**SZABADKAI MŰSZAKI SZAKFŐISKOLA**

**PROJEKTUM**

**Integrált web rendszerek és haladó webprogramozás tantárgyakból**

**Elektronikus nyilvántartás házi kedvencekről**

**HALLGATÓ(K) MENTOR**

**Tumbász Barbara, Fodor Dávid dr Zlatko Čović**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**SZABADKA, 2025.**

## TARTALOM

Catalog

[TARTALOM 2](#_Toc23068)

[A FELADATKÖR MEGHATÁROZÁSA 5](#_Toc22433)

[PIACKUTATÁS 6](#_Toc15816)

[A budapesti belvárosi állatorvosi rendelő weboldala 6](#_Toc31586)

[https://www.cityvet.hu/ 6](#_Toc3538)

[Kisállat regisztrációs platform 6](#_Toc25204)

[https://animal-id.net/ 6](#_Toc14253)

[Homevet Állatorvosi Rendelő weboldala 7](#_Toc6617)

[https://homevetrendelo.hu/ 7](#_Toc31802)

[3 további funkció: 7](#_Toc28440)

[KIDOLGOZÁS 8](#_Toc27247)

[Pet.php 8](#_Toc22215)

[Appointments.php 10](#_Toc10263)

[Vets.php 15](#_Toc7375)

[Admin\_owners.php 16](#_Toc16792)

[AZ ADATBÁZIS STRUKTURÁJA 18](#_Toc21405)

[A PROJEKT MŰKÖDÉSÉNEK LEÍRÁSA 19](#_Toc16611)

[1. Kép: Bejelentkezés oldal 19](#_Toc25027)

[2.Kép: Regisztráció oldal 19](#_Toc3326)

[3.Kép: Profil módosítás oldal 20](#_Toc18919)

[4.Kép: Foglalásaim oldal 20](#_Toc24267)

[5.Kép: Kisállataim oldal 21](#_Toc26357)

[6.Kép: Időpont foglalás oldal 21](#_Toc15595)

[7.Kép: Orvos időpontjai oldal 22](#_Toc26333)

[8.Kép: Időpont szerkesztés oldal 22](#_Toc16378)

[9.Kép: Orvos munkaidő módosítás oldal 23](#_Toc24694)

[10.Kép: Admin orvos kezelő oldal 23](#_Toc10618)

[11.Kép: Admin felhasználó kezelő oldal 24](#_Toc1333)

[12.Kép: Mobil app foglalás 24](#_Toc27418)

[13.Kép: Mobil app időpontjaim 25](#_Toc13785)

[14.Kép: Mobil app profil módosítás 26](#_Toc26948)

[15.Kép: Mobil app házi állataim 27](#_Toc14652)

[FELHASZNÁLT IRODALOM 28](#_Toc23320)

## A FELADATKÖR MEGHATÁROZÁSA

**<https://la.stud.vts.su.ac.rs/elektronikusNyilvantartasHaziKedvencekrol/public/index.php>**A projekt célja egy **állatorvosi elektronikus nyilvántartó rendszer** megvalósítása, amely két fő részből áll:

**Webes felület**: az állatorvosok és adminisztrátorok számára, ahol a páciensek (házi kedvencek) adatai, kórlapok, oltások, időpontfoglalások és egyéb adminisztráció kezelhető.

**Mobilalkalmazás**: a gazdik számára, ahol megtekinthetik kedvenceik adatait, oltásait, foglalhatnak időpontot és értesítéseket kaphatnak a közelgő eseményekről.

A rendszer célja az átlátható, gyors és egyszerű adminisztráció, valamint a gazdik kényelmes és naprakész tájékoztatása.

## PIACKUTATÁS

#### A budapesti belvárosi állatorvosi rendelő weboldala

#### <https://www.cityvet.hu/>

Az időpont foglaláshoz első lépésben az orvost kell kiválasztanunk, majd az oltás vagy az általános vizsgálat közül a szolgáltatást. Az oltásra előrelátott idő 15 perc, a vizsgálatra 30 perc. A szolgáltatás kiválasztása után lehetőségünk nyílik az időpont pontos kiválasztására a szabad időpontok közül. Az időpontok 15 perces intervallumokra vannak osztva. A foglaláshoz Személyes adatainkat (név, email, telefonszám) kell megadnunk, vagy felhasználói fiókkal foglalhatunk.

Előnyök: Az időpontok és a szolgáltatások hossza jó összhangban vannak, így egyszerű a foglalások kezelése.

Hátrányok: Felhasználói fiók nélkül is lehet foglalni.

#### Kisállat regisztrációs platform

**<https://animal-id.net/>**

Az oldal betekintést ad, hogy milyen funkciókkal rendelkezik mobil alkalmazásuk (Pet Care App by Animal ID). Részletes információt tudunk tárolni kedvencünkről. Miután regisztrált a felhasználó, lehetőséget kap, hogy kedvencét is beregisztrálja. Regisztráció nélkül vásárolni tudunk nyakörvet, kiegészítőket, de mindaddig még nincs adat, az állatról, nem képes generálni QR kódot. Nemzetközi adatbázisokat használnak, hogy nagyobb körben tudjanak keresést folytatni.

Előny: QR kód nyakörvet tudunk vásárolni kedvencünknek, amelyen megtalálható a róla nyilvántartott összes adat.

Hátrány: Desktop nézetben is látható a telefonos alkalmazásokhoz a letöltő gombok.

#### Homevet Állatorvosi Rendelő weboldala

**<https://homevetrendelo.hu/>**

Az időpontfoglalásnál, még mielőtt megkezdenénk megnézhetjük az ott dolgozó orvosokat. Kiválasztjuk az állatot, a szolgáltatás, ahol az előrelátott idő is fel van tüntetve, az időpontok a legkorábbitól választhatók. 5 perces időkorlát van beállítva az adminisztrációs adatok kitöltéséhez.

Előnyök: Könnyen kezelhető felület az információk frissítésére.Rugalmas kapcsolatfelvételi lehetőségek.

Hátrány: Információk hiányossága az ott dolgozóknál.

3 további funkció:

Mobil applikációs értesítések

Hűségprogram

Terápia követése, értesítés a jövendő vizsgálatról

## KIDOLGOZÁS

Pet.php  **<?php  
require\_once *\_\_DIR\_\_* . '/../core/init.php';  
require\_once *\_\_DIR\_\_* . '/../includes/auth.php';  
require\_once *\_\_DIR\_\_* . '/../classes/Pet.php';  
require\_once *\_\_DIR\_\_* . '/../classes/PetRepository.php';  
  
  
switch ($method) {  
 case 'GET':  
 $id = $\_GET['id'] ?? null;  
 if ($id) {  
 $pet = $repo->find((int)$id);  
 if (!$pet || !canAccess($user, $pet)) {  
 sendJSON(['message' => 'A keresett házikedvenc nem található vagy nincs jogosultság.'], 404);  
 }  
 sendJSON($pet->toArray());  
 } else {  
 $pets = match ($user['role']) {  
 'admin' => $repo->findAll(),  
 'owner' => $repo->findByOwner($user['sub']),  
 'vet' => $repo->findByVet($user['sub']),  
 };  
 sendJSON(array\_map(fn($p) => $p->toArray(), $pets));  
 }  
 break;  
  
 case 'POST':  
 if ($user['role'] !== 'owner' && $user['role'] !== 'admin') {  
 sendJSON(['message' => 'Nincs jogosultság új házikedvenc létrehozásához.'], 403);  
 }  
  
 $data = json\_decode(file\_get\_contents('php://input'), true);  
 if (!is\_array($data)) {  
 sendJSON(['message' => 'Érvénytelen JSON adatok.'], 400);  
 }  
  
 $required = ['name', 'gender', 'birth\_date', 'species', 'breed', 'vet\_id'];  
 foreach ($required as $field) {  
 if (empty($data[$field])) {  
 sendJSON(['message' => "Hiányzó adat"], 400);  
 }  
 }  
  
 $pet = new Pet($data);  
 $newId = $repo->create($pet, $user['sub']);  
  
 if (!is\_int($newId)) {  
 sendJSON(['message' => $newId], 409);  
 }  
  
 sendJSON(['message' => 'Házikedvenc sikeresen hozzáadva.', 'pet\_id' => $newId], 201);  
 break;  
  
 case 'PUT':  
 $put = json\_decode(file\_get\_contents('php://input'), true);  
 if (!is\_array($put)) {  
 sendJSON(['message' => 'Érvénytelen JSON adatok.'], 400);  
 }  
  
 if (empty($put['pet\_id'])) {  
 sendJSON(['message' => 'Hiányzik az azonosító.'], 400);  
 }  
  
 $existing = $repo->find((int)$put['pet\_id']);  
 if (!$existing || !canAccess($user, $existing)) {  
 sendJSON(['message' => 'Nincs jogosultság a szerkesztéshez.'], 403);  
 }  
  
 $required = ['name', 'gender', 'birth\_date', 'species', 'breed', 'vet\_id'];  
 foreach ($required as $field) {  
 if (empty($put[$field])) {  
 sendJSON(['message' => "Hiányzó adat"], 400);  
 }  
 }  
  
 $updated = new Pet(array\_merge($existing->toArray(), $put, ['pet\_id' => $existing->id]));  
 $repo->update($updated);  
  
 sendJSON(['message' => 'Házikedvenc sikeresen frissítve.']);  
 break;  
  
 case 'DELETE':  
 $petId = $\_GET['pet\_id'] ?? null;  
  
 if (!$petId) {  
 sendJSON(['message' => 'Hiányzik az azonosító.'], 400);  
 }  
  
 $existing = $repo->find((int)$petId);  
 if (!$existing || !canAccess($user, $existing)) {  
 sendJSON(['message' => 'Nincs jogosultság a törléshez.'], 403);  
 }  
  
 $repo->delete($existing->id);  
 sendJSON(['message' => 'Házikedvenc sikeresen törölve.']);  
 break;  
  
 default:  
 sendJSON(['message' => 'A HTTP metódus nem támogatott.'], 405);  
}  
  
/\* Jogosultságellenőrzés \*/  
function canAccess(array $user, Pet $pet): bool  
{  
 return match ($user['role']) {  
 'admin' => true,  
 'owner' => $user['sub'] === $pet->ownerId,  
 'vet' => $user['sub'] === $pet->vetId,  
 default => false  
 };  
}**

Appointments.php **<?php  
require\_once *\_\_DIR\_\_* . '/../core/init.php';  
require\_once *\_\_DIR\_\_* . '/../includes/auth.php';  
  
$user = authMiddleware();  
if (!$user) {  
 sendJSON(['message' => 'Nincs jogosultság'], 401);  
}  
  
$method = $\_SERVER['REQUEST\_METHOD'];  
  
switch ($method) {  
  
 /\* -------------------------------  
 GET: fetch appointments or slots  
 -------------------------------- \*/  
 case 'GET':  
 // If user asks for available slots  
 if ($\_GET['action'] ?? '' === 'slots') {  
 $vetId = (int)($\_GET['vet\_id'] ?? 0);  
 $date = $\_GET['date'] ?? '';  
 $treatmentId = (int)($\_GET['treatment\_id'] ?? 0);  
  
 if (!$vetId || !$date || !$treatmentId) {  
 sendJSON(['message' => 'Hiányzó paraméterek'], 400);  
 }  
  
 // Load treatment duration  
 $stmt = $pdo->prepare("SELECT duration\_min FROM treatment WHERE treatment\_id = ?");  
 $stmt->execute([$treatmentId]);  
 $treatment = $stmt->fetch();  
 if (!$treatment) {  
 sendJSON(['message' => 'Érvénytelen kezelés'], 404);  
 }  
 $duration = (int)$treatment['duration\_min'];  
  
 // Weekday (0 = Sunday, 1 = Monday...)  
 $weekday = date('w', strtotime($date));  
  
 // Load vet schedule  
 $stmt = $pdo->prepare("SELECT \* FROM vet\_schedule WHERE vet\_id = ? AND weekday = ?");  
 $stmt->execute([$vetId, $weekday]);  
 $schedule = $stmt->fetch();  
 if (!$schedule) {  
 sendJSON(['message' => 'Ezen a napon nincs rendelés'], 404);  
 }  
  
 $slotMinutes = (int)$schedule['slot\_minutes'];  
 $startTime = new DateTime($date . ' ' . $schedule['start\_time']);  
 $endTime = new DateTime($date . ' ' . $schedule['end\_time']);  
  
 // Load already booked appointments  
 $stmt = $pdo->prepare("SELECT starts\_at, ends\_at FROM appointment   
 WHERE vet\_id = ? AND DATE(starts\_at) = ? AND status = 'booked'");  
 $stmt->execute([$vetId, $date]);  
 $booked = $stmt->fetchAll();  
  
 $bookedIntervals = array\_map(fn($row) => [  
 new DateTime($row['starts\_at']),  
 new DateTime($row['ends\_at'])  
 ], $booked);  
  
 // Generate available slots  
 $available = [];  
 $current = clone $startTime;  
 while ($current <= $endTime) {  
 $slotEnd = (clone $current)->modify("+{$duration} minutes");  
  
 if ($slotEnd > $endTime) break;  
  
 // Check overlap with booked intervals  
 $conflict = false;  
 foreach ($bookedIntervals as [$bStart, $bEnd]) {  
 if ($current < $bEnd && $slotEnd > $bStart) {  
 $conflict = true;  
 break;  
 }  
 }  
  
 if (!$conflict) {  
 $available[] = $current->format('H:i');  
 }  
  
 $current->modify("+{$slotMinutes} minutes");  
 }  
  
 sendJSON(['available\_slots' => $available]);  
 }  
  
 // Otherwise: list appointments  
 if ($user['role'] === 'admin') {  
 $stmt = $pdo->query("SELECT \* FROM appointment");  
 $appointments = $stmt->fetchAll();  
 } elseif ($user['role'] === 'vet') {  
 $stmt = $pdo->prepare("SELECT a.\*, p.name AS pet\_name, CONCAT(v.first\_name, ' ', v.last\_name) AS vet\_name, t.name AS treatment\_name  
 FROM appointment a  
 JOIN pet p ON a.pet\_id = p.pet\_id  
 JOIN vet v ON a.vet\_id = v.vet\_id  
 JOIN treatment t ON a.treatment\_id = t.treatment\_id  
 WHERE a.vet\_id = ?");  
 $stmt->execute([$user['sub']]);  
 $appointments = $stmt->fetchAll();  
 } elseif ($user['role'] === 'owner') { // owner  
 $stmt = $pdo->prepare("SELECT a.appointment\_id, a.pet\_id, a.vet\_id, a.description, a.starts\_at, a.ends\_at, a.treatment\_id, a.status,  
 p.name AS pet\_name,  
 CONCAT(v.first\_name, ' ', v.last\_name) AS vet\_name,  
 t.name AS treatment\_name  
FROM appointment a  
JOIN pet p ON a.pet\_id = p.pet\_id  
JOIN vet v ON a.vet\_id = v.vet\_id  
JOIN treatment t ON a.treatment\_id = t.treatment\_id  
JOIN owner\_pet op ON p.pet\_id = op.pet\_id  
WHERE op.owner\_id = ? ORDER BY a.starts\_at DESC  
");  
 $stmt->execute([$user['sub']]);  
 $appointments = $stmt->fetchAll();  
 }  
  
 sendJSON($appointments);  
 break;  
  
 /\* -------------------------------  
 POST: create new appointment  
 -------------------------------- \*/  
 case 'POST':  
 $data = json\_decode(file\_get\_contents('php://input'), true);  
  
 $petId = (int)($data['pet\_id'] ?? 0);  
 $vetId = (int)($data['vet\_id'] ?? 0);  
 $treatmentId = (int)($data['treatment\_id'] ?? 0);  
 $startsAt = $data['starts\_at'] ?? '';  
 $description = $data['description'] ?? '';  
  
 if (!$petId || !$vetId || !$treatmentId || !$startsAt) {  
 sendJSON(['message' => 'Hiányzó mezők'], 400);  
 }  
  
 // Owner can only create appointments for his own pets  
 if ($user['role'] === 'owner') {  
 $stmt = $pdo->prepare("SELECT 1 FROM owner\_pet WHERE owner\_id = ? AND pet\_id = ?");  
 $stmt->execute([$user['sub'], $petId]);  
 if (!$stmt->fetch()) {  
 sendJSON(['message' => 'Ez a kisállat nem a tiéd'], 403);  
 }  
 }  
  
 // Load treatment duration  
 $stmt = $pdo->prepare("SELECT duration\_min FROM treatment WHERE treatment\_id = ?");  
 $stmt->execute([$treatmentId]);  
 $treatment = $stmt->fetch();  
 if (!$treatment) {  
 sendJSON(['message' => 'Érvénytelen kezelés'], 404);  
 }  
 $duration = (int)$treatment['duration\_min'];  
  
 $startDt = new DateTime($startsAt);  
 $endDt = (clone $startDt)->modify("+{$duration} minutes");  
  
 // Check conflicts  
 $stmt = $pdo->prepare("SELECT 1 FROM appointment WHERE vet\_id = ? AND status='booked'  
 AND ((starts\_at < ? AND ends\_at > ?) OR (starts\_at < ? AND ends\_at > ?))");  
 $stmt->execute([$vetId, $endDt->format('Y-m-d H:i:s'), $startDt->format('Y-m-d H:i:s'),  
 $endDt->format('Y-m-d H:i:s'), $startDt->format('Y-m-d H:i:s')]);  
 if ($stmt->fetch()) {  
 sendJSON(['message' => 'Időpont ütközés'], 409);  
 }  
  
 // Insert  
 $stmt = $pdo->prepare("INSERT INTO appointment (pet\_id, vet\_id, description, starts\_at, ends\_at, treatment\_id, status)  
 VALUES (?, ?, ?, ?, ?, ?, 'booked')");  
 $stmt->execute([$petId, $vetId, $description, $startDt->format('Y-m-d H:i:s'), $endDt->format('Y-m-d H:i:s'), $treatmentId]);  
  
 sendJSON(['message' => 'Időpont lefoglalva', 'appointment\_id' => $pdo->lastInsertId()], 201);  
 break;  
  
 /\* -------------------------------  
 PATCH: update appointment  
 -------------------------------- \*/  
 case 'PATCH':  
 parse\_str(file\_get\_contents("php://input"), $data);  
  
 $appointmentId = (int)($data['appointment\_id'] ?? 0);  
 $status = $data['status'] ?? '';  
 $description = $data['description'] ?? '';  
  
 if (!$appointmentId) {  
 sendJSON(['message' => 'Hiányzó appointment\_id'], 400);  
 }  
  
 // Load appointment  
 $stmt = $pdo->prepare("SELECT \* FROM appointment WHERE appointment\_id = ?");  
 $stmt->execute([$appointmentId]);  
 $appointment = $stmt->fetch();  
 if (!$appointment) {  
 sendJSON(['message' => 'Időpont nem található'], 404);  
 }  
  
 // Permission checks  
 if ($user['role'] === 'owner') {  
 $stmt = $pdo->prepare("SELECT 1 FROM owner\_pet WHERE owner\_id = ? AND pet\_id = ?");  
 $stmt->execute([$user['sub'], $appointment['pet\_id']]);  
 if (!$stmt->fetch()) {  
 sendJSON(['message' => 'Nincs jogosultság ehhez az időponthoz'], 403);  
 }  
 } elseif ($user['role'] === 'vet' && $appointment['vet\_id'] != $user['sub']) {  
 sendJSON(['message' => 'Ez az időpont nem hozzád tartozik'], 403);  
 }  
  
 // Update allowed fields  
 if ($status) {  
 $stmt = $pdo->prepare("UPDATE appointment SET status = ? WHERE appointment\_id = ?");  
 $stmt->execute([$status, $appointmentId]);  
 }  
 if ($description) {  
 $stmt = $pdo->prepare("UPDATE appointment SET description = ? WHERE appointment\_id = ?");  
 $stmt->execute([$description, $appointmentId]);  
 }  
  
 sendJSON(['message' => 'Időpont frissítve']);  
 break;  
  
 /\* -------------------------------  
 DELETE: delete appointment  
 -------------------------------- \*/  
 case 'DELETE':  
 $data = json\_decode(file\_get\_contents("php://input"), true);  
 $appointmentId = (int)($data['appointment\_id'] ?? 0);  
  
 if (!$appointmentId) {  
 sendJSON(['message' => 'Hiányzó foglalás'], 400);  
 }  
  
 $stmt = $pdo->prepare("SELECT \* FROM appointment WHERE appointment\_id = ?");  
 $stmt->execute([$appointmentId]);  
 $appointment = $stmt->fetch();  
 if (!$appointment) {  
 sendJSON(['message' => 'Időpont nem található'], 404);  
 }  
  
 // Permission checks  
 if ($user['role'] === 'owner') {  
 $stmt = $pdo->prepare("SELECT 1 FROM owner\_pet WHERE owner\_id = ? AND pet\_id = ?");  
 $stmt->execute([$user['sub'], $appointment['pet\_id']]);  
 if (!$stmt->fetch()) {  
 sendJSON(['message' => 'Nincs jogosultság ehhez az időponthoz'], 403);  
 }  
 } elseif ($user['role'] === 'vet' && $appointment['vet\_id'] != $user['sub']) {  
 sendJSON(['message' => 'Ez az időpont nem hozzád tartozik'], 403);  
 }  
  
 $stmt = $pdo->prepare("DELETE FROM appointment WHERE appointment\_id = ?");  
 $stmt->execute([$appointmentId]);  
  
 sendJSON(['message' => 'Időpont törölve']);  
 break;  
  
 default:  
 sendJSON(['message' => 'Metódus nem engedélyezett'], 405);  
}**

#### Vets.php

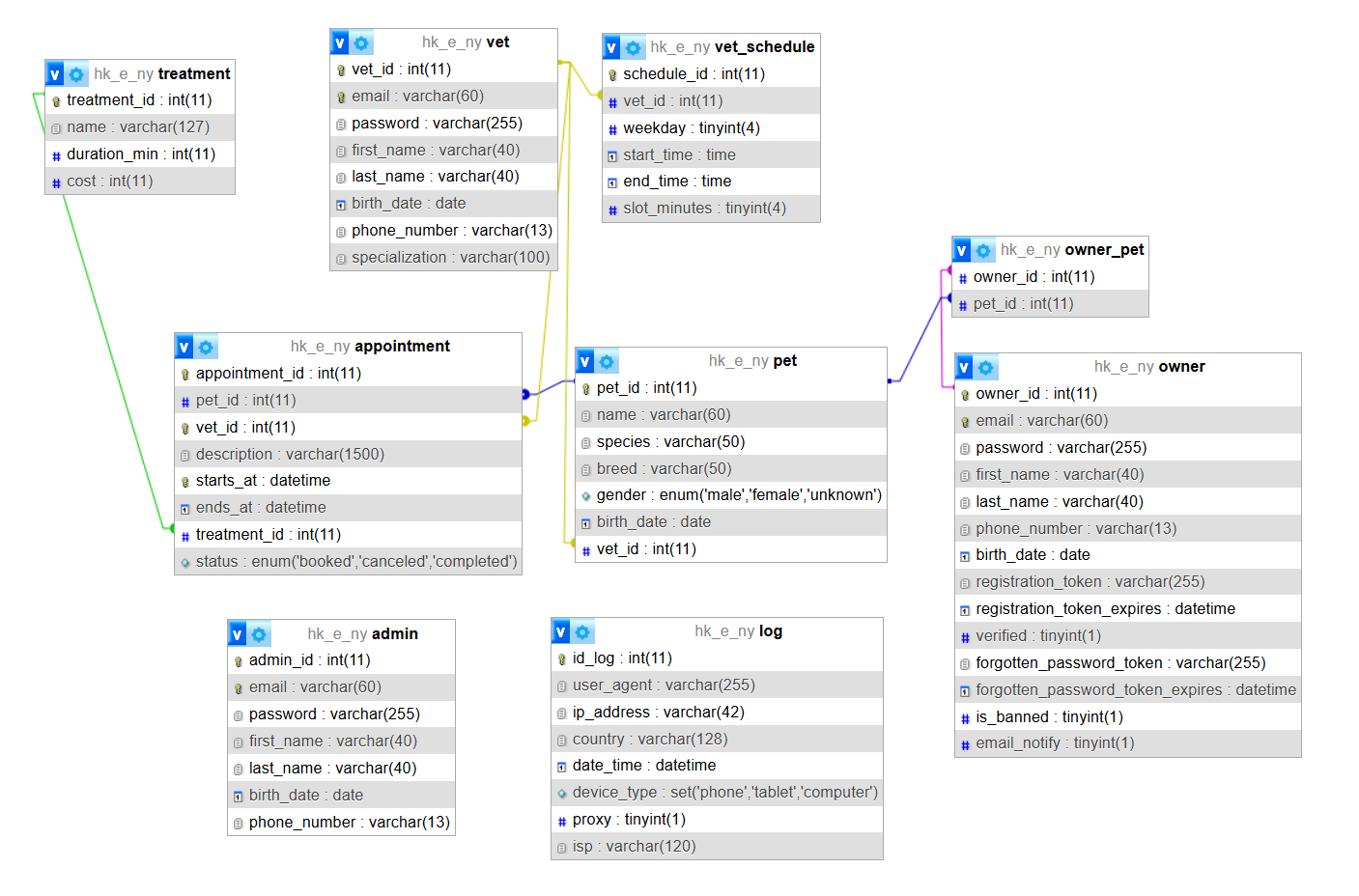
**<?php  
require\_once *\_\_DIR\_\_* . '/../vendor/autoload.php';  
require\_once *\_\_DIR\_\_* . '/../classes/Vet.php';  
require\_once *\_\_DIR\_\_* . '/../classes/VetRepository.php';  
require\_once *\_\_DIR\_\_* . '/authorize\_admin.php';  
require\_once *\_\_DIR\_\_* . '/../core/init.php';  
  
use App\Vet;  
use App\VetRepository;  
  
$vetRepo = new VetRepository($pdo);  
  
$method = $\_SERVER['REQUEST\_METHOD'];  
$id = $\_GET['id'] ?? null;  
  
header('Content-Type: application/json');  
  
switch ($method) {  
  
 case 'GET':  
 if ($id) {  
 $vet = $vetRepo->find((int)$id);  
 echo json\_encode($vet ? $vet->toArray(false) : ['message' => 'Orvos nem található']);  
 // false: ne adjuk vissza a jelszót  
 } else {  
 $vets = array\_map(fn($v) => $v->toArray(false), $vetRepo->findAll());  
 echo json\_encode($vets);  
 }  
 break;  
  
 case 'POST':  
 case 'PUT':  
 case 'DELETE':  
 // Ezekhez kell admin  
 $admin = authorizeAdmin();  
  
 if ($method === 'POST') {  
 $data = json\_decode(file\_get\_contents("php://input"), true);  
 $vet = new Vet($data);  
 if (!$vet->password) {  
 http\_response\_code(400);  
 echo json\_encode(['message' => 'A jelszó kötelező az új orvoshoz']);  
 exit;  
 }  
 $id = $vetRepo->create($vet);  
 echo json\_encode(['message' => 'Orvos létrehozva', 'id' => $id]);  
 }  
  
 if ($method === 'PUT') {  
 if (!$id) {  
 http\_response\_code(400);  
 echo json\_encode(['message' => 'Hiányzik az orvos azonosítója']);  
 exit;  
 }  
 $data = json\_decode(file\_get\_contents("php://input"), true);  
 $vet = new Vet(array\_merge($data, ['vet\_id' => $id]));  
 $vetRepo->update($vet);  
 echo json\_encode(['message' => 'Orvos frissítve']);  
 }  
  
 if ($method === 'DELETE') {  
 if (!$id) {  
 http\_response\_code(400);  
 echo json\_encode(['message' => 'Hiányzik az orvos azonosítója']);  
 exit;  
 }  
 $vetRepo->delete((int)$id);  
 echo json\_encode(['message' => 'Orvos törölve']);  
 }  
 break;  
  
 default:  
 http\_response\_code(405);  
 echo json\_encode(['message' => 'Nem támogatott HTTP metódus']);  
}**

#### Admin\_owners.php

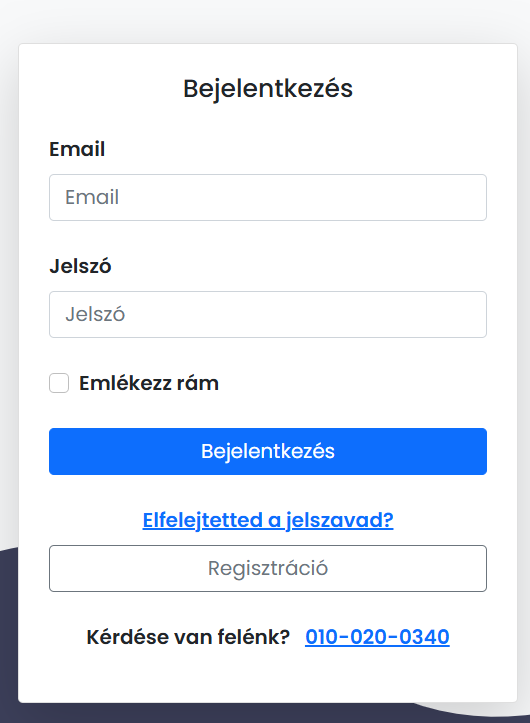
**<?php  
header("Content-Type: application/json");  
  
require\_once *\_\_DIR\_\_* . "/../includes/db.php"; // your PDO connection  
  
$method = $\_SERVER['REQUEST\_METHOD'];  
$action = $\_GET['action'] ?? '';  
  
try {  
 // 🔹 List all owners  
 if ($method === 'GET' && $action === 'list') {  
 $stmt = $pdo->query("SELECT owner\_id, last\_name, first\_name, email, phone\_number, birth\_date, is\_banned FROM owner");  
 echo json\_encode($stmt->fetchAll(PDO::*FETCH\_ASSOC*));  
 exit;  
 }  
  
 // 🔹 Update owner  
 if ($method === 'PUT' && $action === 'update') {  
 $data = json\_decode(file\_get\_contents("php://input"), true);  
  
 if (!$data || !$data['id']) {  
 http\_response\_code(400);  
 echo json\_encode(["success" => false, "message" => "Missing ID"]);  
 exit;  
 }  
  
 $banned = !empty($data['banned']) ? 1 : 0;  
  
 $stmt = $pdo->prepare("  
 UPDATE owner  
 SET last\_name=:last\_name, first\_name=:first\_name, email=:email,   
 phone\_number=:phone, birth\_date=:birth\_date, is\_banned=:banned   
 WHERE owner\_id=:id  
 ");  
 $stmt->execute([  
 ":id" => $data['id'],  
 ":last\_name" => $data['last\_name'],  
 ":first\_name" => $data['first\_name'],  
 ":email" => $data['email'],  
 ":phone" => $data['phone'],  
 ":birth\_date" => $data['birth\_date'],  
 ":banned" => $banned  
 ]);  
  
 echo json\_encode(["success" => true]);  
 exit;  
 }  
  
 // 🔹 Delete owner  
 if ($method === 'DELETE' && $action === 'delete') {  
 $id = $\_GET['id'] ?? null;  
 if (!$id) {  
 http\_response\_code(400);  
 echo json\_encode(["success" => false, "message" => "Missing ID"]);  
 exit;  
 }  
  
 $stmt = $pdo->prepare("DELETE FROM owner WHERE owner\_id=:id");  
 $stmt->execute([":id" => $id]);  
  
 echo json\_encode(["success" => true]);  
 exit;  
 }  
  
 http\_response\_code(400);  
 echo json\_encode(["success" => false, "message" => "Invalid request"]);  
} catch (Exception $e) {  
 http\_response\_code(500);  
 echo json\_encode(["success" => false, "message" => $e->getMessage()]);  
}**

## AZ ADATBÁZIS STRUKTURÁJA

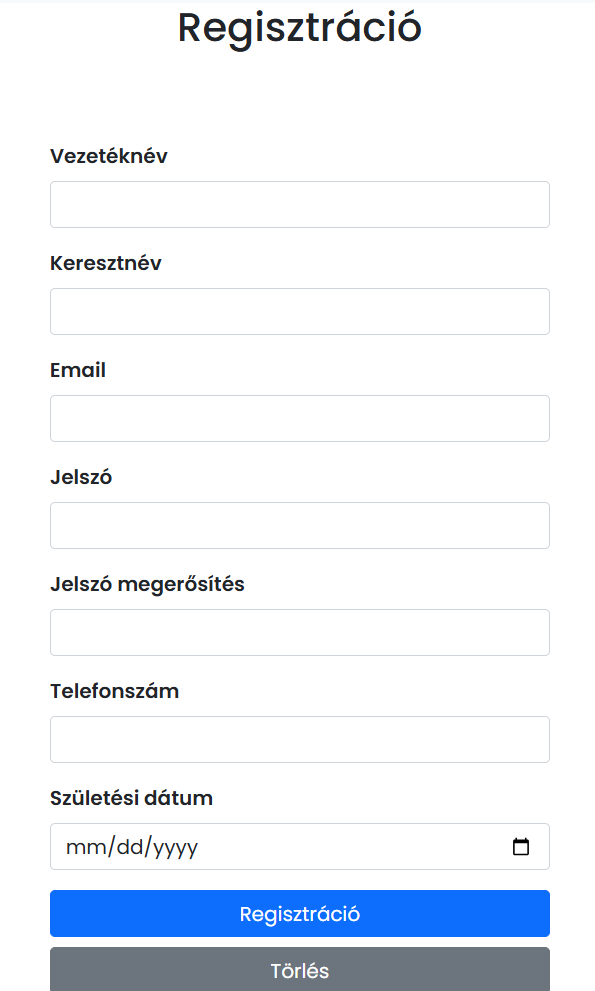
Használja a MySQL workbench programot <https://www.mysql.com/products/workbench>



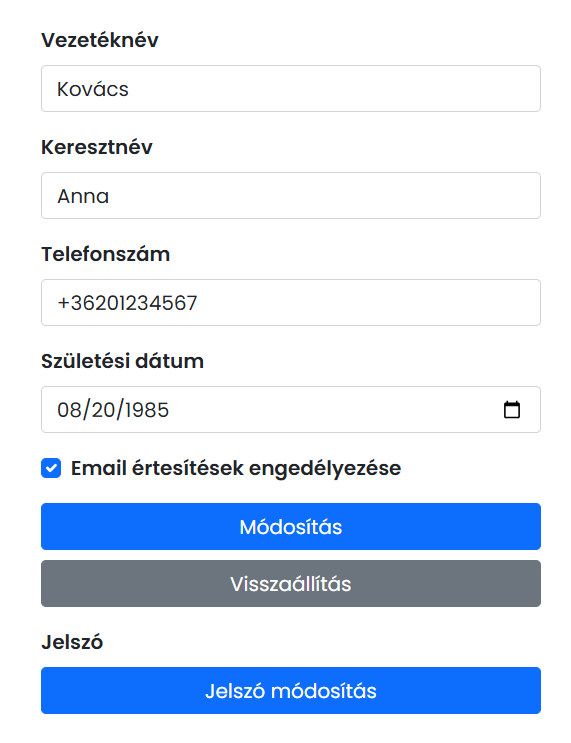
## A PROJEKT MŰKÖDÉSÉNEK LEÍRÁSA



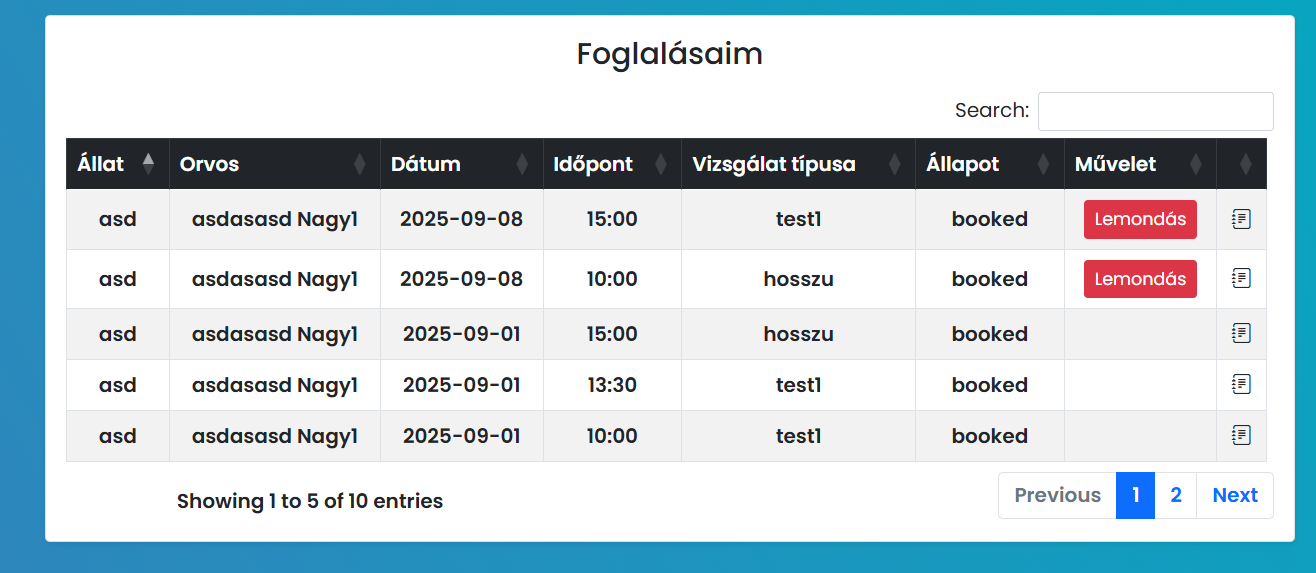
1. Kép: Bejelentkezés oldal



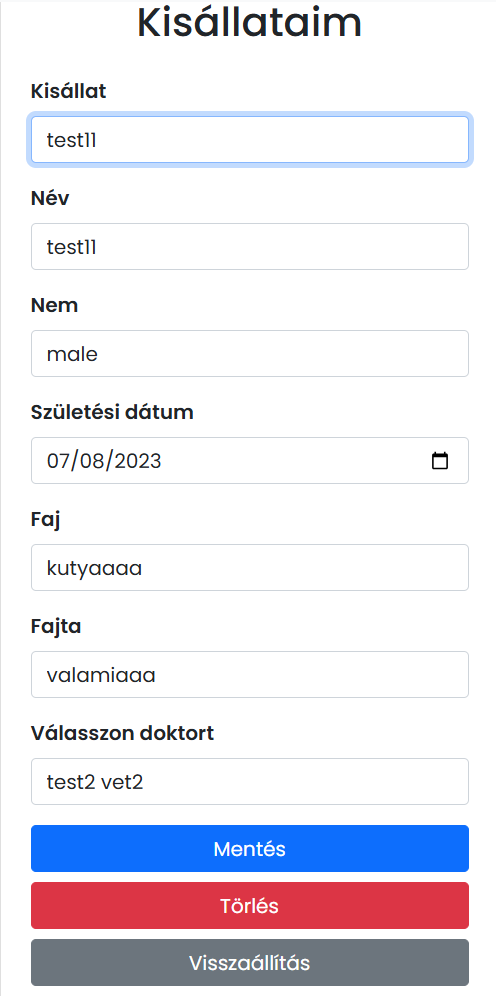
2.Kép: Regisztráció oldal



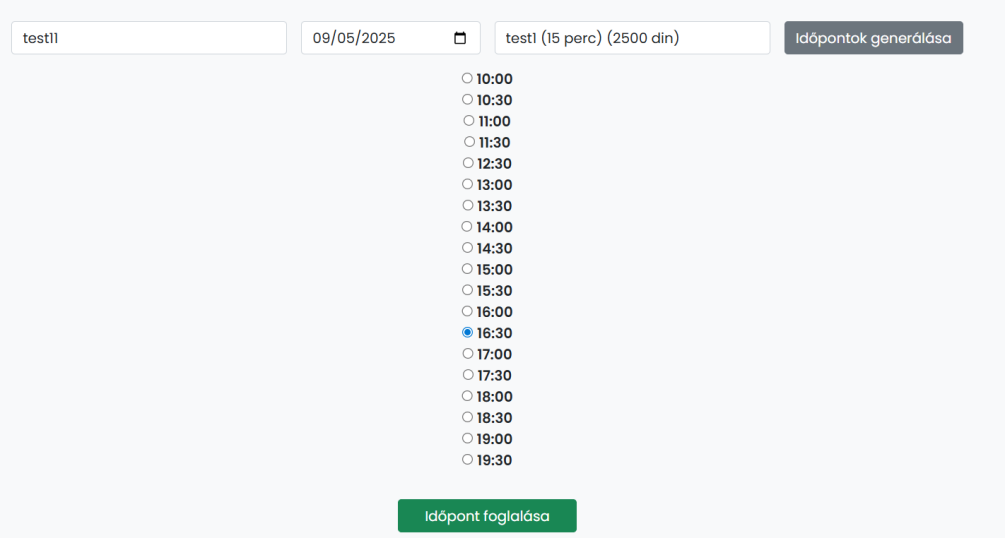
3.Kép: Profil módosítás oldal



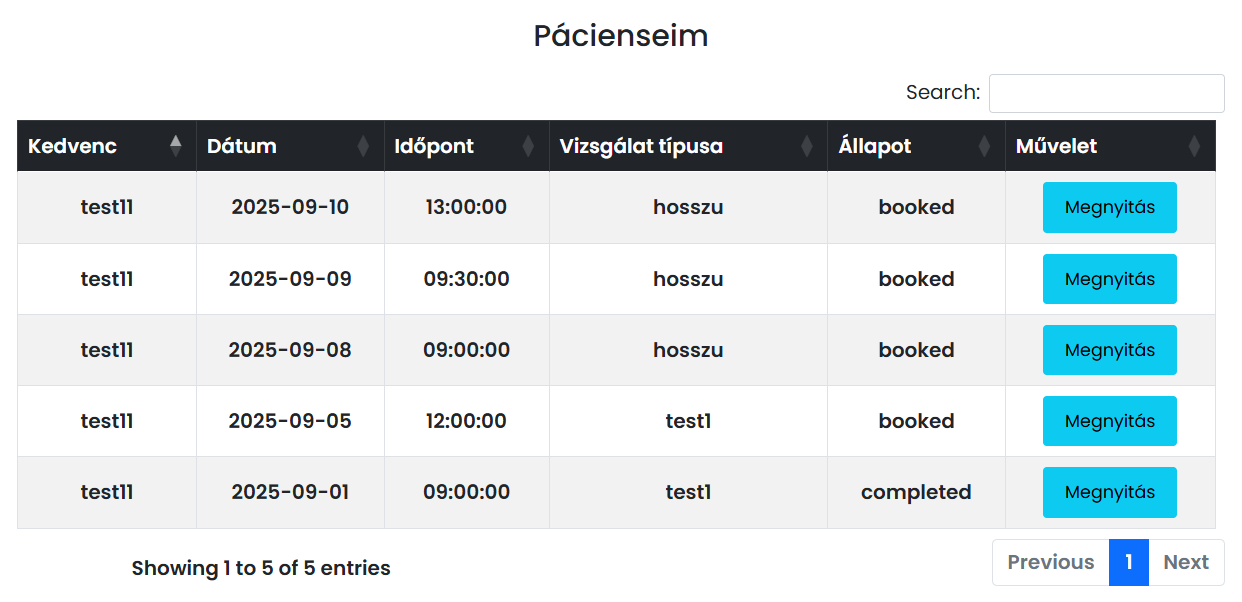
4.Kép: Foglalásaim oldal



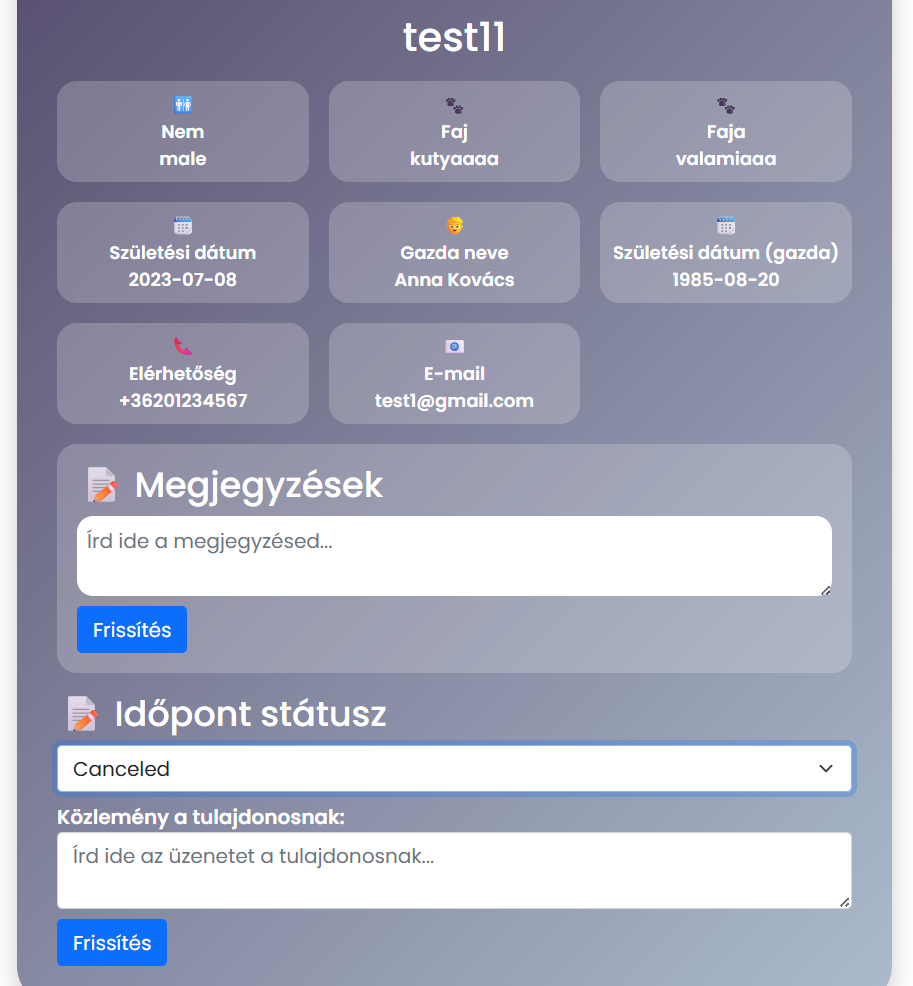
5.Kép: Kisállataim oldal



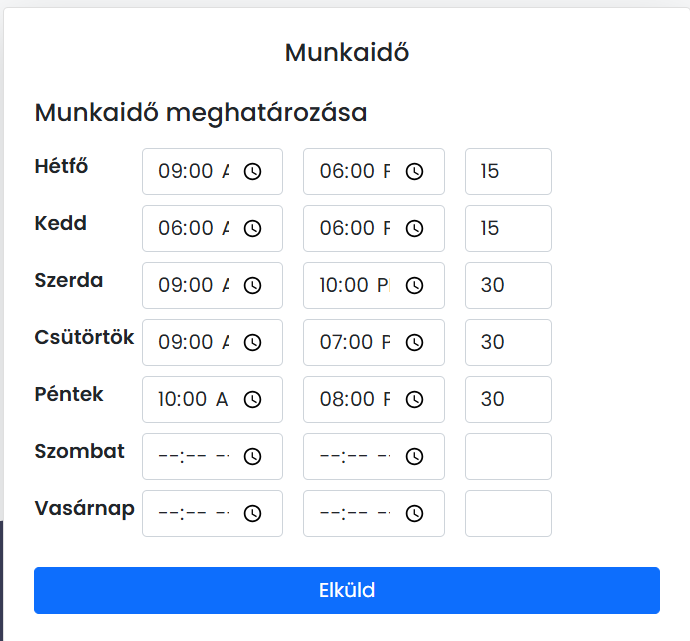
6.Kép: Időpont foglalás oldal



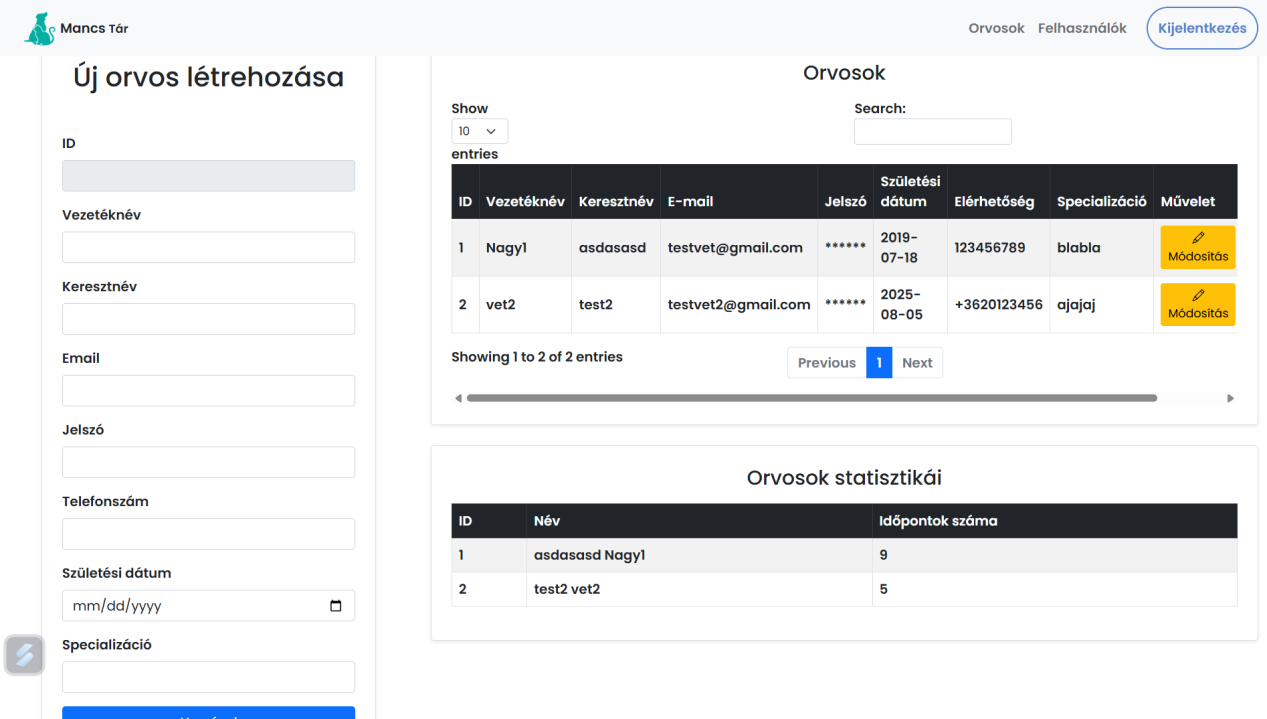
7.Kép: Orvos időpontjai oldal



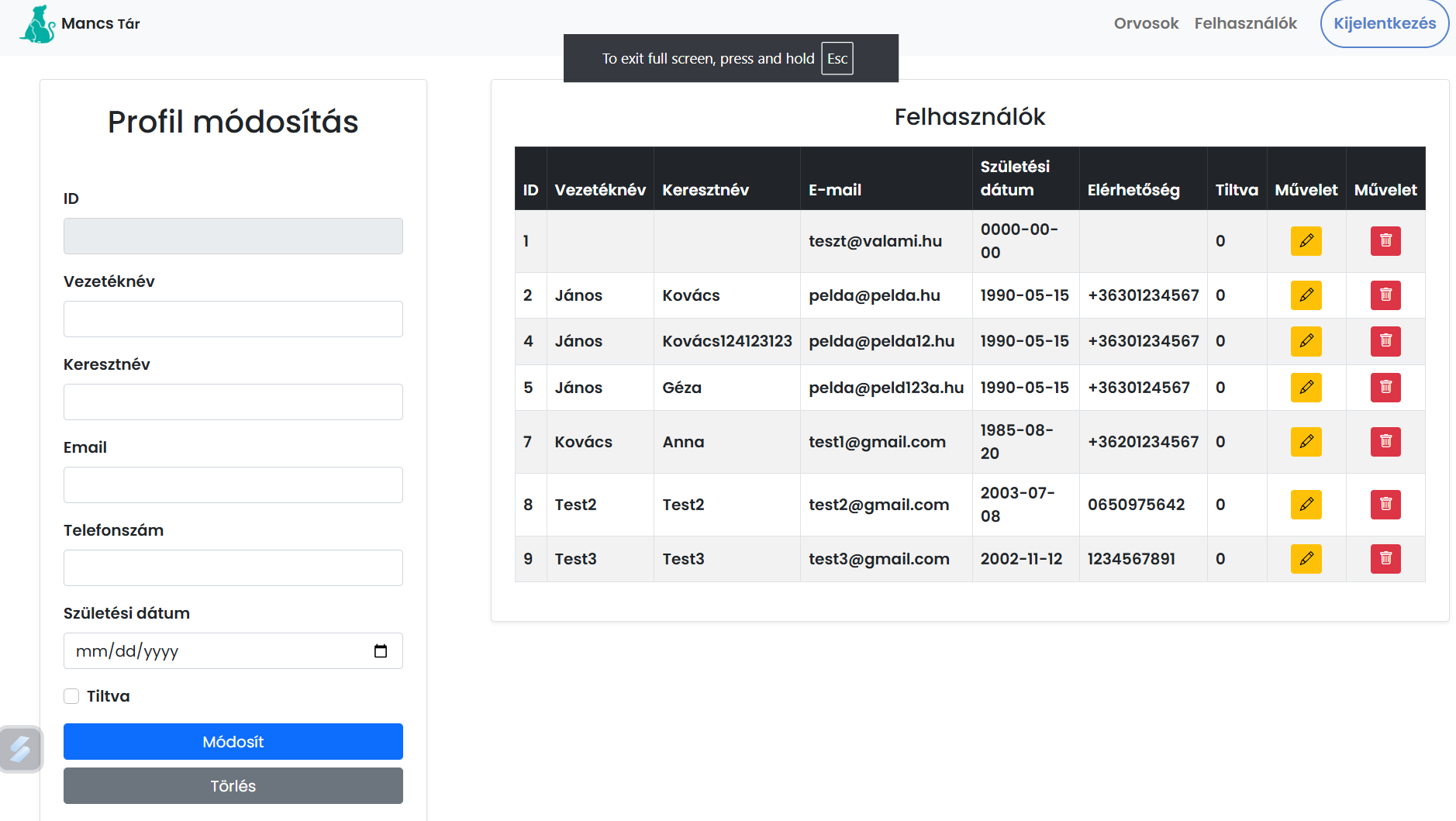
8.Kép: Időpont szerkesztés oldal



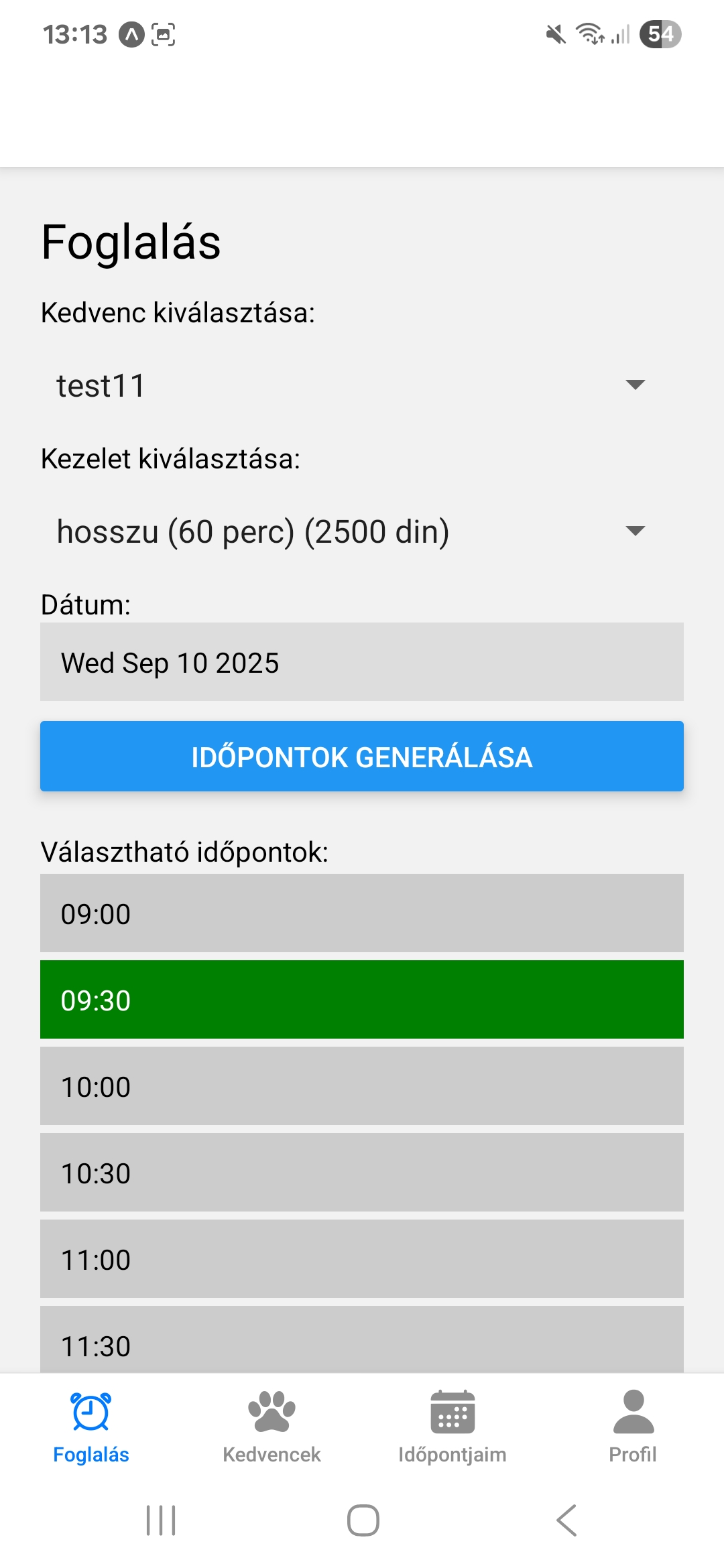
9.Kép: Orvos munkaidő módosítás oldal



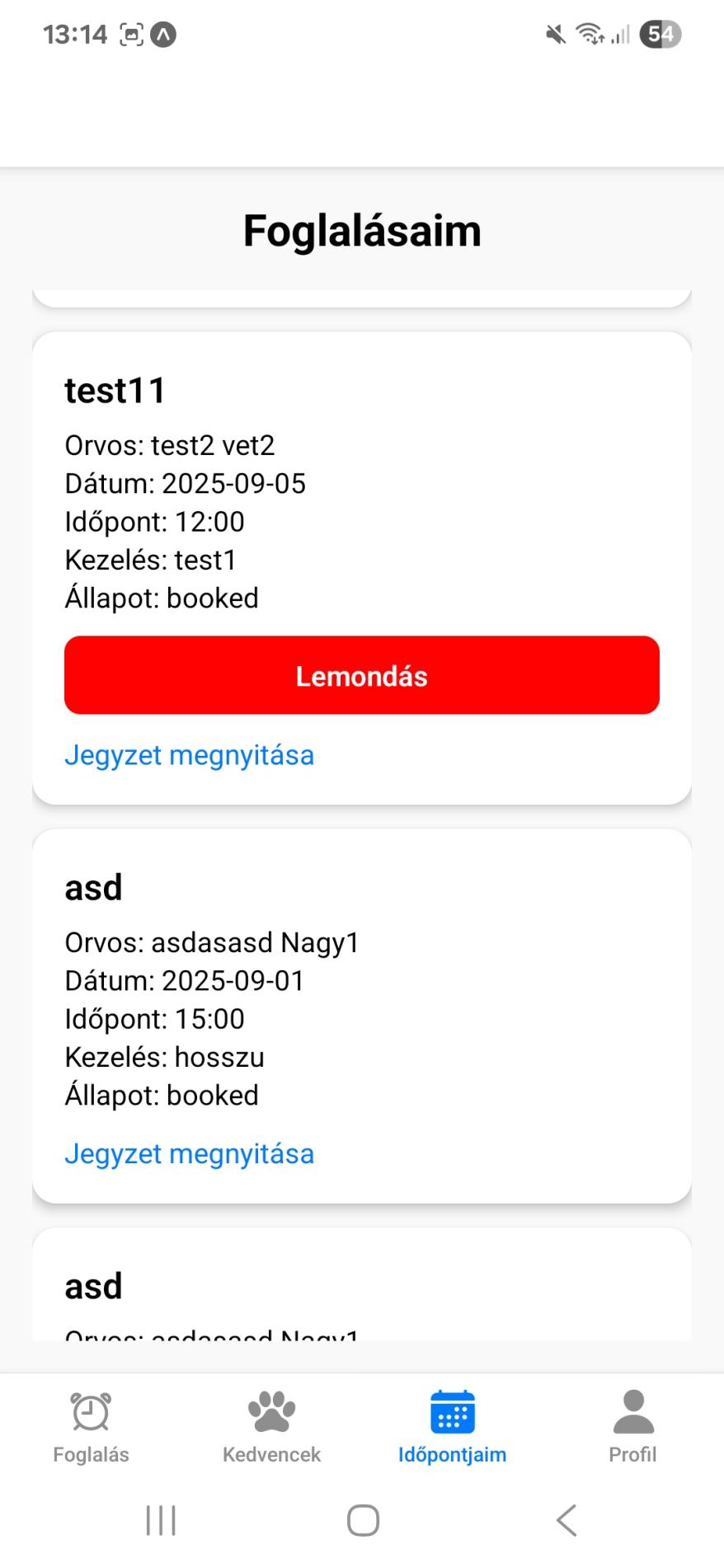
10.Kép: Admin orvos kezelő oldal



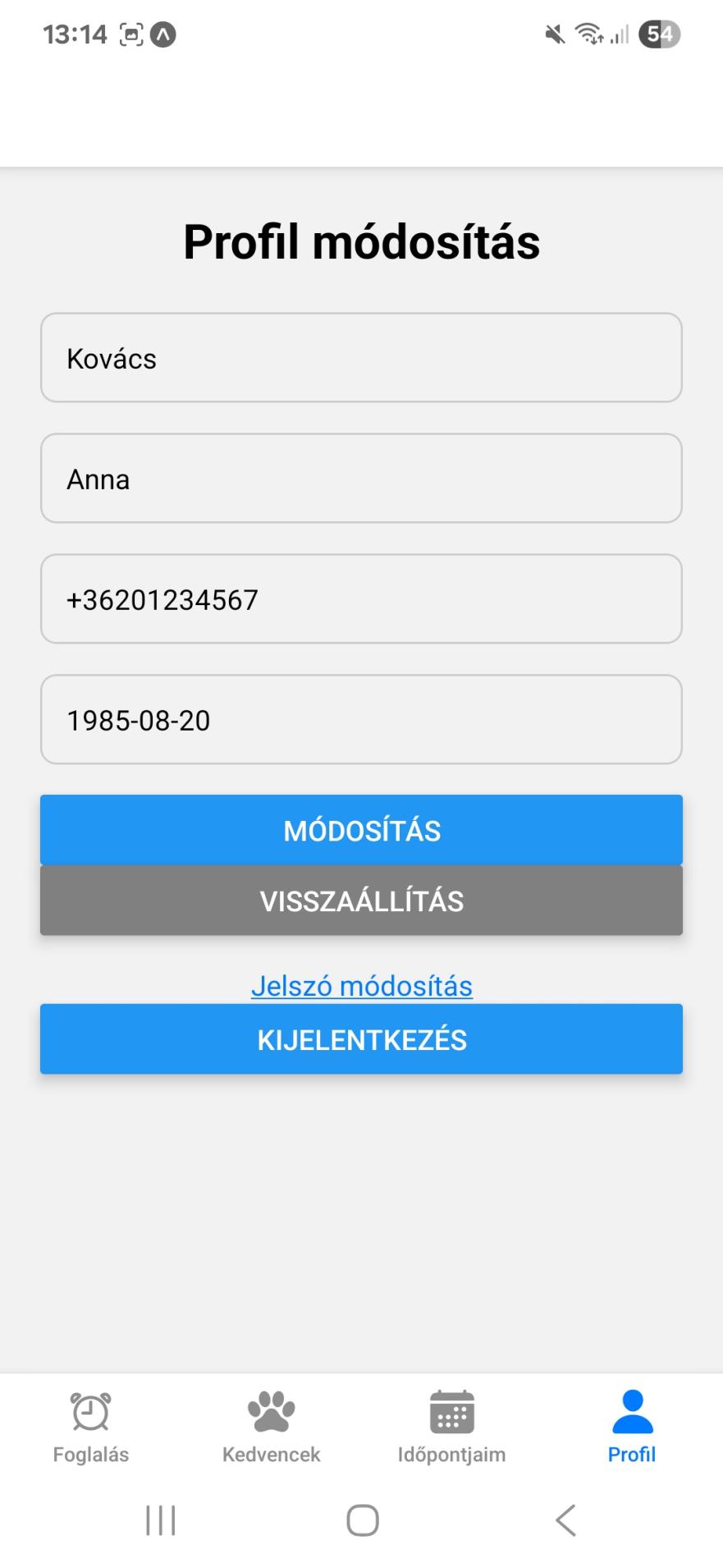
11.Kép: Admin felhasználó kezelő oldal

****

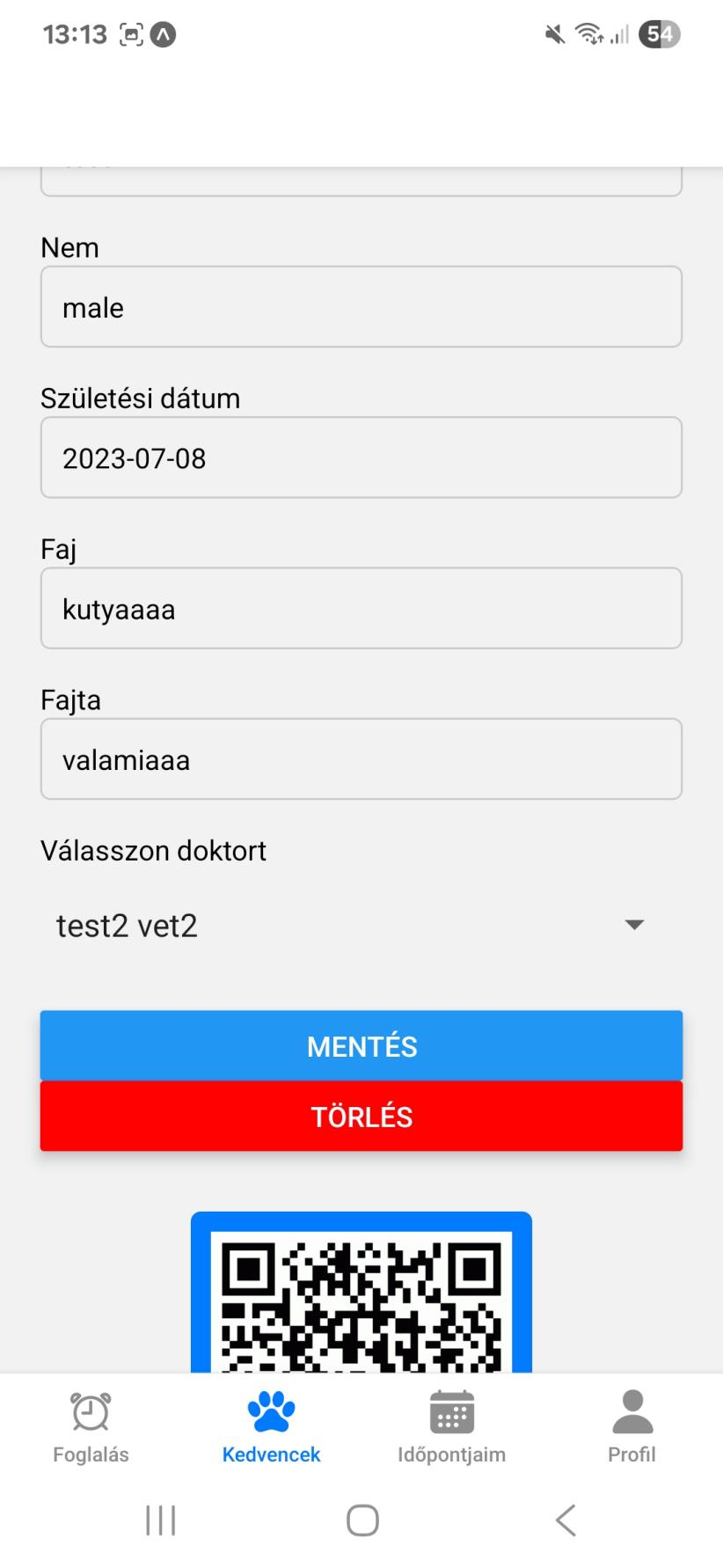
12.Kép: Mobil app foglalás



13.Kép: Mobil app időpontjaim



14.Kép: Mobil app profil módosítás



15.Kép: Mobil app házi állataim

## FELHASZNÁLT IRODALOM

A felhasznált irodalomban a szerzők nevét ábécé sorrendben kell feltüntetni. Példa:

1. Matt Zandstra, “PHP 8 objekti, obrasci i praksa objektno orijentisan pristup”, Kompjuter biblioteka, Beograd, 2021
2. Robin Nixon, “Naučite PHP, MySQL i JavaScript: razvoj dinamičkih veb strana korak po korak prevod 6. izdanja”, Kompjuter biblioteka, Beograd, 2021

W1. <https://reactnative.dev/docs/tutorial>

W2. <https://phptherightway.com/>