HAPIVET Recruitment Process - Assessment Submission

Team Name: Vetcare

Team Members :

Mudavath Kalyan
Malavath Rahul
Korra Praveen
Keerthi

1. Problem Statement

Develop a web-based Pet Appointment Booking System that allows customers to book appointments with veterinarians. Doctors can set their available time slots and approve or reject appointments. The system should provide authentication for both doctors and customers, manage pet details, and display appointment status in real-time. The project aims to streamline veterinary appointment management using the MERN stack.

2. Tools and Technologies Used

• Frontend: React.js, Material UI
• Backend: Node.js, Express.js
• Database: MongoDB (Mongoose ORM)
• Authentication: JWT (JSON Web Token)
• Notifications: Email (Nodemailer)
• Validation: Express-Validator

3. Code Implementation (Team Vetcare)

a) System Roles

• Customer: Register/Login, Add Pet Details, View Doctors, Book Appointments, View Status
• Doctor: Register/Login, Set Available Slots, Approve/Reject Appointments
• Admin (optional): Manage users, doctors, and appointments

b) System Flow

1. Customer registers or logs in to obtain a JWT.
2. Customer adds pet details (name, type, age, gender, description).

3. Customer selects a doctor and views available slots.
4. Customer books an appointment; backend saves appointment with status 'Pending'.
5. Doctor is notified of the booking and can approve or reject the request.
6. Customer sees the updated appointment status.

c) Database Schema Design

User Model (Customer & Doctor):
const userSchema = new mongoose.Schema({ name, email, password, role, specialization, availableSlots });

Pet Model:
const petSchema = new mongoose.Schema({ ownerId, name, type, age, gender, description });

Appointment Model:
const appointmentSchema = new mongoose.Schema({ customerId, doctorId, petId, slot, status, createdAt });

d) API Endpoints

Auth Routes:
POST /api/auth/register – Register new user
POST /api/auth/login – Login and obtain JWT

Doctor Routes:
GET /api/doctor – Get all doctors
GET /api/doctor/:id/slots – Get doctor's available slots
PUT /api/doctor/:id/slots – Update available slots
GET /api/doctor/:id/appointments – Get appointment requests
PUT /api/appointment/:id/status – Approve/Reject appointment

Customer Routes:
POST /api/pet – Add pet details
POST /api/appointment – Book an appointment
GET /api/appointment/:id – Get appointment status

e) Frontend Flow (React)

• Login and Register pages handle authentication.
• PetForm.js allows customers to add pet details.
• DoctorList.js displays available doctors.
• AvailableSlots.js shows available appointment slots.
• BookAppointment.js allows customers to book appointments.
• DoctorDashboard.js lets doctors manage appointment requests.

f) Security and Validation

• Passwords hashed using bcrypt.
• Protected routes via JWT middleware.
• Input validation with express-validator.
• Only authorized roles can access restricted endpoints.

g) Future Enhancements

• Real-time updates using Socket.io
• Calendar-based slot management
• Online payment integration
• Chat system between doctor and customer