  cursor: pointer;
}

.form-box button:hover {
  background-color: #0056b3;
}

.login-link {
  margin-top: 10px;
```

```

        text-align: left;
    }

    .login-link a {
        color: #007BFF;
        text-decoration: none;
        font-size: 14px;
    }

    .login-link a:hover {
        text-decoration: underline;
    }
</style>
</head>
<body>
    <!-- Fejléc és navigáció -->
    <header>
        <a href="index.html" class="logo">Időpontfoglaló <span id="felhasznalo-
nev"></span></a>
        <nav>
            <a href="szolgaltatok.html">Szolgáltatók</a>
            <a href="idopontok.html">Időpontok</a>
            <a href="ertekelesek.html">Értékelések</a>
            <a href="bejelentkezes.html"><b>Bejelentkezés</b></a>
        </nav>
    </header>

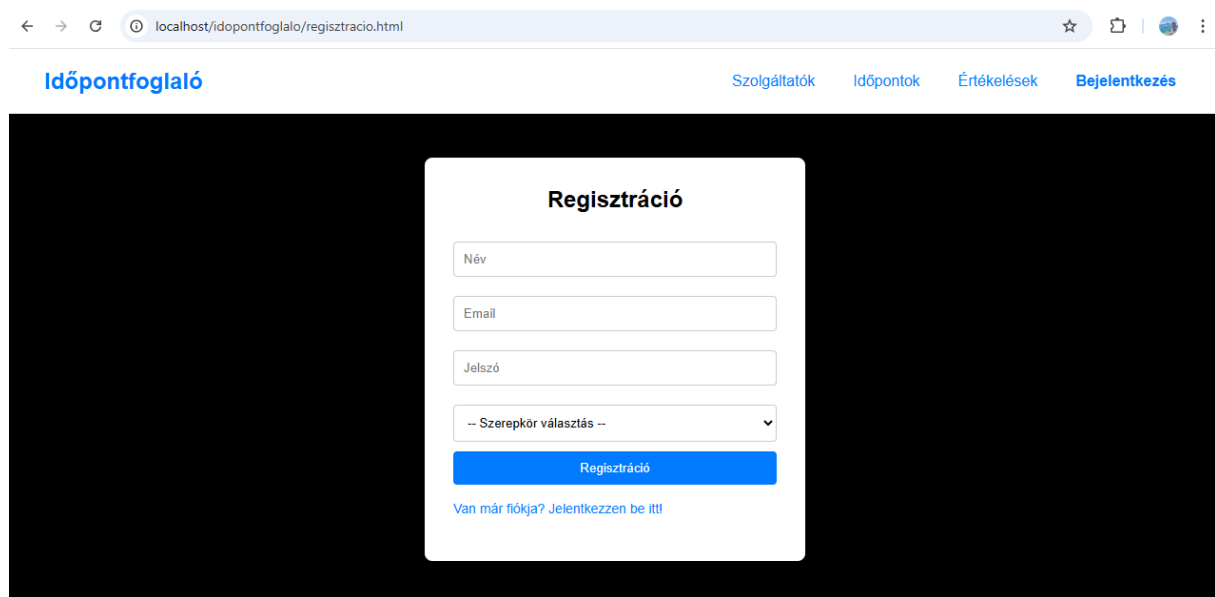
    <!-- Regisztrációs űrlap -->
    <div class="container">
        <div class="form-box">
            <h2>Regisztráció</h2>
            <form id="regForm">
                <!-- Név -->
                <input type="text" name="nev" placeholder="Név" required />
                <!-- Email -->
                <input type="email" name="email" placeholder="Email" required />
                <!-- Jelszó -->
                <input type="password" name="jelszo" placeholder="Jelszó" required />
                <!-- Szerepkör -->
                <select name="szerepko" required>
                    <option value="">-- Szerepkör választás --</option>
                    <option value="felhasznalo">Felhasználó</option>
                    <option value="admin">Admin</option>
                </select>
                <!-- Küldés -->
                <button type="submit">Regisztráció</button>
            </form>
            <div class="login-link">

```



```
        <p><a href="bejelentkezes.html">Van már fiókja? Jelentkezzen be  
itt!</a></p>  
    </div>  
</div>  
</div>  
  
<script>  
    // Form elküldése AJAX-szal  
    const form = document.getElementById('regForm');  
    form.addEventListener('submit', async (e) => {  
        e.preventDefault();  
        const data = {  
            nev: form.nev.value,  
            email: form.email.value,  
            jelszo: form.jelszo.value,  
            szerepkor: form.szerepkor.value  
        };  
        // Regisztrációs kérés  
        const response = await fetch('regisztracio.php', {  
            method: 'POST',  
            headers: { 'Content-Type': 'application/json' },  
            body: JSON.stringify(data)  
        });  
        const result = await response.json();  
        if (response.ok) {  
            alert('Sikeres regisztráció! Most már bejelentkezhetsz!');  
            window.location.href = 'bejelentkezes.html';  
        } else {  
            alert(result.error || 'Hiba történt a regisztráció során.');        }  
    });  
</script>  
</body>  
</html>
```

HTML kimenet



The screenshot shows a web browser window with the address bar displaying 'localhost/idopontfoglalo/registrazcio.html'. The page has a dark blue header with the site name 'Időpontfoglaló' and navigation links: 'Szolgáltatók', 'Időpontok', 'Értékelések', and 'Bejelentkezés'. The main content area is a white registration form titled 'Regisztráció' centered on a dark background. The form contains four input fields: 'Név', 'Email', 'Jelszó', and a dropdown menu for role selection labeled '-- Szerepkör választás --'. Below these is a blue 'Regisztráció' button and a link that says 'Van már fiókja? Jelentkezzen be itt!'

1.2 Bejelentkezési funkció működése

Feladatrészben betöltöm a kapcsolatot létrehozó kapcsolat.php fájlt, és csak POST kéréseket fogadok el, JSON választ szolgáltatva. Dekódolom a beérkező JSON-t, meggyőződöm róla, hogy asszociatív tömb, és ellenőrzöm, hogy az email és jelszo mezők nem hiányoznak; hibás bemenet esetén 400-as vagy 405-ös kóddal kilépek. Kiolvasom az email és jelszo értékét, majd PDO-val lekérdezem az adatbázisból a megfelelő felhasználót. Ha a felhasználó nem létezik, vagy a jelszó nem egyezik, 401-es hibát küldök vissza. Sikeres egyezéskor JSON-ban visszaküldöm a „Sikeres bejelentkezés!” státuszt és a felhasználó id, nev, szerepkör illetve email mezőit. A felületen egy középre igazított, fehér háttérű, minimális CSS-sel készült HTML űrlapot jelenítek meg, ahol az email és jelszó beviteli mezők érhetők el. A JavaScript eseménykezelő AJAX hívással továbbítja az űrlapadatokat a bejelentkezés.php végpontra, majd a kapott válasz alapján értesítem a felhasználót, és siker esetén átirányítom a megfelelő oldalra.

PHP forrásfájl

```
<?php
// Betöltöm a kapcsolatot létrehozó fájlt, hogy használhassam a $pdo változót.
require_once "kapcsolat.php";

// A válasz tartalmát JSON formátumra állítom.
header('Content-Type: application/json');

// Csak a POST kéréseket fogadom el.
if ($_SERVER['REQUEST_METHOD'] !== 'POST') {
    http_response_code(405);
    echo json_encode(['error' => 'Csak POST metódussal érhető el ez a végpont.']);
    exit;
}

// A bejövő nyers JSON-t dekódolom asszociatív tömbbé.
```

```
$input = json_decode(file_get_contents('php://input'), true);

// Ellenőrzöm, hogy sikerült-e dekódolni és valóban tömb-e.
if (!is_array($input)) {
    http_response_code(400);
    echo json_encode(['error' => 'Nem sikerült értelmezni a JSON bemenetet.']);
    exit;
}

// Ellenőrzöm, hogy a kötelező mezők (email, jelszo) megvannak-e, és nem üresek-e.
if (empty($input['email']) || empty($input['jelszo'])) {
    http_response_code(400);
    echo json_encode(['error' => 'Hiányzó email vagy jelszó.']);
    exit;
}

// Kiolvasom és elmented a változókba a bejövő emailt és jelszót.
$email = $input['email'];
$jelszo = $input['jelszo'];

// Lekérdezem az adatbázisból a felhasználót az email alapján.
$stmt = $pdo->prepare('SELECT id, nev, szerekor, jelszo FROM felhasznalok WHERE email = :email');
$stmt->execute([':email' => $email]);
$user = $stmt->fetch();

if (!$user || $user['jelszo'] !== $jelszo) {
    http_response_code(401);
    echo json_encode(['error' => 'Érvénytelen email vagy jelszó.']);
    exit;
}

// Ha minden rendben van, visszaküldöm a felhasználó adatait JSON-ban.
echo json_encode([
    'status' => 'Sikeres bejelentkezés!',
    'user' => [
        'id' => (int)$user['id'],
        'nev' => $user['nev'],
        'szerekor' => $user['szerekor'],
        'email' => $email
    ]
]);
```

HTML forrásfájl

```
<!DOCTYPE html>
<html lang="hu">
```

```
<head>
  <!-- Dokumentum típus és nyelv beállítása -->
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <!-- Külső stíluslapok betöltése -->
  <link rel="stylesheet" href="menu.css">
  <link rel="stylesheet" href="style.css">
  <link rel="stylesheet" href="hatter.css">
  <!-- Oldal címe -->
  <title>Bejelentkezés</title>
</head>
<body>
  <!-- Fejléc és navigáció -->
  <header>
    <a href="index.html" class="logo">Időpontfoglaló</a>
    <nav>
      <a href="szolgaltatok.html">Szolgáltatók</a>
      <a href="idopontok.html">Időpontok</a>
      <a href="ertekelesek.html">Értékelések</a>
      <a href="bejelentkezes.html"><b>Bejelentkezés</b></a>
    </nav>
  </header>

  <!-- Bejelentkezési űrlap központi doboza -->
  <div class="container">
    <div class="login-box">
      <h2>Bejelentkezés</h2>
      <form id="loginForm">
        <!-- Email mező -->
        <input type="email" name="email" placeholder="Email" required />
        <!-- Jelszó mező -->
        <input type="password" name="jelszo" placeholder="Jelszó" required />
        <!-- Bejelentkezés gomb -->
        <button type="submit">Bejelentkezés</button>
      </form>
      <div class="register-link">
        <!-- Regisztrációs link -->
        <p><a href="regisztracio.html">Nincs fiókja? Regisztráljon itt!</a></p>
      </div>
    </div>
  </div>

  <!-- JavaScript az űrlap kezeléséhez -->
  <script>
    const form = document.getElementById('loginForm');
    form.addEventListener('submit', async function (e) {
      e.preventDefault();
      // Beviteli adatok gyűjtése
```

```
const email = form.email.value;
const jelszo = form.jelszo.value;

// Kérés a bejelentkezési végpontra
const response = await fetch('bejelentkezés.php', {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify({ email, jelszo })
});

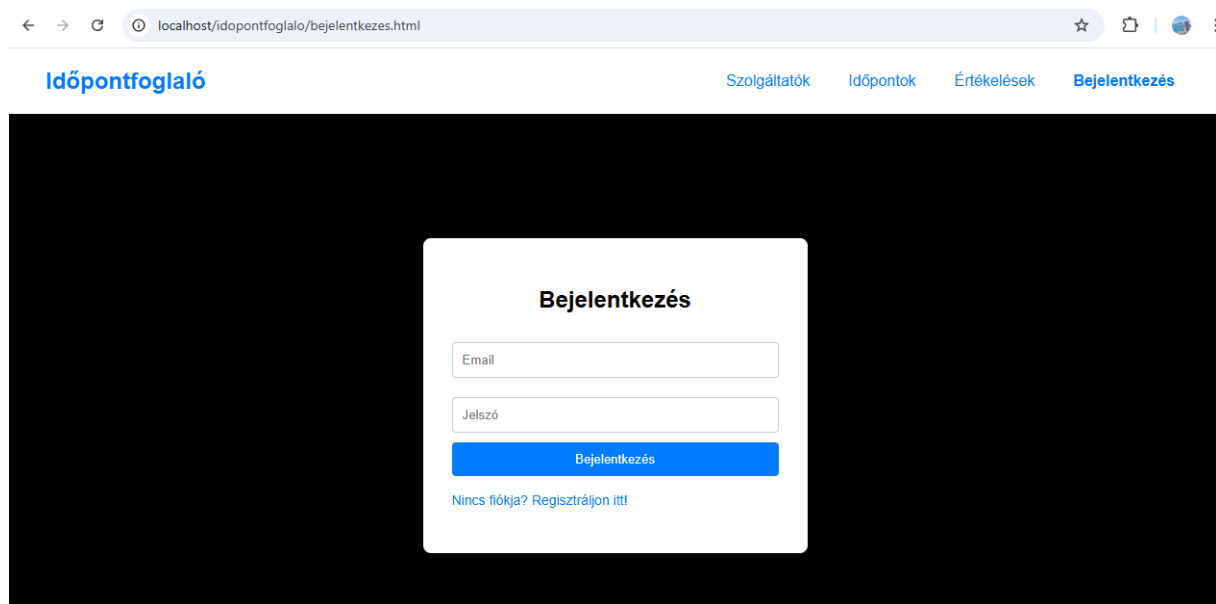
const data = await response.json();
if (response.ok) {
  // Sikeres bejelentkezés: felhasználó mentése és átirányítás
  sessionStorage.setItem('felhasznalo', JSON.stringify(data.user));
  if (data.user.szerepkor === 'admin') {
    window.location.href = 'admin.html';
  } else {
    window.location.href = 'felhasznalo.html';
  }
} else {
  // Hiba kezelése
  alert(data.error || 'Hiba történt a bejelentkezés során.');
```

```

}
});
</script>

</body>
</html>
```

HTML kimenet



1.3 Szolgáltatók funkció működése

Feladatrészben betöltöm a PDO-kapcsolatot létrehozó kapcsolat.php fájlt, és kizárólag POST metódussal fogadom a JSON kéréseket, egyéb esetben 405-ös hibakóddal kilépek. A bejövő JSON-t asszociatív tömbbé dekódolom, ellenőrzöm, hogy tartalmazza a nev, leiras, szolgáltatás_típusok_id és aktív mezőket, és hogy ezek érvényes értékek-e, hibás bemenetnél 400-as választ adok. Trimmereléssel megtisztítom az értékeket, validálom, hogy a név és a leírás nem üres, a típusazonosító pozitív, majd PDO-val ellenőrzöm, hogy a megadott szolgáltatás-típus ténylegesen létezik; ha nem, szintén 400-as hibát küldök. Ha minden feltétel teljesül, INSERT utasítással létrehozom az új szolgáltatót, és sikeres beszúrás esetén 201 Created státusszal JSON-ben küldöm vissza a státuszt és az új azonosítót; adatbázis-hiba esetén 500-as választ adok. Módosításkor PUT metódust várok az URL-ben szereplő id paraméterrel, ellenőrzöm a rekord létezését, dinamikusan összeállított UPDATE utasítással frissítem a megadott mezőket, és 200-as státusszal jelezve a sikeres módosítást válaszolok. Törlésnél DELETE kérést fogadok, előbb ellenőrzöm az id létezését, majd végrehajtom a DELETE utasítást, 200-as kóddal visszaadva a törölt rekord azonosítóját, vagy 404/500 hibával jelezve, ha valami nem stimmel. A felhasználói felületen középre igazított, fehér háttérű, minimális CSS-sel kialakított HTML űrlapok (létrehozás, módosítás, törlés) kérik be az adatokat, melyeket beépített JavaScript AJAX hívásokkal továbbítok a megfelelő PHP végpontokhoz, majd az eredménytől függően értesítem és átirányítom a felhasználót.

PHP forrásfájlok

Szolgáltatók létrehozása

```
<?php
// Betöltöm a PDO-kapcsolatot létrehozó fájlt, hogy legyen $pdo objektumom.
require_once "kapcsolat.php";

// Beállítom, hogy a válasz mindig JSON formátumú legyen.
header('Content-Type: application/json');

// Ellenőrzöm, hogy a kliens csak POST metódussal érhesse el ezt a végpontot.
// Ha más metódus érkezik, 405-ös hibával kilépek.
if ($_SERVER['REQUEST_METHOD'] !== 'POST') {
    http_response_code(405);
    echo json_encode(['error' => 'Csak POST metódussal érhető el ez a végpont.']);
    exit;
}

// Beolvasom a bejövő JSON-t, és asszociatív tömbbé alakítom.
$input = json_decode(file_get_contents('php://input'), true);

// Ellenőrzöm, hogy a JSON egy tömb legyen, és tartalmazza a 'nev', 'leiras',
// 'szolgaltatas_tipusok_id' kulcsokat.
// Az 'aktiv' mezőnek pedig vagy 0-nak, vagy 1-nek kell lennie.
if (
    !is_array($input) ||
    !isset($input['nev'], $input['leiras'], $input['szolgaltatas_tipusok_id']) ||
    (!isset($input['aktiv']) && $input['aktiv'] !== 0 && $input['aktiv'] !== 1)
) {
    http_response_code(400);
    echo json_encode(['error' => 'Hiányzó vagy érvénytelen mezők.']);
    exit;
}

// Kitisztítom és változóba mentem a bemenetet.
$nev = trim($input['nev']);
$leiras = trim($input['leiras']);
$szolgaltatas_tipusok_id = (int)$input['szolgaltatas_tipusok_id'];
$aktiv = (int)$input['aktiv'];

// Ellenőrzöm, hogy a név, leírás nem üres, és a típus-id pozitív szám-e.
if ($nev === '' || $leiras === '' || $szolgaltatas_tipusok_id <= 0) {
    http_response_code(400);
    echo json_encode(['error' => 'A név, leírás és szolgáltatástípus azonosító nem lehet üres.']);
    exit;
}

// Megnézem, hogy valóban létezik-e a megadott szolgáltatás-típus a táblában.
// Ha nem, visszadobom a 400-as hibát.
try {
```

```

    $stmtFK = $pdo->prepare('SELECT COUNT(*) FROM szolgaltatas_tipusok WHERE id =
:tid');
    $stmtFK->bindParam(':tid', $szolgaltatas_tipusok_id, PDO::PARAM_INT);
    $stmtFK->execute();
    if ($stmtFK->fetchColumn() == 0) {
        http_response_code(400);
        echo json_encode(['error' => 'A megadott szolgáltatás-típus nem
létezik.']);
        exit;
    }
} catch (PDOException $e) {
    // Ha a lekérdezés közben hiba történik, 500-as hibát küldök a részletes
üzenettel.
    http_response_code(500);
    echo json_encode(['error' => 'Adatbázis hiba (típus ellenőrzés): ' . $e-
->getMessage()]);
    exit;
}

// Most beszúrom az új szolgáltatót az adatbázisba.
try {
    $stmt = $pdo->prepare('
        INSERT INTO szolgaltatok (szolgaltatas_tipusok_id, nev, leiras, aktiv)
        VALUES (:tid, :nev, :leiras, :aktiv)
    ');
    $stmt->bindParam(':tid', $szolgaltatas_tipusok_id, PDO::PARAM_INT);
    $stmt->bindParam(':nev', $nev, PDO::PARAM_STR);
    $stmt->bindParam(':leiras', $leiras, PDO::PARAM_STR);
    $stmt->bindParam(':aktiv', $aktiv, PDO::PARAM_INT);
    $stmt->execute();

    // Sikeres beszúrás esetén kiolvasom az új rekord AUTO_INCREMENT értékét.
    $newId = $pdo->lastInsertId();
    http_response_code(201);
    echo json_encode([
        'status' => 'Szolgáltató sikeresen létrehozva',
        'szolgaltato_id' => (int)$newId
    ]);
    exit;
} catch (PDOException $e) {
    // Ha az INSERT során adatbázis-hiba történik, 500-as hibakódot küldök vissza.
    http_response_code(500);
    echo json_encode(['error' => 'Adatbázis hiba (szolgáltató létrehozása): ' .
    $e->getMessage()]);
    exit;
}

```


Szolgáltatók módosítása

```
<?php
// Betöltöm a kapcsolat.php fájlt, hogy a $pdo változó elérhető legyen az
adatbázis-műveletekhez.
require_once "kapcsolat.php";

// Beállítom, hogy minden válasz JSON formátumban érkezzen a kliens felé.
header('Content-Type: application/json');

// Ellenőrzöm, hogy valóban PUT metódussal hívják-e meg ezt a végpontot.
if ($_SERVER['REQUEST_METHOD'] !== 'PUT') {
    // Ha nem PUT, 405-ös (Method Not Allowed) HTTP státuszkódot küldök
    http_response_code(405);
    echo json_encode(['error' => 'Csak PUT metódussal érhető el ez a végpont.']);
    exit;
}

// Ellenőrzöm, hogy szerepel-e az URL-ben az "id" paraméter, és az csak számokat
tartalmaz-e.
if (!isset($_GET['id']) || !ctype_digit($_GET['id'])) {
    // Ha nincs érvényes id, 400-as (Bad Request) státuszt küldök
    http_response_code(400);
    echo json_encode(['error' => 'Hiányzó vagy érvénytelen szolgáltató-id.']);
    exit;
}

// Átalakítom az id-t integerre.
$szolgáltato_id = (int)$_GET['id'];

// A beérkező JSON-t beolvasom és asszociatív tömbbé alakítom.
$input = json_decode(file_get_contents('php://input'), true);
if (!is_array($input)) {
    // Ha nem tudtam tömbbé dekódolni a JSON-t, 400-as hibát adok vissza
    http_response_code(400);
    echo json_encode(['error' => 'Nem sikerült értelmezni a JSON bemenetet.']);
    exit;
}

try {
    // Lekérdezem, hogy létezik-e ilyen "id"-jű rekord a szolgálatok táblában.
    $stmtCheck = $pdo->prepare('SELECT COUNT(*) FROM szolgálatok WHERE id =
:id');
    $stmtCheck->bindParam(':id', $szolgáltato_id, PDO::PARAM_INT);
    $stmtCheck->execute();
    if ($stmtCheck->fetchColumn() == 0) {
        // Ha nem létezik, 404-es (Not Found) hibát küldök.
        http_response_code(404);
        echo json_encode(['error' => 'A megadott szolgáltató nem található.']);
        exit;
    }
}
```

```

    }
} catch (PDOException $e) {
    // Ha adatbázis hiba történik a létezés-ellenőrzés során, 500-as hibát adok
    vissza.
    http_response_code(500);
    echo json_encode(['error' => 'Adatbázis hiba (létezés ellenőrzés): ' . $e-
    >getMessage()]);
    exit;
}

// Összeállítom, mely mezőket (oszlopokat) szeretnék frissíteni, és az ezekhez
tartozó paramétereket.
$fieldsToUpdate = [];
$params = [ ':id' => $szolgáltato_id ];

// Ha a bejövő JSON tartalmaz "nev"-et, akkor hozzáadom a frissítendő mezők közé.
if (isset($input['nev'])) {
    $fieldsToUpdate[] = 'nev = :nev';
    $params[':nev'] = trim($input['nev']);
}
// Ha tartalmaz "leiras"-t, hozzáadom.
if (isset($input['leiras'])) {
    $fieldsToUpdate[] = 'leiras = :leiras';
    $params[':leiras'] = trim($input['leiras']);
}
// Ha tartalmaz "szolgaltatas_tipusok_id"-t, hozzáadom és integer típusra cast-
olom.
if (isset($input['szolgaltatas_tipusok_id'])) {
    $fieldsToUpdate[] = 'szolgaltatas_tipusok_id = :tid';
    $params[':tid'] = (int)$input['szolgaltatas_tipusok_id'];
}
// Ha tartalmaz "aktiv"-ot, hozzáadom és integerre cast-olom.
if (isset($input['aktiv'])) {
    $fieldsToUpdate[] = 'aktiv = :aktiv';
    $params[':aktiv'] = (int)$input['aktiv'];
}

// Ha nem volt egyetlen frissítendő mező sem, 400-as hibát küldök.
if (empty($fieldsToUpdate)) {
    http_response_code(400);
    echo json_encode(['error' => 'Nincs megadva egyetlen módosítandó mező sem.']);
    exit;
}

// Ha a szolgaltatas_tipusok_id mezőt is adják, előbb ellenőrzöm, hogy az id
létezik-e a szolgaltatas_tipusok táblában.
if (isset($params[':tid'])) {
    try {

```

```

        $stmtFK = $pdo->prepare('SELECT COUNT(*) FROM szolgaltatas_tipusok WHERE
id = :tid');
        $stmtFK->bindParam(':tid', $params[':tid'], PDO::PARAM_INT);
        $stmtFK->execute();
        if ($stmtFK->fetchColumn() == 0) {
            // Ha nincs ilyen típus, 400-as hibát küldök.
            http_response_code(400);
            echo json_encode(['error' => 'A megadott (új) szolgáltatás-típus nem
létezik.']);
            exit;
        }
    } catch (PDOException $e) {
        // Ha adatbázis hiba adódik a típus ellenőrzésénél, 500-as hibát küldök.
        http_response_code(500);
        echo json_encode(['error' => 'Adatbázis hiba (típus ellenőrzés): ' . $e-
>getMessage()]);
        exit;
    }
}

// Összerakom az UPDATE SQL utasítást a kiválasztott mezőkkel.
$sql = 'UPDATE szolgaltatok SET ' . implode(' ', $fieldsToUpdate) . ' WHERE id =
:id';

try {
    // Előkészítem és végrehajtom a frissítést a paraméterekkel.
    $stmt = $pdo->prepare($sql);
    $stmt->execute($params);

    // Ha sikeres a módosítás, 200-as státuszt és egy JSON-feleletet küldök.
    http_response_code(200);
    echo json_encode(['status' => 'Szolgáltató sikeresen módosítva']);
    exit;
} catch (PDOException $e) {
    // Ha adatbázis hiba történik a frissítéskor, 500-as hibát küldök.
    http_response_code(500);
    echo json_encode(['error' => 'Adatbázis hiba (szolgáltató módosítása): ' . $e-
>getMessage()]);
    exit;
}

```

Szolgáltatók törlése

```
<?php
```

```
// Betöltöm a PDO-kapcsolatot kezelő fájlt, hogy legyen $pdo objektumom.
require_once "kapcsolat.php";

// Beállítom, hogy a válasz JSON formátumban érkezzon a kliens felé.
header('Content-Type: application/json');

// Ellenőrzöm, hogy a kérés csak DELETE metódussal érhető el.
// Ha más metódust látok, 405-ös hibával kilépek.
if ($_SERVER['REQUEST_METHOD'] !== 'DELETE') {
    http_response_code(405);
    echo json_encode(['error' => 'Csak DELETE metódussal érhető el ez a
végpont.']);
    exit;
}

// Lekérem az id-t a query stringből, és ellenőrzöm, hogy számjegyekből áll-e.
// Ha hiányzik vagy nem digitális, 400-as hibát küldök.
if (empty($_GET['id']) || !ctype_digit($_GET['id'])) {
    http_response_code(400);
    echo json_encode(['error' => 'Hiányzó vagy érvénytelen id.']);
    exit;
}

$id = (int)$_GET['id']; // Ellenőrzöm, hogy egész szám az id.

// Először ellenőrzöm, létezik-e a megadott id-val szolgáltató.
// Ha nincs ilyen sor a táblában, 404-es hibát adok vissza.
try {
    $stmtChk = $pdo->prepare('SELECT COUNT(*) FROM szolgáltatok WHERE id = :id');
    $stmtChk->execute([':id' => $id]);
    if ($stmtChk->fetchColumn() == 0) {
        http_response_code(404);
        echo json_encode(['error' => 'A megadott id-val nem található
szolgáltató.']);
        exit;
    }
} catch (PDOException $e) {
    // Ha az ellenőrzés közben adatbázis-hiba történik, 500-as hibával válaszolok.
    http_response_code(500);
    echo json_encode(['error' => 'Adatbázis hiba (ellenőrzés): ' . $e-
>getMessage()]);
    exit;
}

// Ha tényleg létezik az id, végrehajtom a DELETE utasítást.
try {
    $stmt = $pdo->prepare('DELETE FROM szolgáltatok WHERE id = :id');
    $stmt->execute([':id' => $id]);
}
```

```
// Sikeres törlés esetén 200-as státusszal visszaküldöm az eltávolított id-t.
http_response_code(200);
echo json_encode([
    'status' => 'Sikeres törlés',
    'deleted_id' => $id
]);
exit;
} catch (PDOException $e) {
    // Ha a törlés közben hiba történik, 500-as hibával jelzem a részleteket.
    http_response_code(500);
    echo json_encode(['error' => 'Adatbázis hiba (delete): ' . $e->getMessage()]);
    exit;
}
```

HTML forrásfájlok

Szolgáltatók létrehozása

```
<!DOCTYPE html>
<html lang="hu">
<head>
    <!-- Dokumentum típus és nyelv -->
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <!-- Oldal címe -->
    <title>Szolgáltató létrehozása</title>
    <!-- Menü stíluslap -->
    <link rel="stylesheet" href="menu.css">
    <style>
        /* Alapértelmezett test és háttér */
        body {
            font-family: Arial, sans-serif;
            background-color: #f4f4f4;
            margin: 0;
            padding: 0;
        }
        /* Űrlap konténer */
        form {
            max-width: 400px;
            margin: 40px auto;
            padding: 20px;
            background: white;
            border-radius: 10px;
            box-shadow: 0 2px 8px rgba(0,0,0,0.1);
        }
        /* Űrlap mezők és gomb */
        form input,
```

```
form select,
form button {
  width: 100%;
  margin: 10px 0;
  padding: 10px;
  font-size: 16px;
  border-radius: 5px;
  border: 1px solid #ccc;
}
/* Gomb színezése és kurzor */
form button {
  background-color: #007BFF;
  color: white;
  border: none;
  cursor: pointer;
}
form button:hover {
  background-color: #0056b3;
}
/* Segélykategória stílusok */
.color {
  color: red;
}
/* Rejtett elem osztály */
.hidden {
  display: none !important;
}
/* Mobilra reagáló stílus */
@media (max-width: 500px) {
  form {
    margin: 20px 10px;
    padding: 15px;
  }
  form input,
  form select,
  form button {
    font-size: 15px;
    padding: 9px;
  }
  h2 {
    font-size: 20px;
    padding: 0 10px;
  }
}
</style>
</head>
<body>
  <!-- Fejléc és navigáció -->
```

```

<header>
  <a href="index.html" class="logo">Időpontfoglaló <span id="felhasznalo-
nev"></span></a>
  <nav>
    <a href="szolgaltatok.html">Szolgáltatók</a>
    <a href="idopontok.html">Időpontok</a>
    <a href="ertekelesek.html">Értékelések</a>
    <a href="bejelentkezes.html" id="nav-login"><b>Bejelentkezés</b></a>
    <a href="fiokom.html" id="nav-account" class="hidden">Fiókom</a>
    <a href="#" id="nav-logout" onclick="kijelentkezes()" class="hidden"><span
class="color">Kijelentkezés</span></a>
  </nav>
</header>

<!-- Címsor a form felett -->
<h2 style="text-align: center;">Szolgáltató létrehozása</h2>
<!-- Szolgáltató létrehozása űrlap -->
<form id="letrehozForm">
  <input type="text" name="nev" placeholder="Szolgáltató neve" required />
  <input type="text" name="leiras" placeholder="Leírás" required />
  <input type="number" name="szolgaltatas_tipusok_id" placeholder="Típus ID"
required />
  <select name="aktiv" required>
    <option value="">-- Aktivitás kiválasztása --</option>
    <option value="1">Aktív</option>
    <option value="0">Inaktív</option>
  </select>
  <button type="submit">Létrehozás</button>
</form>

<!-- AJAX kód az űrlap elküldéséhez -->
<script>
const form = document.getElementById('letrehozForm');
form.addEventListener('submit', async (e) => {
  e.preventDefault();
  const data = Object.fromEntries(new FormData(form));
  try {
    const response = await fetch('Szolgaltatok_letrehozasa.php', {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify(data)
    });
    const result = await response.json();
    alert(result.status || result.error);
    if (response.ok) {
      window.location.href = 'szolgaltatok.html';
    }
  } catch (err) {

```

```
        alert('Hiba történt: ' + err.message);
    }
});
</script>

<!-- Kijelentkezés funkció -->
<script>
    function kijelentkezés() {
        sessionStorage.removeItem('felhasznalo');
        window.location.href = 'index.html';
    }
</script>

<!-- Felhasználó adatok beállítása a navigációhoz -->
<script>
    const user = JSON.parse(sessionStorage.getItem('felhasznalo'));
    if (user && user.nev) {
        document.getElementById('felhasznalo-nev').textContent = ` - ${user.nev}`;
        document.getElementById('nav-login').classList.add('hidden');
        document.getElementById('nav-account').classList.remove('hidden');
        document.getElementById('nav-logout').classList.remove('hidden');
    }
</script>
</body>
</html>
```

Szolgáltatók módosítása

```
<!DOCTYPE html>
<html lang="hu">
<head>
    <!-- Dokumentum típusa és nyelv beállítása -->
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <!-- Oldal címe -->
    <title>Szolgáltató módosítása</title>
    <!-- Menü stíluslap betöltése -->
    <link rel="stylesheet" href="menu.css">
    <style>
        /* Oldal alapstílusai */
        body {
            font-family: Arial, sans-serif;
            background-color: #f4f4f4;
            margin: 0;
            padding: 0;
        }
        /* Űrlap stílusai */
        form {
```



```
max-width: 400px;
margin: 40px auto;
padding: 20px;
background: white;
border-radius: 10px;
box-shadow: 0 2px 8px rgba(0,0,0,0.1);
}
/* Űrlap mezők és gomb */
form input,
form select,
form button {
  width: 100%;
  margin: 10px 0;
  padding: 10px;
  font-size: 16px;
  border-radius: 5px;
  border: 1px solid #ccc;
}
/* Gomb színeztése */
form button {
  background-color: #007BFF;
  color: white;
  border: none;
  cursor: pointer;
}
form button:hover {
  background-color: #0056b3;
}
/* Cím középre igazítása */
.center {
  text-align: center;
}
/* Kijelentkezés piros színnel */
.color {
  color: red;
}
/* Rejtett elemek elrejtése */
.hidden {
  display: none !important;
}
/* Mobil nézet esetén */
@media (max-width: 500px) {
  form {
    margin: 20px 10px;
    padding: 15px;
  }
  form input,
  form select,
```

```

        form button {
            font-size: 15px;
            padding: 9px;
        }
        h2.center {
            font-size: 20px;
            padding: 0 10px;
        }
    }
</style>
</head>
<body>
    <!-- Fejléc és navigáció -->
    <header>
        <a href="index.html" class="logo">Időpontfoglaló <span id="felhasznalo-
nev"></span></a>
        <nav>
            <a href="szolgaltatok.html">Szolgáltatók</a>
            <a href="idopontok.html">Időpontok</a>
            <a href="ertekelesek.html">Értékelések</a>
            <a href="bejelentkezes.html" id="nav-login"><b>Bejelentkezés</b></a>
            <a href="fiokom.html" id="nav-account" class="hidden">Fiókom</a>
            <a href="#" id="nav-logout" onclick="kijelentkezes()" class="hidden"><span
class="color">Kijelentkezés</span></a>
        </nav>
    </header>

    <!-- Címsor a form felett -->
    <h2 class="center">Szolgáltató módosítása</h2>

    <!-- Szolgáltató adatainak módosító űrlapja -->
    <form id="modositForm">
        <!-- Szolgáltató neve -->
        <input type="text" name="nev" placeholder="Szolgáltató neve" required />
        <!-- Leírás -->
        <input type="text" name="leiras" placeholder="Leírás" required />
        <!-- Típus ID -->
        <input type="number" name="szolgaltatas_tipusok_id" placeholder="Típus ID"
required />
        <!-- Aktivitás kiválasztása -->
        <select name="aktiv">
            <option value="1">Aktív</option>
            <option value="0">Inaktív</option>
        </select>
        <!-- Mentés gomb -->
        <button type="submit">Módosítás mentése</button>
    </form>

```

```
<script>
  // URL paraméterből ID kinyerése
  const urlParams = new URLSearchParams(window.location.search);
  const id = urlParams.get('id');
  if (!id) {
    alert("Hiányzó szolgáltató ID.");
    window.location.href = "szolgaltatok.html";
  }

  // Űrlap feltöltése meglévő adatokkal
  fetch("Szolgaltatok_listazasa.php")
    .then(res => res.json())
    .then(data => {
      const szolg = data.data.find(s => s.id == id);
      if (!szolg) {
        alert("Érvénytelen szolgáltató ID.");
        return;
      }
      const form = document.getElementById('modositForm');
      form.nev.value = szolg.nev;
      form.leiras.value = szolg.leiras;
      form.szolgaltatas_tipusok_id.value = szolg.szolgaltatas_tipusok_id || "";
      form.aktiv.value = szolg.aktiv;
    });

  // PUT kérés küldése módosításhoz
  const form = document.getElementById('modositForm');
  form.addEventListener('submit', async (e) => {
    e.preventDefault();
    const data = Object.fromEntries(new FormData(form));
    const response = await fetch(`Szolgaltatok_modositasa.php?id=${id}`, {
      method: 'PUT',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify(data)
    });
    const result = await response.json();
    alert(result.status || result.error);
  });

  // Kijelentkezés funkció
  function kijelentkezés() {
    sessionStorage.removeItem('felhasznalo');
    window.location.href = "index.html";
  }

  // Navigáció frissítése bejelentkezett felhasználóval
  const user = JSON.parse(sessionStorage.getItem('felhasznalo'));
  if (user && user.nev) {
```

```
        document.getElementById('felhasznalo-nev').textContent = ` - ${user.nev}`;  
        document.getElementById('nav-login').classList.add('hidden');  
        document.getElementById('nav-account').classList.remove('hidden');  
        document.getElementById('nav-logout').classList.remove('hidden');  
    }  
</script>  
</body>  
</html>
```

Szolgáltatók törlése

```
<!DOCTYPE html>  
<html lang="hu">  
<head>  
    <!-- Dokumentum típusa és nyelv beállítása -->  
    <meta charset="UTF-8" />  
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />  
    <!-- Oldal címe -->  
    <title>Szolgáltató törlése</title>  
    <!-- Menü stíluslap betöltése -->  
    <link rel="stylesheet" href="menu.css" />  
    <style>  
        /* Oldal alapstílusai */  
        body {  
            font-family: Arial, sans-serif;  
            background-color: #f4f4f4;  
            margin: 0;  
            padding: 0;  
        }  
        /* Űrlap doboz stílusai */  
        form {  
            max-width: 400px;  
            margin: 40px auto;  
            background: white;  
            padding: 20px;  
            border-radius: 8px;  
            box-shadow: 0 0 8px rgba(0,0,0,0.1);  
        }  
        /* Beviteli mezők és gombok */  
        input,  
        button {  
            width: 95%;  
            margin: 10px 0;  
            padding: 10px;  
            border-radius: 4px;  
            font-size: 16px;  
            border: 1px solid #ccc;  
        }  
    </style>  
</head>  
<body>  
    <div>  
        <h2>Szolgáltató törlése</h2>  
        <div>  
            <div>  
                <input type="text" value="Név" />  
                <input type="text" value="Cím" />  
                <input type="text" value="Telefonszám" />  
            </div>  
            <div>  
                <input type="button" value="Töröl" />  
            </div>  
        </div>  
    </div>  
</body>  
</html>
```

```

/* Gomb stílus */
button {
    background-color: blue;
    color: white;
    border: none;
    cursor: pointer;
    text-align: center;
}
button:hover {
    background-color: blue;
}
/* Figyelmeztető szöveg piros színnel */
.color {
    color: red;
    text-align: center;
}
/* Normál címszöveg */
.color2 {
    color: black;
    text-align: center;
}
/* Rejtett elemek */
.hidden {
    display: none !important;
}
</style>
</head>
<body>
    <!-- Fejléc és navigáció -->
    <header>
        <a href="index.html" class="logo">Időpontfoglaló <span id="felhasznalo-
nev"></span></a>
        <nav>
            <a href="szolgaltatok.html">Szolgáltatók</a>
            <a href="idopontok.html">Időpontok</a>
            <a href="ertekelesek.html">Értékelések</a>
            <a href="bejelentkezes.html" id="nav-login"><b>Bejelentkezés</b></a>
            <a href="fiokom.html" id="nav-account" class="hidden">Fiókom</a>
            <a href="#" id="nav-logout" onclick="kijelentkezes()" class="hidden"><span
class="color">Kijelentkezés</span></a>
        </nav>
    </header>

    <!-- Oldalcím -->
    <h2 class="color2">Szolgáltató törlése</h2>

    <!-- Törlő űrlap -->
    <form id="torlesForm">

```

```
<!-- Szolgáltató ID bevitele -->
<input type="number" name="id" placeholder="Szolgáltató ID" required />
<!-- Törlés gomb -->
<button type="submit">Törlés</button>
</form>

<script>
  // Felhasználó adatainak ellenőrzése
  const user = JSON.parse(sessionStorage.getItem('felhasznalo'));
  if (user && user.nev) {
    document.getElementById('felhasznalo-nev').textContent = ` - ${user.nev}`;
  }

  // Csak adminok érhetik el az oldalt
  if (!user || user.szerepkor !== 'admin') {
    alert('Csak bejelentkezett admin használhatja ezt az oldalt.');
```

window.location.href = 'bejelentkezes.html';

```
  }

  // Törlés feldolgozása
  const form = document.getElementById('torlesForm');
  form.addEventListener('submit', async (e) => {
    e.preventDefault();
    const id = form.id.value;
    if (!confirm(`Biztosan törölni szeretné a(z) ${id}-es azonosítójú
szolgáltatót?`)) return;

    const response = await fetch(`Szolgaltatok_torlese.php?id=${id}`, { method:
'DELETE' });
    const result = await response.json();
    alert(result.status || result.error);
    if (response.ok) window.location.href = 'szolgaltatok.html';
  });

  // Kijelentkezés funkció
  function kijelentkezes() {
    sessionStorage.removeItem('felhasznalo');
    window.location.href = 'index.html';
  }

  // Navigációs elemek frissítése
  if (user && user.nev) {
    document.getElementById('nav-login').classList.add('hidden');
    document.getElementById('nav-account').classList.remove('hidden');
    document.getElementById('nav-logout').classList.remove('hidden');
  }
</script>
</body>
```

</html>

HTML kimenetek

← → ↻

localhost/idopontfoglalo/Szolgaltatok_letrehozasa.html

☆ 📄 ⬇️ 🌐 ⋮

Időpontfoglaló - Kiss Dalma

Szolgáltatók Időpontok Értékelések Fiókom Kijelentkezés

Szolgáltató létrehozása

-- Aktivitás kiválasztása --

▼

Létrehozás

← → ↻

localhost/idopontfoglalo/Szolgaltatok_modositasa.html?id=3

☆ 📄 ⬇️ 🌐 ⋮

Időpontfoglaló - Kiss Dalma

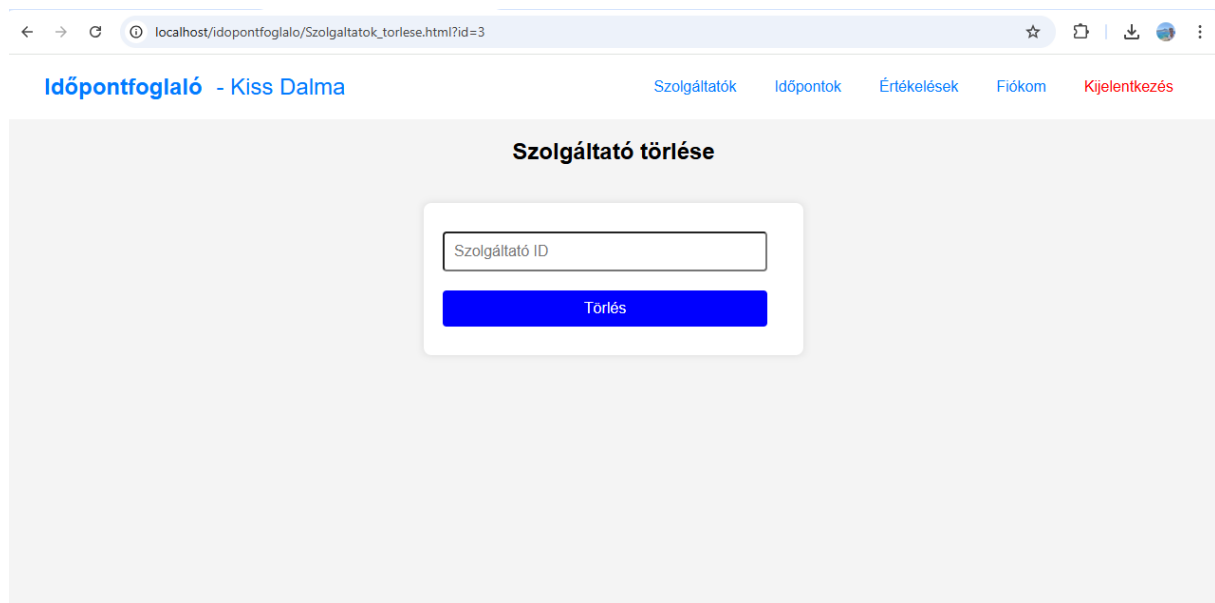
Szolgáltatók Időpontok Értékelések Fiókom Kijelentkezés

Szolgáltató módosítása

Aktív

▼

Módosítás mentése



1.4 Időpontok funkció működése

Feladatrészben betöltöm a kapcsolat.php fájlt, és kizárólag POST metódust fogadok el, JSON formátumú választ szolgáltatva. Dekódolom a beérkező JSON-t, ellenőrzöm, hogy tartalmazza a szolgaltatok_id, datum és ido mezőket, és hiány esetén 400-as hibakóddal kilépek. A mezőket trimeléssel és típuskonverzióval előkészítem, a foglalhato érték hiányakor alapértelmezetten 1-et állítok be. PDO-val beszúrom az új időpontot az idopontok táblába, és sikeres létrehozáskor 201 Created státusszal JSON-ben visszaküldöm az idopont_id-t, adatbázis-hiba esetén 500-as hibát adok

. Módosításhoz PUT kérést várok az URL-ben szereplő id paraméterrel, először ellenőrzöm a rekord létezését, majd dinamikusan frissítem a megadott mezőket és JSON-ben igazolom vissza a sikeres módosítást. Törléskor DELETE metódussal fogadom az id-t, létezés ellenőrzése után végrehajtom a törlést, és JSON-ben küldöm vissza a törlés eredményét. A felületen reszponzív, középre igazított HTML űrlapok kérik be az adatokat, amelyeket beépített JavaScript AJAX hívásokkal továbbítok, admin jogosultságot ellenőrizve, majd a válasz alapján értesítem és szükség szerint átirányítom a felhasználót.

PHP forrásfájlok:

Időpontok létrehozása

```
<?php
// Csatlakozom az adatbázishoz.
require_once "kapcsolat.php";

// Beállítom, hogy a válasz JSON formátumú legyen.
header('Content-Type: application/json');

// Csak POST metódus engedélyezett, más esetben 405 hibát küldök vissza.
if ($_SERVER['REQUEST_METHOD'] !== 'POST') {
```



```

    http_response_code(405);
    echo json_encode(['error' => 'Csak POST metódussal érhető el ez a végpont.']);
    exit;
}

// Beolvasom a JSON formátumú bemenetet és tömbbé alakítom.
$input = json_decode(file_get_contents('php://input'), true);

// Ellenőrzöm, hogy minden szükséges adat megérkezett-e.
if (
    !is_array($input) ||
    !isset($input['szolgaltatok_id'], $input['datum'], $input['ido'])
) {
    http_response_code(400);
    echo json_encode(['error' => 'Hiányzó mezők (szolgaltatok_id, datum, ido).']);
    exit;
}

// Az értékeket előkészítem a beszúráshoz.
$szId = (int)$input['szolgaltatok_id'];
$datum = trim($input['datum']);
$ido = trim($input['ido']);
$fogl = isset($input['foglalhato']) ? (int)$input['foglalhato'] : 1; // ha nincs
megadva, 1-et állítok be

try {
    // Elkészítem a SQL utasítást az új időpont beszúráshoz.
    $stmt = $pdo->prepare("
        INSERT INTO idopontok (szolgaltatok_id, datum, ido, foglalhato)
        VALUES (:szid, :datum, :ido, :foglalhato)
    ");

    // Lefuttatom a lekérdezést a megadott adatokkal.
    $stmt->execute([
        ':szid' => $szId,
        ':datum' => $datum,
        ':ido' => $ido,
        ':foglalhato' => $fogl
    ]);

    // Sikeres beszúrást esetén visszaküldöm az új rekord azonosítóját.
    http_response_code(201);
    echo json_encode([
        'status' => 'Időpont sikeresen létrehozva',
        'idopont_id' => (int)$pdo->lastInsertId()
    ]);
} catch (PDOException $e) {
    // Ha hiba történik, visszajelzést adok róla JSON formátumban.

```

```

    http_response_code(500);
    echo json_encode(['error' => 'Adatbázis hiba: ' . $e->getMessage()]);
}

```

Időpontok módosítása

```

<?php
// Betöltöm az adatbázis kapcsolatot.
require_once "kapcsolat.php";
header('Content-Type: application/json');

// Csak PUT metódust engedek, egyéb esetben hibát dobok.
if ($_SERVER['REQUEST_METHOD'] !== 'PUT') {
    http_response_code(405);
    echo json_encode(['error' => 'Csak PUT metódussal érhető el ez a végpont.']);
    exit;
}

// Ellenőrzöm, hogy kapok-e érvényes id-t GET paraméterként.
if (empty($_GET['id']) || !ctype_digit($_GET['id'])) {
    http_response_code(400);
    echo json_encode(['error' => 'Hiányzó vagy érvénytelen id.']);
    exit;
}
$id = (int)$_GET['id']; // Az ID számot egész számként tárolom.

// Beolvasom a JSON-bemenetet, és tömbbé alakítom.
$input = json_decode(file_get_contents('php://input'), true);
if (!is_array($input)) {
    http_response_code(400);
    echo json_encode(['error' => 'Nem sikerült értelmezni a JSON bemenetet.']);
    exit;
}

// Ellenőrzöm, hogy az adott ID-val létezik-e időpont az adatbázisban.
try {
    $stmtChk = $pdo->prepare('SELECT COUNT(*) FROM idopontok WHERE id = :id');
    $stmtChk->execute([':id' => $id]);
    if ($stmtChk->fetchColumn() == 0) {
        http_response_code(404);
        echo json_encode(['error' => 'Nincs ilyen id-jú időpont.']);
        exit;
    }
} catch (PDOException $e) {
    http_response_code(500);
    echo json_encode(['error' => 'Adatbázis hiba (ellenőrzés): ' . $e->getMessage()]);
    exit;
}

```

```

}

// Felépítem a frissítendő mezők listáját és az értékeket.
$fields = [];
$params = ['id' => $id];

if (isset($input['szolgaltatok_id'])) {
    $fields[] = 'szolgaltatok_id = :szid';
    $params[':szid'] = (int)$input['szolgaltatok_id'];
}
if (isset($input['datum'])) {
    $fields[] = 'datum = :datum';
    $params[':datum'] = trim($input['datum']);
}
if (isset($input['ido'])) {
    $fields[] = 'ido = :ido';
    $params[':ido'] = trim($input['ido']);
}
if (isset($input['foglalhato'])) {
    $fields[] = 'foglalhato = :fogl';
    $params[':fogl'] = (int)$input['foglalhato'];
}

// Ha egyetlen mező sincs megadva, nem végzek módosítást.
if (empty($fields)) {
    http_response_code(400);
    echo json_encode(['error' => 'Nincs módosítandó mező.']);
    exit;
}

// Összeállítom a dinamikus SQL lekérdezést.
$sql = 'UPDATE idopontok SET ' . implode(', ', $fields) . ' WHERE id = :id';

try {
    // Lefuttatom a frissítést az előkészített adatokkal.
    $stmt = $pdo->prepare($sql);
    $stmt->execute($params);
    echo json_encode(['status' => 'Időpont sikeresen módosítva']);
} catch (PDOException $e) {
    // Hiba esetén részletes hibaüzenetet küldök vissza.
    http_response_code(500);
    echo json_encode(['error' => 'Adatbázis hiba (update idopont): ' . $e->getMessage()]);
    exit;
}

```

Időpontok törlése

```
<?php
// Csatlakozom az adatbázishoz a külön fájlban tárolt kapcsolatot segítségével.
require_once "kapcsolat.php";

// Beállítom a válasz formátumát JSON-ra.
header('Content-Type: application/json');

// Ellenőrzöm, hogy a kérés metódusa valóban DELETE-e.
if ($_SERVER['REQUEST_METHOD'] !== 'DELETE') {
    http_response_code(405); // Nem engedélyezett metódus
    echo json_encode(['error' => 'Csak DELETE metódussal érhető el ez a
végpont.']);
    exit;
}

// Ellenőrzöm, hogy kaptam-e érvényes numerikus id paramétert.
if (empty($_GET['id']) || !ctype_digit($_GET['id'])) {
    http_response_code(400); // Rossz kérés
    echo json_encode(['error' => 'Hiányzó vagy érvénytelen id.']);
    exit;
}

$id = (int)$_GET['id']; // Az ID-t egész számmá alakítom.

try {
    // Lekérdezem, hogy létezik-e az adott ID-jú időpont.
    $stmtChk = $pdo->prepare('SELECT COUNT(*) FROM idopontok WHERE id = :id');
    $stmtChk->execute([':id' => $id]);
    if ($stmtChk->fetchColumn() == 0) {
        http_response_code(404); // Nem található.
        echo json_encode(['error' => 'Nincs ilyen id-jú időpont.']);
        exit;
    }
} catch (PDOException $e) {
    // Hiba történt az ellenőrzés közben.
    http_response_code(500);
    echo json_encode(['error' => 'Adatbázis hiba (ellenőrzés): ' . $e-
>getMessage()]);
    exit;
}

try {
    // Törölöm az időpontot az adatbázisból.
    $stmt = $pdo->prepare('DELETE FROM idopontok WHERE id = :id');
    $stmt->execute([':id' => $id]);

    // Visszajelzést adok, ha sikeres volt a törlés.
    echo json_encode(['status' => 'Időpont sikeresen törölve']);
}
```



```
/* Beviteli mezők és legördülő listák */
```

```
input,
```

```
select {
```

```
    width: 100%;
```

```
    padding: 10px;
```

```
    margin-top: 5px;
```

```
    border: 1px solid #ccc;
```

```
    border-radius: 4px;
```

```
}
```

```
/* Küldés gomb */
```

```
button {
```

```
    margin-top: 20px;
```

```
    width: 100%;
```

```
    padding: 12px;
```

```
    background-color: #28a745;
```

```
    color: white;
```

```
    border: none;
```

```
    border-radius: 5px;
```

```
    cursor: pointer;
```

```
}
```

```
button:hover {
```

```
    background-color: #218838;
```

```
}
```

```
/* Középre igazítás */
```

```
.center {
```

```
    text-align: center;
```

```
    color: black;
```

```
}
```

```
/* Figyelmeztető szöveg */
```

```
.color {
```

```
    color: red;
```

```
}
```

```
/* Rejtett elem */
```

```
.hidden {
```

```
    display: none !important;
```

```
}
```

```
/* Kisebb képernyő esetén */
```

```
@media (max-width: 768px) {
```

```
    .container {
```

```
        width: 90%;
```

```
        padding: 20px;
```

```
        margin-top: 20px;
```

```
    }
```

```
    h2 {
```

```
        font-size: 24px;
```

```
    }
```

```
    label,
```

```
    input,
```

```
    select,
    button {
        font-size: 15px;
    }
    button {
        padding: 10px;
    }
}
/* Mobil nézet */
@media (max-width: 480px) {
    h2 {
        font-size: 20px;
    }
    label,
    input,
    select,
    button {
        font-size: 14px;
    }
    button {
        padding: 10px;
    }
}
</style>
</head>
<body>
    <!-- Fejléc és navigáció -->
    <header>
        <a href="index.html" class="logo">Időpontfoglaló <span id="felhasznalo-
nev"></span></a>
        <nav>
            <a href="szolgaltatok.html">Szolgáltatók</a>
            <a href="idopontok.html">Időpontok</a>
            <a href="ertekelesek.html">Értékelések</a>
            <a href="bejelentkezes.html" id="nav-login"><b>Bejelentkezés</b></a>
            <a href="fiokom.html" id="nav-account" class="hidden">Fiókom</a>
            <a href="#" id="nav-logout" onclick="kijelentkezes()" class="hidden"><span
class="color">Kijelentkezés</span></a>
        </nav>
    </header>

    <!-- Oldalcím a form felett -->
    <h2 class="center">Időpont létrehozása</h2>

    <!-- Űrlap konténer -->
    <div class="container">
        <form id="letrehozas-form">
            <!-- Szolgáltató kiválasztása -->
```

```
<label for="szid">Szolgáltató:</label>
<select id="szid" required>
  <option value="">Betöltés...</option>
</select>

<!-- Dátum kiválasztása -->
<label for="datum">Dátum:</label>
<input type="date" id="datum" required />

<!-- Idő kiválasztása -->
<label for="ido">Idő:</label>
<input type="time" id="ido" required />

<!-- Foglalhatóság jelölése -->
<label for="foglalhato">Foglalható:</label>
<select id="foglalhato">
  <option value="1">Igen</option>
  <option value="0">Nem</option>
</select>

<!-- Létrehozás gomb -->
<button type="submit">Létrehozás</button>
</form>
</div>

<script>
  // Ellenőrzöm, hogy admin-e a felhasználó
  const felhasznalo = JSON.parse(sessionStorage.getItem('felhasznalo'));
  if (!felhasznalo || felhasznalo.szerepkor !== 'admin') {
    alert('Csak admin jogosultsággal hozható létre időpont.');
```

window.location.href = 'index.html';

```
  }

  // Szolgáltatók betöltése a legördülő listába
  fetch('Szolgaltatok_listazasa.php')
    .then(res => res.json())
    .then(data => {
      if (data.status === 'siker') {
        const select = document.getElementById('szid');
        select.innerHTML = '';
        data.data.forEach(s => {
          const option = document.createElement('option');
          option.value = s.id;
          option.textContent = s.nev;
          select.appendChild(option);
        });
      }
    });
});
```



```
// Űrlap elküldése AJAX-szal
document.getElementById('letrehozas-form').addEventListener('submit',
function(e) {
    e.preventDefault();
    const payload = {
        szolgaltatok_id: parseInt(document.getElementById('szid').value),
        datum: document.getElementById('datum').value,
        ido: document.getElementById('ido').value,
        foglalhato: parseInt(document.getElementById('foglalhato').value)
    };
    fetch('Idopontok_letrehozasa.php', {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify(payload)
    })
    .then(res => res.json())
    .then(data => {
        if (data.status) {
            alert('Időpont sikeresen létrehozva.');
```

```
            window.location.href = 'idopontok.html';
        } else {
            alert('Hiba: ' + (data.error || 'Ismeretlen hiba'));
        }
    })
    .catch(err => {
        console.error(err);
        alert('Hiba történt a létrehozás során.');
```

```
    });
});

// Kijelentkezés funkció
function kijelentkezés() {
    sessionStorage.removeItem('felhasznalo');
    window.location.href = 'index.html';
}

// Navigáció frissítése bejelentkezési állapot szerint
const user = JSON.parse(sessionStorage.getItem('felhasznalo'));
if (user && user.nev) {
    document.getElementById('felhasznalo-nev').textContent = ` - ${user.nev}`;
    document.getElementById('nav-login').classList.add('hidden');
    document.getElementById('nav-account').classList.remove('hidden');
    document.getElementById('nav-logout').classList.remove('hidden');
}
</script>
</body>
</html>
```

Időpontok módosítása

```
<!DOCTYPE html>
<html lang="hu">
<head>
  <!-- Dokumentum típusa és nyelv beállítása -->
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <!-- Menü stíluslap betöltése -->
  <link rel="stylesheet" href="menu.css" />
  <!-- Oldal címe -->
  <title>Időpont módosítása</title>
  <style>
    /* Alapértelmezett test és háttér */
    body {
      font-family: Arial, sans-serif;
      background-color: #f5f5f5;
    }
    /* Űrlap konténer */
    .container {
      max-width: 500px;
      margin: 0 auto;
      background-color: white;
      padding: 30px;
      border-radius: 8px;
      box-shadow: 0 2px 8px rgba(0,0,0,0.1);
    }
    /* Oldalcím */
    h2 {
      text-align: center;
      color: black;
    }
    /* Űrlap címkék */
    label {
      display: block;
      margin-top: 15px;
      font-weight: bold;
    }
    /* Beviteli mezők és legördülő listák */
    input,
    select {
      width: 100%;
      padding: 10px;
      margin-top: 5px;
      border: 1px solid #ccc;
      border-radius: 4px;
    }
  </style>
</head>
<body>
  <div class="container">
    <h2>Időpont módosítás</h2>
    <div class="form">
      <div>
        <input type="text" value="<div class="form">
          <div>
            <input type="text" value="</div>
          </div>
        </div>
      </div>
    </div>
  </body>
</html>
```

```
/* Küldés gomb */
button {
  margin-top: 20px;
  width: 100%;
  padding: 12px;
  background-color: #007BFF;
  color: white;
  border: none;
  border-radius: 5px;
  cursor: pointer;
}
button:hover {
  background-color: #0056b3;
}
/* Középre igazítás */
.center {
  text-align: center;
  color: black;
}
/* Figyelmeztető szöveg */
.color {
  color: red;
}
/* Rejtett elem */
.hidden {
  display: none !important;
}
/* Kisebb képernyő esetén */
@media (max-width: 768px) {
  .container {
    width: 90%;
    padding: 20px;
    margin-top: 20px;
  }
  h2 {
    font-size: 24px;
  }
  label,
  input,
  select,
  button {
    font-size: 15px;
  }
  button {
    padding: 10px;
  }
}
/* Mobil nézet */
```

```
@media (max-width: 480px) {
  h2 {
    font-size: 20px;
  }
  label,
  input,
  select,
  button {
    font-size: 14px;
  }
  button {
    padding: 10px;
  }
}
</style>
</head>
<body>
  <!-- Fejléc és navigáció -->
  <header>
    <a href="index.html" class="logo">Időpontfoglaló <span id="felhasznalo-
nev"></span></a>
    <nav>
      <a href="szolgaltatok.html">Szolgáltatók</a>
      <a href="idopontok.html">Időpontok</a>
      <a href="ertekelesek.html">Értékelések</a>
      <a href="bejelentkezes.html" id="nav-login"><b>Bejelentkezés</b></a>
      <a href="fiokom.html" id="nav-account" class="hidden">Fiókom</a>
      <a href="#" id="nav-logout" onclick="kijelentkezes()" class="hidden"><span
class="color">Kijelentkezés</span></a>
    </nav>
  </header>

  <!-- Oldalcím a form felett -->
  <h2 class="center">Időpont módosítása</h2>

  <!-- Űrlap konténer -->
  <div class="container">
    <form id="modositas-form">
      <!-- Szolgáltató kiválasztása -->
      <label for="szid">Szolgáltató:</label>
      <select id="szid" required>
        <option value="">Betöltés...</option>
      </select>

      <!-- Dátum kiválasztása -->
      <label for="datum">Dátum:</label>
      <input type="date" id="datum" required />
    </form>
  </div>
</body>
</html>
```

```
<!-- Idő kiválasztása -->
<label for="ido">Idő:</label>
<input type="time" id="ido" required />

<!-- Foglalhatóság jelölése -->
<label for="foglalhato">Foglalható:</label>
<select id="foglalhato">
  <option value="1">Igen</option>
  <option value="0">Nem</option>
</select>

<!-- Mentés gomb -->
<button type="submit">Mentés</button>
</form>
</div>

<script>
  // Ellenőrzöm, hogy admin-e a felhasználó
  const felhasznalo = JSON.parse(sessionStorage.getItem('felhasznalo'));
  if (!felhasznalo || felhasznalo.szerepkor !== 'admin') {
    alert('Csak admin jogosultsággal módosítható időpont.');
    window.location.href = 'index.html';
  }

  // URL paraméterből ID kinyerése
  const urlParams = new URLSearchParams(window.location.search);
  const id = urlParams.get('id');
  if (!id) {
    alert('Hiányzó időpont azonosító.');
    window.location.href = 'idopontok.html';
  }

  // Szolgáltatók betöltése a legördülő listába
  fetch('Szolgaltatok_listazasa.php')
    .then(res => res.json())
    .then(data => {
      if (data.status === 'siker') {
        const select = document.getElementById('szid');
        select.innerHTML = '';
        data.data.forEach(s => {
          const option = document.createElement('option');
          option.value = s.id;
          option.textContent = s.nev;
          select.appendChild(option);
        });
      }
    });
  });
```

```
// Időpont adatok betöltése
fetch('idopontok_listazasa.php')
  .then(res => res.json())
  .then(data => {
    if (data.status === 'siker') {
      const idopont = data.data.find(i => i.id == id);
      if (!idopont) {
        alert('Nincs ilyen ID-jű időpont.');
```

window.location.href = 'idopontok.html';

return;

}

document.getElementById('datum').value = idopont.datum;

document.getElementById('ido').value = idopont.ido;

document.getElementById('foglalhato').value = idopont.foglalhato;

// Szolgáltató előválasztása

const setSzyd = () => {

const select = document.getElementById('szyd');

if (select.options.length > 0) {

select.value = idopont.szolgaltatok_id;

} else {

setTimeout(setSzyd, 100);

}

};

setSzyd();

}

});

// Űrlap elküldése AJAX-szal PUT metódussal

document.getElementById('modositas-form').addEventListener('submit',

function(e) {

e.preventDefault();

const payload = {

szolgaltatok_id: parseInt(document.getElementById('szyd').value),

datum: document.getElementById('datum').value,

ido: document.getElementById('ido').value,

foglalhato: parseInt(document.getElementById('foglalhato').value)

};

fetch(`Idopontok_modositasa.php?id=\${id}`, {

method: 'PUT',

headers: { 'Content-Type': 'application/json' },

body: JSON.stringify(payload)

})

.then(res => res.json())

.then(data => {

if (data.status) {

alert('Időpont sikeresen módosítva.');

window.location.href = 'idopontok.html';

```

        } else {
            alert('Hiba: ' + (data.error || 'Ismeretlen hiba'));
        }
    })
    .catch(err => {
        console.error(err);
        alert('Hiba történt a módosítás során.');
```

});

});

// Kijelentkezés funkció

```

function kijelentkezés() {
    sessionStorage.removeItem('felhasznalo');
    window.location.href = 'index.html';
}

// Navigáció frissítése bejelentkezési állapot szerint
const user = JSON.parse(sessionStorage.getItem('felhasznalo'));
if (user && user.nev) {
    document.getElementById('felhasznalo-nev').textContent = ` - ${user.nev}`;
    document.getElementById('nav-login').classList.add('hidden');
    document.getElementById('nav-account').classList.remove('hidden');
    document.getElementById('nav-logout').classList.remove('hidden');
}
</script>
</body>
</html>
```

Időpontok törlése

```

<!DOCTYPE html>
<html lang="hu">
<head>
    <!-- Dokumentum típusa és nyelv beállítása -->
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <!-- Menü stíluslap betöltése -->
    <link rel="stylesheet" href="menu.css" />
    <!-- Oldal címe -->
    <title>Időpontok törlése</title>
    <style>
        /* Oldal alapstílusai */
        body {
```

```
    font-family: Arial, sans-serif;
    background-color: #f5f5f5;
    margin: 0;
    padding: 0;
}
/* Űrlap konténer középre igazítása */
.container {
    max-width: 400px;
    margin: 30px auto;
    background-color: white;
    padding: 30px;
    border-radius: 10px;
    box-shadow: 0 2px 8px rgba(0,0,0,0.1);
    text-align: center;
}
/* Beviteli mező stílusa */
.container input {
    width: 100%;
    padding: 12px;
    margin-bottom: 20px;
    font-size: 16px;
    border: 1px solid #ccc;
    border-radius: 5px;
}
/* Gomb stílusa */
.container button {
    width: 100%;
    padding: 12px;
    font-size: 16px;
    color: white;
    background-color: #007BFF;
    border: none;
    border-radius: 5px;
    cursor: pointer;
}
.container button:hover {
    background-color: #0056b3;
}
/* Oldalcím */
h2 {
    text-align: center;
    color: black;
}

.color {
    color: red;
}
/* Rejtett elem */
```



```
.hidden {
  display: none !important;
}
/* Mobil nézet 768px alatt */
@media (max-width: 768px) {
  .container {
    width: 90%;
    padding: 20px;
  }
  h2 {
    font-size: 24px;
  }
  input,
  button {
    font-size: 15px;
  }
  button {
    padding: 10px;
  }
}
/* Mobil nézet 480px alatt */
@media (max-width: 480px) {
  h2 {
    font-size: 20px;
  }
  input,
  button {
    font-size: 14px;
  }
  button {
    padding: 9px;
  }
}
</style>
</head>
<body>
  <!-- Fejléc és navigáció -->
  <header>
    <a href="index.html" class="logo">Időpontfoglaló <span id="felhasznalo-
nev"></span></a>
    <nav>
      <a href="szolgaltatok.html">Szolgáltatók</a>
      <a href="idopontok.html">Időpontok</a>
      <a href="ertekelesek.html">Értékelések</a>
      <!-- Bejelentkezés/Fiók/Kijelentkezés linkek kezelése -->
      <a href="bejelentkezés.html" id="nav-login"><b>Bejelentkezés</b></a>
      <a href="fiokom.html" id="nav-account" class="hidden">Fiókom</a>
```

```

        <a href="#" id="nav-logout" onclick="kijelentkezes()" class="hidden"><span
class="color">Kijelentkezés</span></a>
    </nav>
</header>

<!-- Oldalcím -->
<h2>Időpont törlése</h2>

<!-- Törlő űrlap -->
<div class="container">
    <input type="number" id="torles-id" placeholder="Időpont ID" required />
    <button id="btn-delete">Törlés</button>
</div>

<script>
    // Törlés gomb eseménykezelője
    document.getElementById('btn-delete').addEventListener('click', () => {
        const id = document.getElementById('torles-id').value.trim();
        if (!id) {
            alert('Kérem, adja meg a törlendő ID-t.');
```

```

            return;
        }
        // Megerősítés
        if (!confirm(`Biztosan törölni szeretné a(z) ${id} ID-jű időpontot?`))
return;

```

```

        // DELETE kérés küldése
        fetch(`Idopontok_torlese.php?id=${id}`, { method: 'DELETE' })
            .then(res => res.json())
            .then(data => {
                alert(data.status || data.error);
                if (data.status) window.location.href = 'idopontok.html';
            })
            .catch(err => {
                console.error(err);
                alert('Hiba történt a törlés során.');
```

```

            });
        });

        // Kijelentkezés funkció
        function kijelentkezes() {
            sessionStorage.removeItem('felhasznalo');
            window.location.href = 'index.html';
        }

```

```

        // Navigáció frissítése bejelentkezési állapot szerint
        const user = JSON.parse(sessionStorage.getItem('felhasznalo'));
        if (user && user.nev) {

```

Szakmai Gyakorlat I. – Frontend – Backend integrációs szakasz

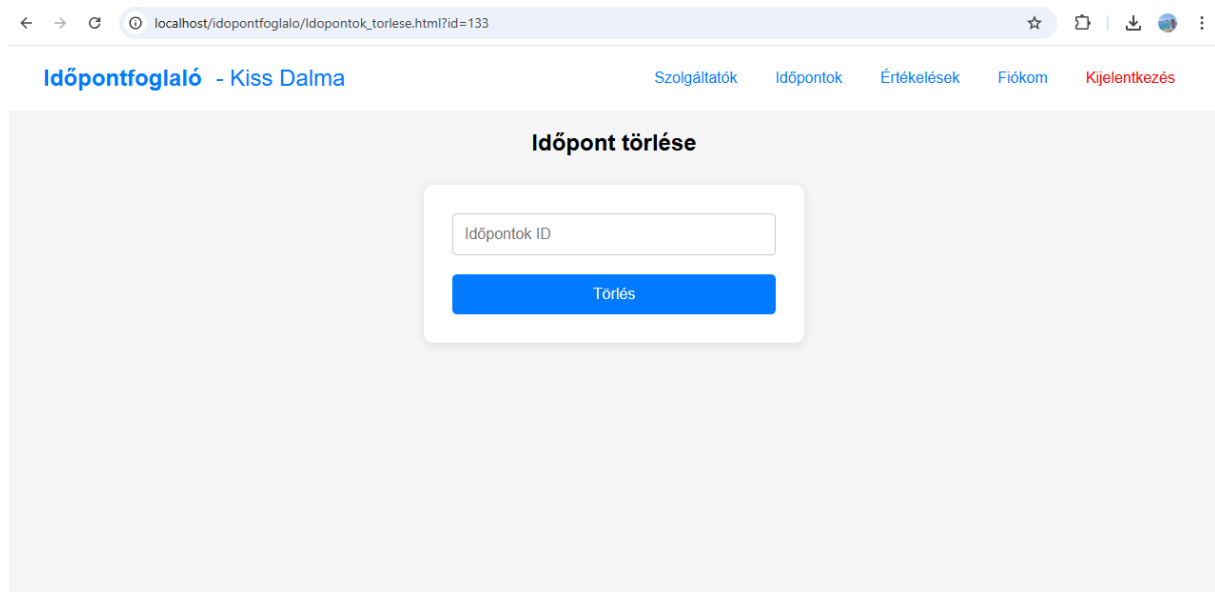
Orosz Kristóf – EYZWG9

```
document.getElementById('felhasznalo-nev').textContent = ` - ${user.nev}`;  
document.getElementById('nav-login').classList.add('hidden');  
document.getElementById('nav-account').classList.remove('hidden');  
document.getElementById('nav-logout').classList.remove('hidden');  
}  
</script>  
</body>  
</html>
```

HTML kimenetek:

The first screenshot shows the 'Időpontfoglaló - Kiss Dalma' web application. The main heading is 'Időpont létrehozása'. The form contains the following fields: 'Szolgáltató:' with a dropdown menu showing 'SmileDent'; 'Dátum:' with a date picker showing 'éééé . hh . nn .'; 'Idő:' with a time picker showing '-- : --'; 'Foglalható:' with a dropdown menu showing 'Igen'; and a green 'Létrehozás' button.

The second screenshot shows the 'Időpont módosítása' form. It contains the following fields: 'Szolgáltató:' with a dropdown menu showing 'Cardio4U Fitness'; 'Dátum:' with a date picker showing '2025 . 06 . 19 .'; 'Idő:' with a time picker showing '20 : 14'; 'Foglalható:' with a dropdown menu showing 'Igen'; and a blue 'Mentés' button.



1.5 Értékelések funkció működése

Feladatrészen betöltöm a kapcsolat.php fájlt, és csak POST módszerrel engedélyezem az értékelés létrehozását, JSON válasszal. Dekódolom a bejövő JSON-t, ellenőrzöm, hogy van-e foglalás_id és érvényes, 1–5 közötti értékes mező, különben 400-as hibakódot adok. A validált adatokat PDO-val beszúrom az értékelések táblába az aktuális dátummal, majd 201 Created státusszal visszaigazolom az „Értékelés rögzítve” státuszt. Törléskor csak DELETE módszert fogadok, ellenőrzöm az URL-ben érkező id paramétert, és ha hiányzik vagy érvénytelen, 400-as kódot küldök; majd a megfelelő id-jú rekordot törölöm, és visszaadom a „Értékelés törölve” státuszt. A frontend egy középre igazított, fehér háttérű táblázatot jelenít meg a meglévő értékelésekről, admin jogosultság esetén törlés gombbal. A JavaScript AJAX hívással lekérem az értékeléseket, dinamikusan generálom a sorokat, és a felhasználói szerepkör alapján jelenítem meg a törlési lehetőséget. Új értékelés írásakor a felhasználó az űrlapot kitöltve POST kérést indít, majd a válasz alapján értesítem és frissítem az oldalt.

PHP forrásfájlok

Értékelés létrehozása

```
<?php
// Kapcsolódok az adatbázishoz.
require_once "kapcsolat.php";
header('Content-Type: application/json');

// Csak POST módszerrel engedem elérni ezt a fájlt.
if ($_SERVER['REQUEST_METHOD'] !== 'POST') {
    http_response_code(405);
    echo json_encode(['error' => 'Csak POST kérés engedélyezett']);
    exit;
}
```

```

}

// Bekérem a JSON adatokat és ellenőrzöm a szükséges mezőket.
$input = json_decode(file_get_contents("php://input"), true);
if (
    !isset($input['foglalas_id'], $input['ertekeles']) ||           // Ha
    hiányzik a foglalás ID vagy az értékelés.
    !is_numeric($input['ertekeles']) ||                             // vagy
    nem szám az értékelés.
    $input['ertekeles'] < 1 || $input['ertekeles'] > 5             // vagy
    nem 1-5 közötti szám.
) {
    http_response_code(400);
    echo json_encode(['error' => 'Hiányzó vagy érvénytelen adatok']);
    exit;
}

// Kiszűröm és átalakítom az értékeket.
$foglalas_id = (int)$input['foglalas_id'];
$ertekeles = (int)$input['ertekeles'];
$velemeney = isset($input['velemeney']) ? trim($input['velemeney']) : null;

try {
    // Elmentem az értékelést az adatbázisba, az aktuális dátummal.
    $stmt = $pdo->prepare("
        INSERT INTO ertekelesok (foglalasok_id, ertekeles, velemeney, datum)
        VALUES (:fid, :ert, :v, NOW())
    ");
    $stmt->execute([
        ':fid' => $foglalas_id,
        ':ert' => $ertekeles,
        ':v'   => $velemeney
    ]);

    // Sikeres mentés esetén visszajelzést küldök.
    http_response_code(201);
    echo json_encode(['status' => 'Értékelés rögzítve']);
} catch (PDOException $e) {
    // Hiba esetén visszajelzek a problémáról.
    http_response_code(500);
    echo json_encode(['error' => 'Adatbázis hiba: ' . $e->getMessage()]);
}

```

Értékelés törlése

```

<?php
// Először beillesztem az adatbázis-kapcsolatot.
require_once "kapcsolat.php";

```

```
header('Content-Type: application/json');

// Csak DELETE metódust engedek ezen az API-n keresztül.
if ($_SERVER['REQUEST_METHOD'] !== 'DELETE') {
    http_response_code(405);
    echo json_encode(['error' => 'Csak DELETE metódussal érhető el ez a
végpont.']);
    exit;
}

// Ellenőrzöm, hogy van-e 'id' megadva a kérésben, és szám-e.
if (!isset($_GET['id']) || !ctype_digit($_GET['id'])) {
    http_response_code(400);
    echo json_encode(['error' => 'Hiányzó vagy érvénytelen értékelés ID']);
    exit;
}

$id = (int)$_GET['id']; // Egész számmá konvertálom.

try {
    // Elvégzem az értékelés törlését az adott azonosító alapján.
    $stmt = $pdo->prepare("DELETE FROM ertekelesek WHERE id = :id");
    $stmt->execute([':id' => $id]);

    // Sikeres törlés után visszajelzést küldök JSON-ben.
    echo json_encode(['status' => 'Értékelés törölve']);
} catch (PDOException $e) {
    // Ha bármilyen adatbázishiba történik, azt is lekezelem.
    http_response_code(500);
    echo json_encode(['error' => 'Adatbázis hiba: ' . $e->getMessage()]);
}
```

HTML forrásfájlok

Értékelés létrehozása

```
<!DOCTYPE html>
<html lang="hu">
<head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <link rel="stylesheet" href="menu.css"> <!-- Betöltöm a navigáció stílusait -->
    <title>Értékelések listája</title>
    <style>
        /* Alap oldalstílus: betűtípus, háttér */
        body {
            font-family: Arial, sans-serif;
            background-color: #f9f9f9;
            margin: 0;
```

```
padding: 0;
}
/* Táblázat és fejléc formázása */
table {
width: 90%;
margin: 30px auto;
border-collapse: collapse;
}
th, td {
padding: 12px;
border: 1px solid #ddd;
text-align: center;
}
th {
background-color: #007BFF;
color: white;
}

/* Admin gombok stílusa */
.admin-button {
background-color: #6c757d;
color: white;
padding: 6px 18px;
border: none;
border-radius: 4px;
cursor: pointer;
transition: background 0.2s;
}
.admin-button:hover {
background-color: #5a6268;
}
/* Zöld akciógomb formázása */
.zold-gomb {
background-color: #28a745;
color: white;
border: none;
border-radius: 5px;
padding: 8px 20px;
font-weight: 500;
cursor: pointer;
transition: background 0.2s;
}
.zold-gomb:hover {
background-color: #218838;
}
/* Hiba- és töltésüzenetek */
.error { color: red; text-align: center; margin: 20px 0; }
#loading { text-align: center; margin: 20px 0; }
```

```

/* Értékelés író űrlap konténer */
#ertekeles-form-container {
    max-width: 600px;
    margin: 40px auto;
    background: #fff;
    padding: 24px;
    border-radius: 10px;
    box-shadow: 0 2px 6px #0001;
    display: none; /* Csak felhasználónak látszik majd */
}
/* Reszponzív beállítások */
@media (max-width: 768px) {
    table { width: 100%; font-size: 14px; }
    .admin-button, .zold-gomb { padding: 8px 12px; font-size: 14px; }
    #ertekeles-form-container { width: 95%; padding: 20px; }
}
@media (max-width: 480px) {
    table { font-size: 13px; }
    .admin-button, .zold-gomb { font-size: 13px; padding: 6px 10px; }
    #ertekeles-form-container { padding: 16px; }
}
</style>
</head>
<body>
    <header>
        <!-- Logó és felhasználónév -->
        <a href="index.html" class="logo">Időpontfoglaló <span id="felhasznalo-
nev"></span></a>
        <nav>
            <a href="szolgaltatok.html">Szolgáltatók</a>
            <a href="idopontok.html">Időpontok</a>
            <a href="ertekelesek.html">Értékelések</a>
            <a href="bejelentkezes.html" id="nav-login"><b>Bejelentkezés</b></a>
            <a href="fiokom.html" id="nav-account" class="hidden">Fiókom</a>
            <a href="#" id="nav-logout" onclick="kijelentkezes()" class="hidden">
                <span class="color">Kijelentkezés</span>
            </a>
        </nav>
    </header>

    <!-- Töltés és hibaüzenet helye -->
    <div id="loading">Betöltés...</div>
    <div class="error" id="error" style="display:none;"></div>

    <!-- Értékelések táblázata -->
    <table id="ertekelesek-table" style="display:none;">
        <thead>
            <tr>

```



```
        <th>Felhasználó</th>
        <th>Szolgáltató</th>
        <th>Értékelés</th>
        <th>Vélemény</th>
        <th>Dátum</th>
        <th id="ertekeles-gomb-fejlec" style="display:none;">Művelet</th>
    </tr>
</thead>
<tbody id="ertekelesek-body"></tbody>
</table>

<!-- Űrlap új értékeléshez (csak felhasználónak) -->
<div id="ertekeles-form-container">
    <h3>Új értékelés írása</h3>
    <form id="ertekeles-form">
        <label>Foglalás ID: <input type="number" name="foglalas_id" required min="1"
style="width:90px;"></label>
        <label>Értékelés (1-5):
            <select name="ertekeles" required>
                <option>1</option><option>2</option><option>3</option><option>4</option>
<option>5</option>
            </select>
        </label>
        <label>Vélemény (opcionális):<textarea name="velemenyn"
rows="3"></textarea></label>
        <button type="submit" class="zold-gomb">Beküldés</button>
        <span id="ertekeles-valasz"></span>
    </form>
</div>

<script>
    // Bejelentkezett felhasználó adatok
    const felhasznalo = JSON.parse(sessionStorage.getItem('felhasznalo'));
    const szerepkor = felhasznalo?.szerepkor || null;
    const isFelhasznalo = szerepkor === 'felhasznalo';
    const isAdmin = szerepkor === 'admin';

    // Ha be vagyok jelentkezve, megjelenítem a nevem és a menüt
    if (felhasznalo?.nev) {
        document.getElementById('felhasznalo-nev').textContent = ` -
${felhasznalo.nev}`;
        document.getElementById('nav-login').classList.add('hidden');
        document.getElementById('nav-account').classList.remove('hidden');
        document.getElementById('nav-logout').classList.remove('hidden');
    }
    // Ha admin vagyok, engedélyezem a művelet oszlopot
    if (isAdmin) document.getElementById('ertekeles-gomb-fejlec').style.display =
'';
```

```
// Lekérem és megjelenítem az értékeléseket
fetch('Ertekelesek_listazasa.php')
  .then(res => res.json())
  .then(data => {
    document.getElementById('loading').style.display = 'none';
    if (data.error || !Array.isArray(data.data)) throw new Error(data.error ||
'Váratlan válasz');
    if (data.count === 0) {
      document.getElementById('error').textContent = 'Nincs még értékelés.';
      return;
    }
    const tbody = document.getElementById('ertekelesek-body');
    // Minden értékelést sorba rendelek a táblázatba
    data.data.forEach(e => {
      const tr = document.createElement('tr');
      let muvelet = '';
      if (isAdmin) muvelet = `<td><button class="admin-button"
onclick="torles(${e.id})">Törlés</button></td>`;
      tr.innerHTML = `
        <td>${e.felhasznalo_nev}</td>
        <td>${e.szolgaltato_nev}</td>
        <td>${e.ertekeles} / 5</td>
        <td>${e.velemenye} || ''</td>
        <td>${e.datum ? new Date(e.datum).toLocaleDateString('hu-HU')} :
''</td>
        <td>${muvelet}</td>`;
      tbody.appendChild(tr);
    });
    document.getElementById('ertekelesek-table').style.display = '';
  })
  .catch(err => {
    document.getElementById('loading').style.display = 'none';
    document.getElementById('error').textContent = `Hiba: ${err.message}`;
    document.getElementById('error').style.display = '';
  });

// Műveletek: törlés, kijelentkezés, új értékelés űrlap megjelenítés
function torles(id) {
  if (confirm('Biztosan törlöm?')) window.location.href =
`Ertekelesek_torlese.html?id=${id}`;
}
function kijelentkezés() {
  sessionStorage.removeItem('felhasznalo');
  window.location.href = 'index.html';
}
if (isFelhasznalo) document.getElementById('ertekeles-form-
container').style.display = 'block';
```

```
// Kezelem az új értékelés beküldését
document.getElementById('ertekeles-form')?.addEventListener('submit',
function(e) {
    e.preventDefault();
    const data = {foglalas_id: this.foglalas_id.value, ertekeles:
this.ertekeles.value, velemeney: this.velemeney.value};
    const valaszElem = document.getElementById('ertekeles-valasz');
    valaszElem.textContent = 'Küldés...';
    fetch('Ertekelesek_letrehozasa.php', {method: 'POST', headers: {'Content-
Type': 'application/json'}, body: JSON.stringify(data)})
        .then(res => res.json())
        .then(res => {
            valaszElem.style.color = res.status ? 'green' : 'red';
            valaszElem.textContent = res.status ? 'Siker!' : res.error || 'Hiba!';
            if (res.status) this.reset();
        })
        .catch(() => valaszElem.textContent = 'Hiba!');
    });
</script>
</body>
</html>
```

Értékelés törlése

```
<!DOCTYPE html>
<html lang="hu">
<head>
    <!-- Dokumentum típusa és nyelv beállítása -->
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <!-- Külső stíluslap betöltése -->
    <link rel="stylesheet" href="menu.css" />
    <!-- Oldal címe -->
    <title>Értékelés törlése</title>
    <style>
        /* Oldal alapstílusai */
        body {
            font-family: Arial, sans-serif;
            background-color: #f5f5f5;
            margin: 0;
            padding: 0;
        }
        /* Tartalom tároló középre igazítása */
        .container {
            max-width: 400px;
            margin: 30px auto;
            background-color: white;
```

```
padding: 30px;
border-radius: 10px;
box-shadow: 0 2px 8px rgba(0,0,0,0.1);
text-align: center;
}
/* Beviteli mező stílusa */
.container input {
width: 100%;
padding: 12px;
margin-bottom: 20px;
font-size: 16px;
border: 1px solid #ccc;
border-radius: 5px;
}
/* Gomb stílusa */
.container button {
width: 100%;
padding: 12px;
font-size: 16px;
color: white;
background-color: blue;
border: none;
border-radius: 5px;
cursor: pointer;
margin-bottom: 10px;
}
.container button:hover {
background-color: blue;
}
/* Címsor stílusa */
h2 {
text-align: center;
color: #007BFF;
}
/* Eredmény üzenet doboz */
.result {
margin-top: 18px;
font-size: 1.06em;
padding: 13px;
border-radius: 5px;
min-height: 26px;
text-align: center;
font-weight: bold;
}
/* Siker üzenet stílusa */
.success {
color: #18823a;
background: #e9faed;
```

```
        border: 1px solid #28a745;
    }
    /* Hiba üzenet stílusa */
    .error {
        color: #b1001c;
        background: #fff0f3;
        border: 1px solid #e30035;
    }
    /* Figyelmeztető szöveg */
    .color {
        color: red;
    }
    /* Rejtett elem */
    .hidden {
        display: none !important;
    }
    /* Mobil nézet 768px alatt */
    @media (max-width: 768px) {
        .container {
            width: 90%;
            padding: 20px;
        }
        .container input,
        .container button {
            font-size: 15px;
        }
        h2 {
            font-size: 24px;
            margin-top: 20px;
        }
    }
    /* Mobil nézet 480px alatt */
    @media (max-width: 480px) {
        .container {
            padding: 16px;
        }
        .container input,
        .container button {
            font-size: 14px;
            padding: 10px;
        }
        h2 {
            font-size: 20px;
        }
    }
</style>
</head>
<body>
```

```
<!-- Fejléc és navigáció -->
<header>
  <a href="index.html" class="logo">Időpontfoglaló <span id="felhasznalo-
nev"></span></a>
  <nav>
    <a href="szolgaltatok.html">Szolgáltatók</a>
    <a href="idopontok.html">Időpontok</a>
    <a href="ertekelesek.html">Értékelések</a>
    <a href="bejelentkezes.html" id="nav-login"><b>Bejelentkezés</b></a>
    <a href="fiokom.html" id="nav-account" class="hidden">Fiókom</a>
    <a href="#" id="nav-logout" onclick="kijelentkezes()" class="hidden"><span
class="color">Kijelentkezés</span></a>
  </nav>
</header>

<!-- Oldalcím -->
<h2>Értékelés törlése</h2>

<!-- Törlő űrlap és eredmény megjelenítő -->
<div class="container">
  <input type="number" id="torles-id" placeholder="Értékelés ID" required />
  <button id="btn-delete">Törlés</button>
  <div class="result" id="result"></div>
</div>

<script>
  // URL paraméterből ID betöltése, ha elérhető
  const urlParams = new URLSearchParams(window.location.search);
  const idFromUrl = urlParams.get('id');
  if (idFromUrl) {
    document.getElementById('torles-id').value = idFromUrl;
  }

  // Törlés gomb eseménykezelője
  document.getElementById('btn-delete').addEventListener('click', () => {
    const id = document.getElementById('torles-id').value.trim();
    const resultDiv = document.getElementById('result');
    resultDiv.textContent = '';
    resultDiv.className = 'result';
    if (!id) {
      resultDiv.textContent = 'Kérem, adja meg a törlendő értékelés ID-t.';
      resultDiv.classList.add('error');
      return;
    }
    // Megerősítés
    if (!confirm(`Biztosan törölni szeretné a(z) ${id} ID-jú értékelést?`))
return;
```

```

// POST kérés törléshez
fetch(`Ertekelesek_torlese.php?id=${id}`, {
  method: 'POST'
})
.then(res => res.json())
.then(data => {
  if (data.status === 'siker' || data.success) {
    resultDiv.textContent = 'Törlés sikeres.';
    resultDiv.classList.add('success');
    setTimeout(() => { window.location.href = 'ertekelesek.html'; }, 1000);
  } else {
    resultDiv.textContent = data.error || 'Ismeretlen hiba történt.';
    resultDiv.classList.add('error');
  }
})
.catch(err => {
  console.error(err);
  resultDiv.textContent = 'Hálózati vagy szerver hiba!';
  resultDiv.classList.add('error');
});
});

// Kijelentkezés funkció
function kijelentkezés() {
  sessionStorage.removeItem('felhasznalo');
  window.location.href = 'index.html';
}

// Navigáció frissítése bejelentkezési állapot szerint
const user = JSON.parse(sessionStorage.getItem('felhasznalo'));
if (user && user.nev) {
  document.getElementById('felhasznalo-nev').textContent = ` - ${user.nev}`;
  document.getElementById('nav-login').classList.add('hidden');
  document.getElementById('nav-account').classList.remove('hidden');
  document.getElementById('nav-logout').classList.remove('hidden');
}
</script>
</body>
</html>

```

HTML kimenetek

The first screenshot shows the 'Új értékelés írása' (Write new review) form. It includes a 'Foglalás azonosítója' (Booking ID) input field, a rating dropdown set to '1', and a 'Vélemény (opcionális)' (Optional comment) text area. A green 'Értékelés beküldése' (Submit review) button is at the bottom.

The second screenshot shows the 'Értékelések' (Reviews) table for 'Kovacs Bela'. The table has columns: Felhasználó, Szolgáltató, Értékelés, Vélemény, Dátum, and Művelet. The first row shows a rating of '1 / 5' and a date of '2025. 06. 23.'. The 'Művelet' column contains a 'Értékelés törlése' (Delete review) button.

Felhasználó	Szolgáltató	Értékelés	Vélemény	Dátum
Kovacs Bela	Elegance Spa	1 / 5		2025. 06. 23.

Felhasználó	Szolgáltató	Értékelés	Vélemény	Dátum	Művelet
Kovacs Bela	Elegance Spa	1 / 5		2025. 06. 23.	Értékelés törlése

1.6 Fiókom funkció működése

Feladatrészben betöltöm a kapcsolat.php-t, JSON-fejléccet állítok be, és csak POST kéréseket fogadok el. A beérkező JSON-bemenetet dekódolom, ellenőrzöm a felhasználó_id és időpont_id mezők meglétét, hiány esetén 400-as hibát adok. Tranzakciót indítok, beszúrom a foglalás rekordot, frissítem az időpont foglalhatóságát, majd commit-olom, és JSON-ben visszaküldöm a sikeres státuszt. Módosításkor GET paraméterként kérem az id-t, a JSON-bemenet alapján dinamikusan összeállítom az UPDATE utasítást, végrehajtom, és visszaigazolom a frissítést. Törléskor GET id-vel fogadom a kérést, végrehajtom a DELETE utasítást, és JSON-ben visszaadom a törlés eredményét. A frontend három HTML-oldalt kínál: létrehozáshoz űrlapot a dátum, idő és megjegyzés beviteléhez; módosításhoz szerkeszthető állapot- és megjegyzésmezőket; törléshez megerősítő képernyőt. A JavaScript eseménykezelők AJAX hívásokkal kommunikálnak a PHP végpontokkal, ellenőrzöm a felhasználói jogosultságot, és a válasz alapján értesítem vagy átirányítom a felhasználót.

PHP forrásfájlok

Foglalások létrehozása

```
<?php
// Betöltöm az adatbázis-kapcsolatot biztosító fájlt.
require_once "kapcsolat.php";

// Beállítom, hogy JSON formátumban küldöm vissza a választ.
header('Content-Type: application/json');

// Beolvasom a JSON bemenetet, amit a kliens küldött.
$input = json_decode(file_get_contents("php://input"), true);

// Ellenőrzöm, hogy megkaptam-e a szükséges mezőket.
if (!isset($input['felhasznalo_id'], $input['idopont_id'])) {
    http_response_code(400);
    echo json_encode(['error' => 'Hiányzó mezők']);
    exit;
}

try {
    // Elindítom az adatbázis tranzakciót, hogy egyszerre tudjak több műveletet.
    biztonságosan végrehajtani
    $pdo->beginTransaction();

    // Létrehozok egy új foglalást az adatbázisban.
    $stmt = $pdo->prepare("INSERT INTO foglalasok (felhasznalok_id, idopontok_id,
allapot, megjegyzes)
                                VALUES (:fid, :iid, 'lefoglalt', :msg)");
    $stmt->execute([
        ':fid' => $input['felhasznalo_id'],          // A bejelentkezett felhasználó
ID-je
        ':iid' => $input['idopont_id'],              // A kiválasztott időpont ID-je
        ':msg' => $input['megjegyzes'] ?? null      // Opcionális megjegyzés
    ]);

    // A kiválasztott időpontot már nem lehet újra lefoglalni, ezért frissítem az
állapotát.
    $pdo->prepare("UPDATE idopontok SET foglalhato = 0 WHERE id = :iid")
        ->execute([':iid' => $input['idopont_id']]);

    // Ha minden sikerült, elkötelezem a tranzakciót
    $pdo->commit();

    // Sikeres foglalás visszajelzése JSON-ben.
    echo json_encode(['status' => 'Sikeres foglalás']);
}
```

```
} catch (PDOException $e) {  
    // Ha bármilyen hiba történik, visszavonom a tranzakciót  
    $pdo->rollBack();  
    http_response_code(500);  
    echo json_encode(['error' => $e->getMessage()]);  
}
```

Foglalások módosítása

```
<?php  
// Betöltöm az adatbázis kapcsolatot.  
require_once "kapcsolat.php";  
  
// Beállítom, hogy JSON formátumban válaszoljon a szerver.  
header('Content-Type: application/json');  
  
// Ellenőrzöm, hogy érkezett-e érvényes ID a lekérdezéshez.  
if (!isset($_GET['id']) || !ctype_digit($_GET['id'])) {  
    http_response_code(400);  
    echo json_encode(['error' => 'Hiányzó vagy hibás foglalás ID']);  
    exit;  
}  
  
// Beolvasom a JSON-bemenetet és tömbbé alakítom.  
$input = json_decode(file_get_contents('php://input'), true);  
  
// Előkészítem a módosítandó mezőket és paramétereket.  
$fields = [];  
$params = [':id' => $_GET['id']];  
  
// Ha az "allapot" mező szerepel a kérésben, hozzáadom a frissítéshez.  
if (isset($input['allapot'])) {  
    $fields[] = 'allapot = :a';  
    $params[':a'] = $input['allapot'];  
}  
  
// Ha a "megjegyzes" mező is szerepel, azt is hozzáadom.  
if (isset($input['megjegyzes'])) {  
    $fields[] = 'megjegyzes = :m';  
    $params[':m'] = $input['megjegyzes'];  
}  
  
// Ha egyik módosítható mező sincs megadva, hibaüzenetet küldök.  
if (empty($fields)) {  
    http_response_code(400);  
    echo json_encode(['error' => 'Nincs módosítandó mező']);  
    exit;  
}
```

```
// Összeállítom a dinamikus SQL lekérdezést.
$sql = "UPDATE foglalasok SET " . implode(' ', $fields) . " WHERE id = :id";

try {
    // Elküldöm az SQL frissítést az adatbázisnak.
    $stmt = $pdo->prepare($sql);
    $stmt->execute($params);

    // Sikeres válasz JSON-ben.
    echo json_encode(['status' => 'Foglalás módosítva']);
} catch (PDOException $e) {
    // Ha valami hiba történik, visszajelzek róla
    http_response_code(500);
    echo json_encode(['error' => $e->getMessage()]);
}
```

Foglalások törlése

```
<?php
// Először betöltöm az adatbáziskapcsolatot.
require_once "kapcsolat.php";

// Beállítom, hogy JSON válaszokat küldjek vissza.
header('Content-Type: application/json');

// Ellenőrzöm, hogy kaptam-e érvényes foglalás ID-t a GET paraméterek között.
if (!isset($_GET['id']) || !ctype_digit($_GET['id'])) {
    http_response_code(400); // Hibás kérés
    echo json_encode(['error' => 'Érvénytelen foglalás ID']);
    exit;
}

// Megpróbálom törölni a megadott ID-jú foglalást az adatbázisból.
try {
    $stmt = $pdo->prepare("DELETE FROM foglalasok WHERE id = :id");
    $stmt->execute([':id' => $_GET['id']]);

    // Ha sikerült, visszajelzem JSON-ben.
    echo json_encode(['status' => 'Foglalás törölve']);
} catch (PDOException $e) {
    // Ha hiba történik az adatbázisművelet közben, hibakódot küldök vissza.
    http_response_code(500);
    echo json_encode(['error' => $e->getMessage()]);
}
```

HTML forrásfájlok

Foglalások létrehozása

```
<!DOCTYPE html>
<html lang="hu">
<head>
  <!-- **Dokumentum típusa és karakterkészlet beállítása** -->
  <meta charset="UTF-8" />
  <!-- **Reszponzív nézet beállítása** -->
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <!-- **Navigációs menü stíluslap betöltése** -->
  <link rel="stylesheet" href="menu.css" />
  <!-- **Oldal címe** -->
  <title>Foglalás létrehozása</title>
  <style>
    /* **Oldal alapstílusai** */
    body {
      font-family: Arial, sans-serif;
      background: #f5f5f5;
      margin: 0;
      padding: 0;
    }
    /* **Űrlap konténer középre igazítása és stílus** */
    .container {
      max-width: 500px;
      margin: 40px auto;
      background: white;
      padding: 30px;
      border-radius: 8px;
      box-shadow: 0 2px 8px rgba(0,0,0,0.1);
    }
    /* **Címsor stílusa** */
    h2 {
      text-align: center;
      color: black;
    }
    /* **Űrlapmezők címkéinek stílusa** */
    label {
      display: block;
      margin-top: 15px;
      font-weight: bold;
    }
    /* **Beviteli mezők és szövegterület** */
    input,
    textarea {
      width: 100%;
      padding: 10px;
      margin-top: 5px;
    }
  </style>
</head>
<body>
  <div class="container">
    <h2>Foglalás létrehozása</h2>
    <div>
      <label>Név</label>
      <input type="text">
    </div>
    <div>
      <label>Telefon</label>
      <input type="text">
    </div>
    <div>
      <label>Email</label>
      <input type="text">
    </div>
    <div>
      <label>Szöveg</label>
      <textarea>
    </div>
    <input type="button" value="Foglalás" />
  </div>
</body>
</html>
```

```
border: 1px solid #ccc;
border-radius: 4px;
}
/* **Szövegterület méretezése** */
textarea {
  resize: vertical;
  min-height: 80px;
}
/* **Küldőgomb stílusa** */
button {
  margin-top: 20px;
  width: 100%;
  padding: 12px;
  background-color: #28a745;
  color: white;
  border: none;
  border-radius: 5px;
  cursor: pointer;
}
/* **Gomb hover állapota** */
button:hover {
  background-color: #218838;
}
/* **Figyelmeztető szöveg stílusa** */
.color {
  color: red;
}
/* **Rejtett elem osztály** */
.hidden {
  display: none !important;
}
/* **Reszponzív stílusok (768px alatt)** */
@media (max-width: 768px) {
  .container {
    width: 90%;
    padding: 20px;
    margin: 30px auto;
  }
  h2 {
    font-size: 24px;
  }
  label,
  input,
  textarea,
  button {
    font-size: 15px;
  }
  button {
```

```
        padding: 10px;
    }
}
/* **Mobil nézet (480px alatt)** */
@media (max-width: 480px) {
    h2 {
        font-size: 20px;
    }
    label,
    input,
    textarea,
    button {
        font-size: 14px;
    }
    button {
        padding: 10px;
    }
}
</style>
</head>
<body>
    <!-- **Fejléc és navigáció** -->
    <header>
        <a href="index.html" class="logo">Időpontfoglaló <span id="felhasznalo-
nev"></span></a>
        <nav>
            <a href="szolgaltatok.html">Szolgáltatók</a>
            <a href="idopontok.html">Időpontok</a>
            <a href="ertekelesek.html">Értékelések</a>
            <a href="bejelentkezes.html" id="nav-login"><b>Bejelentkezés</b></a>
            <a href="fiokom.html" id="nav-account" class="hidden">Fiókom</a>
            <a href="#" id="nav-logout" onclick="kijelentkezes()" class="hidden"><span
class="color">Kijelentkezés</span></a>
        </nav>
    </header>

    <!-- **Űrlap előkészítő szkript: felhasználó ellenőrzése és navigáció** -->
    <script>
        const user = JSON.parse(sessionStorage.getItem('felhasznalo'));
        if (!user || !user.id) {
            alert("Csak bejelentkezett felhasználó tud foglalni.");
            window.location.href = "bejelentkezes.html";
        }
        if (user.nev) {
            document.getElementById('felhasznalo-nev').textContent = ` - ${user.nev}`;
        }
        document.getElementById('nav-login').classList.add('hidden');
        document.getElementById('nav-account').classList.remove('hidden');
```

```
document.getElementById('nav-logout').classList.remove('hidden');
</script>

<!-- **Foglalás megerősítése űrlap** -->
<h2>Foglalás megerősítése</h2>
<div class="container">
  <form id="foglalas-form">
    <!-- **Dátum mező (csak olvasható)** -->
    <label>Dátum:</label>
    <input type="text" id="datum" readonly />

    <!-- **Idő mező (csak olvasható)** -->
    <label>Idő:</label>
    <input type="text" id="ido" readonly />

    <!-- **Megjegyzés mező (opcionális)** -->
    <label for="megjegyzes">Megjegyzés (opcionális):</label>
    <textarea id="megjegyzes" placeholder="Pl. Kérem, érkezzen pontosan..."></textarea>

    <!-- **Beküldés gomb** -->
    <button type="submit">Foglalás megerősítése</button>
  </form>
</div>

<script>
  // **URL paraméterek olvasása**
  const params = new URLSearchParams(window.location.search);
  const idopontId = params.get('id');
  if (!idopontId) {
    alert("Hiányzó időpont azonosító.");
    window.location.href = "idopontok.html";
  }

  // **Időpont adatok lekérése és beállítása**
  fetch("idopontok_listazasa.php")
    .then(res => res.json())
    .then(data => {
      if (data.status !== 'siker') throw new Error("Időpontok betöltése sikertelen.");
      const rec = data.data.find(r => String(r.id) === idopontId);
      if (!rec) {
        alert("Nincs ilyen időpont.");
        window.location.href = "idopontok.html";
      }
      document.getElementById('datum').value = rec.datum;
      document.getElementById('ido').value = rec.ido;
    })
  </script>
```

```

        .catch(err => {
            console.error(err);
            alert(err.message);
            window.location.href = "idopontok.html";
        });

// **Űrlap beküldése AJAX-szal**
document.getElementById('foglalas-form').addEventListener('submit',
function(e) {
    e.preventDefault();
    const megj = document.getElementById('megjegyzes').value.trim();

    fetch("Foglalasok_letrehozasa.php", {
        method: "POST",
        headers: { "Content-Type": "application/json" },
        body: JSON.stringify({
            felhasznalo_id: user.id,
            idopont_id: parseInt(idopontId),
            megjegyzes: megj || null
        })
    })
    .then(res => res.json())
    .then(data => {
        if (data.status) {
            alert("Foglalás sikeresen létrehozva.");
            window.location.href = "idopontok.html";
        } else {
            alert("Hiba: " + (data.error || "Ismeretlen hiba"));
        }
    })
    .catch(err => {
        console.error(err);
        alert("Hiba a foglalás során.");
    });
});

// **Kijelentkezés funkció**
function kijelentkezés() {
    sessionStorage.removeItem('felhasznalo');
    window.location.href = "index.html";
}
</script>
</body>
</html>

```


Foglalások módosítása

```
<!DOCTYPE html>
<html lang="hu">

<head>
  <!-- Dokumentum típusa és karakterkészlet beállítása -->
  <meta charset="UTF-8" />
  <!-- Reszponzív nézet beállítása -->
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <!-- Navigációs menü stíluslap betöltése -->
  <link rel="stylesheet" href="menu.css" />
  <!-- Oldal címe -->
  <title>Foglalás módosítása</title>
  <style>
    /* Alapoldal stílusok */
    body {
      font-family: Arial, sans-serif;
      background-color: #f9f9f9;
      margin: 0;
      padding: 0;
    }
    /* Láthatatlan osztály */
    .hidden {
      display: none !important;
    }
    /* Űrlap tároló stílusai */
    .container {
      max-width: 500px;
      margin: 40px auto;
      background: white;
      padding: 30px;
      border-radius: 8px;
      box-shadow: 0 2px 8px rgba(0,0,0,0.1);
    }
    /* Címsor stílusa */
    h2 {
      text-align: center;
      color: black;
      margin-bottom: 20px;
    }
    /* Űrlapmező címkék */
    label {
      display: block;
      margin-top: 15px;
```

```
        font-weight: bold;
    }
    /* Beviteli mezők és szövegterület stílusai */
    input,
    textarea {
        width: 100%;
        padding: 10px;
        margin-top: 5px;
        border: 1px solid #ccc;
        border-radius: 4px;
        box-sizing: border-box;
    }
    /* Gomb stílusok */
    button {
        margin-top: 20px;
        width: 100%;
        padding: 12px;
        background-color: #007BFF;
        color: white;
        border: none;
        border-radius: 5px;
        cursor: pointer;
    }
    /* Gomb hover állapota */
    button:hover {
        background-color: #0056b3;
    }
    /* Reszponzív (768px alatt) */
    @media (max-width: 768px) {
        .container {
            width: 90%;
            padding: 20px;
            margin: 30px auto;
        }
        h2 {
            font-size: 24px;
        }
        label,
        input,
        textarea,
        button {
            font-size: 15px;
        }
        button {
            padding: 10px;
        }
    }
    /* Mobil nézet (480px alatt) */
```

```
@media (max-width: 480px) {
  h2 {
    font-size: 20px;
  }
  label,
  input,
  textarea,
  button {
    font-size: 14px;
  }
  button {
    padding: 10px;
  }
}
</style>
</head>

<body>
  <!-- Fejléc és navigáció -->
  <header>
    <a href="index.html" class="logo">
      Időpontfoglaló <span id="felhasznalo-nev"></span>
    </a>
    <nav>
      <a href="szolgaltatok.html">Szolgáltatók</a>
      <a href="idopontok.html">Időpontok</a>
      <a href="ertekelesek.html">Értékelések</a>
      <a href="bejelentkezes.html" id="nav-login"><b>Bejelentkezés</b></a>
      <a href="fiokom.html" id="nav-account" class="hidden">Fiókom</a>
      <a href="#" id="nav-logout" class="hidden" onclick="kijelentkezes()">
        <span style="color:red">Kijelentkezés</span>
      </a>
    </nav>
  </header>

  <!-- Oldalcím -->
  <h2>Foglalás módosítása</h2>

  <!-- Űrlap konténer -->
  <div class="container">
    <form id="modositas-form">
      <!-- Állapot mező -->
      <label for="allapot">Állapot:</label>
      <input type="text" id="allapot" name="allapot" required />

      <!-- Megjegyzés mező -->
      <label for="megjegyzes">Megjegyzés:</label>
      <textarea id="megjegyzes" name="megjegyzes" rows="4"></textarea>
    </form>
  </div>
</body>
</html>
```

```
<!-- Mentés gomb -->
<button type="submit">Mentés</button>
</form>
</div>

<script>
  // Felhasználó ellenőrzése
  const user = JSON.parse(sessionStorage.getItem('felhasznalo'));
  if (!user || user.szerekor !== 'felhasznalo') {
    alert("Csak bejelentkezett felhasználó módosíthat foglalást.");
    window.location.href = "bejelentkezes.html";
  }
  // Felhasználónév beállítása
  document.getElementById('felhasznalo-nev').textContent = ` - ${user.nev}`;
  // Navigáció állapotok beállítása
  document.getElementById('nav-login').classList.add('hidden');
  document.getElementById('nav-account').classList.remove('hidden');
  document.getElementById('nav-logout').classList.remove('hidden');

  // URL paraméterből ID kinyerése
  const params = new URLSearchParams(window.location.search);
  const id = params.get('id');
  if (!id) {
    alert("Hiányzó foglalás ID.");
    window.location.href = "fiokom.html";
  }

  // Foglalás adatainak betöltése
  fetch(`Foglalasaim_listazasa.php?felhasznalo_id=${user.id}`)
    .then(res => res.json())
    .then(json => {
      if (json.error) throw new Error(json.error);
      const fog = json.data.find(f => f.id == id);
      if (!fog) {
        alert("Nincs ilyen foglalás.");
        window.location.href = "fiokom.html";
      }
      document.getElementById('allapot').value = fog.allapot;
      document.getElementById('megjegyzes').value = fog.megjegyzes || '';
    })
    .catch(err => {
      console.error(err);
      alert("Hiba a foglalás betöltésekor: " + err.message);
      window.location.href = "fiokom.html";
    });

  // Form beküldése PUT módszer szimulációval
```

```
document.getElementById('modositas-form').addEventListener('submit', e => {
  e.preventDefault();
  const payload = {
    _method: 'PUT',
    allapot: document.getElementById('allapot').value,
    megjegyzes: document.getElementById('megjegyzes').value
  };
  // Továbbítás POST-tal
  fetch(`Foglalasok_modositasa.php?id=${id}`, {
    method: 'POST',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify(payload)
  })
  .then(res => res.json())
  .then(res => {
    if (res.status) {
      alert("Foglalás sikeresen módosítva.");
      window.location.href = "fiokom.html";
    } else {
      alert("Hiba: " + (res.error||'Ismeretlen hiba'));
    }
  })
  .catch(err => {
    console.error(err);
    alert("Hiba a módosítás során.");
  });
});

// Kijelentkezés függvény
function kijelentkezés() {
  sessionStorage.removeItem('felhasznalo');
  window.location.href = "index.html";
}
</script>
</body>
</html>
```

Foglalások törlése

```
<!DOCTYPE html>
<html lang="hu">
<head>
  <!-- Dokumentum típusa és karakterkészlet beállítása -->
  <meta charset="UTF-8"/>
  <!-- Reszponzív nézet beállítása -->
  <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
  <!-- Navigációs menü stíluslap betöltése -->
```

```
<link rel="stylesheet" href="menu.css">
<!-- Oldal címe -->
<title>Foglalás törlése</title>
<style>
  /* Alapoldal stílusok */
  body {
    font-family: Arial, sans-serif;
    background-color: #f9f9f9;
    margin: 0; padding: 0;
  }
  /* Űrlap tároló középre igazítása */
  .container {
    max-width: 500px;
    margin: 60px auto;
    background: white;
    padding: 30px;
    border-radius: 8px;
    box-shadow: 0 2px 8px rgba(0,0,0,0.1);
    text-align: center;
  }
  /* Címsor stílusa */
  h2 {
    text-align: center;
    color: #dc3545;
    margin-bottom: 20px;
  }
  /* Információs szöveg */
  p {
    margin-bottom: 30px;
    font-size: 16px;
  }
  /* Gombok alapstílusai */
  button {
    padding: 10px 20px;
    font-size: 16px;
    border: none;
    border-radius: 5px;
    cursor: pointer;
  }
  /* Törlés gomb */
  .delete-btn {
    background-color: blue;
    color: white;
    margin-right: 10px;
  }
  .delete-btn:hover {
    background-color: blue;
  }
}
```

```
/* Mégse gomb */
.cancel-btn {
  background-color: #6c757d;
  color: white;
}
.cancel-btn:hover {
  background-color: #5a6268;
}
/* Mobil nézet (768px alatt) */
@media (max-width: 768px) {
  .container {
    width: 90%;
    padding: 20px;
    margin: 40px auto;
  }
  h2 { font-size: 24px; }
  p { font-size: 15px; }
  button {
    width: 100%;
    margin: 10px 0;
    font-size: 15px;
  }
}
/* Mobil nézet (480px alatt) */
@media (max-width: 480px) {
  h2 { font-size: 20px; }
  p { font-size: 14px; }
  button {
    font-size: 14px;
    padding: 10px;
  }
}
</style>
</head>
<body>
  <!-- Fejléc és navigáció -->
  <header>
    <a href="index.html" class="logo">
      Időpontfoglaló <span id="felhasznalo-nev"></span>
    </a>
    <nav>
      <a href="szolgaltatok.html">Szolgáltatók</a>
      <a href="idopontok.html">Időpontok</a>
      <a href="ertekelesek.html">Értékelések</a>
      <a href="fiokom.html" id="nav-account">Fiókom</a>
      <a href="#" onclick="kijelentkezés()">
        <span style="color:red">Kijelentkezés</span>
      </a>
    </nav>
  </header>
</body>
```

```
</nav>
</header>

<!-- Oldalcím -->
<h2>Foglalás törlése</h2>

<!-- Törlési megerősítő konténer -->
<div class="container">
  <!-- Információ az aktuális foglalásról -->
  <p id="info">Betöltés...</p>
  <!-- Véglegesítés gomb -->
  <button class="delete-btn" id="confirm-delete">Törlés véglegesítése</button>
  <!-- Mégse gomb -->
  <button class="cancel-btn" id="cancel">Mégse</button>
</div>

<script>
  // Jogosultság ellenőrzése: csak felhasználó
  const user = JSON.parse(sessionStorage.getItem('felhasznalo'));
  if (!user || user.szerepkor !== 'felhasznalo') {
    alert("Csak bejelentkezett felhasználó törölhet foglalást.");
    location.href = "bejelentkezes.html";
  }
  // Felhasználónév megjelenítése a fejlécben
  document.getElementById('felhasznalo-nev').textContent = ` - ${user.nev}`;

  // URL paraméterként érkező foglalás ID
  const params = new URLSearchParams(window.location.search);
  const id = params.get('id');
  if (!id) {
    alert("Hiányzó foglalás ID.");
    location.href = "fiokom.html";
  }

  // Foglalás adatainak lekérése és info mező frissítése
  fetch(`Foglalasaim_listazasa.php?felhasznalo_id=${user.id}`)
    .then(response => response.json())
    .then(data => {
      const fog = data.data.find(f => f.id == id);
      if (!fog) throw new Error("Érvénytelen foglalás ID.");
      document.getElementById('info').textContent =
        `Biztosan törölni szeretné a következő foglalást? Dátum: ${fog.datum},
        Idő: ${fog.ido}`;
    })
    .catch(error => {
      alert(error.message);
      location.href = "fiokom.html";
    });
});
```



```
// Törlés gomb eseménykezelő
document.getElementById('confirm-delete').addEventListener('click', () => {
    fetch(`Foglalasok_torlese.php?id=${id}`)
        .then(res => res.json())
        .then(result => {
            if (result.status) {
                alert("Foglalás törölve.");
                location.href = "fiokom.html";
            } else {
                alert("Hiba: " + (result.error || "Ismeretlen hiba"));
            }
        })
        .catch(() => {
            alert("Hiba a törlés során.");
        });
});

// Mégse gomb visszairányítás
document.getElementById('cancel').addEventListener('click', () => {
    location.href = "fiokom.html";
});

// Kijelentkezés funkció
function kijelentkezés() {
    sessionStorage.removeItem('felhasznalo');
    location.href = "index.html";
}
</script>
</body>
</html>
```

Szakmai Gyakorlat I. – Frontend – Backend integrációs szakasz

Orosz Kristóf – EYZWG9

← → ↺ ⓘ localhost/ido pontfoglalo/Foglasok_letrehozasa.html?id=133 ☆ 📄 ⬇️ 🌐 ⋮

Időpontfoglaló - Orosz Kristóf

Szolgáltatók Időpontok Értékelések Fiókomban Kijelentkezés

Foglalás megerősítése

Dátum:

2025-06-19

Idő:

20:14:00

Megjegyzés (opcionális):

P1. Kérem, érkezzen pontosan...

Foglalás megerősítése

← → ↺ ⓘ localhost/ido pontfoglalo/Foglasok_modositasa.html?id=33 ☆ 📄 ⬇️ 🌐 ⋮

Időpontfoglaló - Orosz Kristóf

Szolgáltatók Időpontok Értékelések Fiókomban Kijelentkezés

Foglalás módosítása

Állapot:

le foglalt

Megjegyzés:

Mentés

← → ↺ ⓘ localhost/ido pontfoglalo/Foglasok_torlese.html?id=33 ☆ 📄 ⬇️ 🌐 ⋮

Időpontfoglaló - Orosz Kristóf

Szolgáltatók Időpontok Értékelések Fiókomban Kijelentkezés

Foglalás törlése

Biztos törölni szeretné a következő foglalást? Dátum: 2025-06-10, Idő: 13:00:00

Törles véglegesítése

Mégse

82