Veterinary Clinic Management System

This document provides a structured overview of the Veterinary Clinic Management system project. It covers functional requirements written as use cases. A REST API should be designed from these use cases, bearing in mind that a use case may need more than one API call.

Domain Description

The domain for our Veterinary Clinic application revolves around the core entities involved in providing healthcare services to animals. We aim for a realistic representation of how a clinic operates, from scheduling to treatment and billing. You may need additional entities, and the ones appearing in the document are not mandatory.

Core Entities:

1. Person: The foundational entity for anyone interacting with the clinic, whether as a service provider or a client.
   - Attributes: personId (unique identifier), firstName, lastName, phoneNumber, email, address.
2. PetOwner (inherits from Person): Individuals who own pets and bring them to the clinic for care.
   - Attributes: ownerId (unique), loyaltyPoints (for fidelity program), loyaltyTier (e.g., Bronze, Silver, Gold).
   - Relationships: Owns one or more Pets. Receives Invoices.
3. Pet: The animal patients of the clinic.
   - Attributes: petId (unique), name, dateOfBirth, gender, breed, color, weight, microchipId (optional).
   - Relationships: Owned by a PetOwner. Has a PetType. Has many Visits. Can have many Treatments and Medications prescribed over time.
4. PetType: A classification for pets, important for understanding common ailments and specific care needs.
   - Attributes: typeId (unique), name (e.g., Dog, Cat, Bird, Reptile, Rabbit, Guinea Pig), description.
5. Veterinarian (inherits from Person): The medical professionals providing care.
   - Attributes: veterinarianId (unique), licenseNumber, yearsOfExperience.
   - Relationships: Has one or more Specialitys. Performs Visits and prescribes Treatments/Medications.
6. Speciality: Defines areas of expertise for veterinarians.
   - Attributes: specialtyId (unique), name (e.g., General Practice, Surgery, Dermatology, Oncology, Exotics, Cardiology).
   - Relationships: A Veterinarian can have multiple Specialitys.
7. Availability: Represents the availability of a Veterinarian in a day of the week during a period of time (initial and final date). During this period there may be exceptions, for given calendar days, where the Veterinarian is not available
   - Attributes: dayOfWeek, data interval (initial and final period dates)

   - Relationships: A Veterinarian can have multiple AvailabilityS, each Availability can have multiple exceptions

8. ClinicService: Represents distinct services offered by the clinic beyond just consultations.
   - Attributes: serviceId (unique), name (e.g., "Consultation", "Vaccination", "Spay/Neuter Surgery", "Dental Cleaning", "X-Ray", "Blood Test"), basePrice, description.
9. Agenda: Represents all the scheduled VisitS. Usually, a visit has a default duration of 15 minutes, and can be enlarged in periods of 15 minutes. The Agenda is in charge to schedule visits according to the availability of VeterinarianS
   - Attributes:
   - Relationships: Schedules, VeterinarianS of the clinic
10. Visit: Represents a scheduled visit in the future or ad-hoc interaction between a pet, its owner, and a veterinarian. A Visit can be rescheduled, and we want to keep track of the reschedulings.
    - Attributes: date and time, duration, reason for visit, price per 15 minutes bloc
    - Relationships: one Pet that is visited, one PetOwner, may have one rescheduling
11. Veterinary Patient Record (VPR) : Represents the report of a visit. It is a base to make invoices
    - Attributes: visitId (unique), dateTime, reasonForVisit, diagnosis, notes, status (e.g., Scheduled, Completed, Cancelled, In Progress), duration.
    - Relationships: Involves one PetOwner, one Pet, one Veterinarian. Can have multiple Treatments and MedicationPrescriptions. Leads to one Invoice.
12. Treatment: A medical procedure or therapy applied to a pet during or after a visit.
    - Attributes: treatmentId (unique), name (e.g., "Wound Cleaning", "Fluid Therapy", "Suture Removal"), description, cost.
    - Relationships: Associated with a Visit. Can be associated with InvoiceItems.
13. Medication: A specific drug or pharmaceutical product type.
    - Attributes: medicationId (unique), name, activeIngredient, dosageUnit, unitPrice (for selling).
    - Relationships: Can be prescribed via MedicationPrescription.
14. MedicationBatch: A specific delivery or lot of a Medication type, with its own unique expiry and quantity.
    - Attributes: batchId, medicationId, lotNumber, receivedDate, expiryDate, initialQuantity, currentQuantiy, purchagePricePerUnit, storageLocation, reorderThreshold
    - Relationships: Can be sold directly. If sold can be included to an existing invoice or create a specific one.
15. MedicationPrescription: Records a specific medication prescribed during a visit.
    - Attributes: prescriptionId (unique), quantityPrescribed, dosageInstructions, duration.
    - Relationships: Links a Visit to a Medication.

16. Invoice: The bill generated for services, treatments, and medications.
    - Attributes: invoiceId (unique), invoiceDate, totalAmount, discountAmount, finalAmount, status (e.g., Unpaid, Paid, Partially Paid).
    - Relationships: Associated with a PetOwner. Generated from zero or more Visits. Contains InvoiceItems. Records Payments.
17. InvoiceItem: Represents a single line item on an invoice (e.g., a service, a specific treatment, a medication).
    - Attributes: itemId (unique), description, quantity, unitPrice, itemTotal.
    - Relationships: Links to a ClinicService, Treatment, or Medication.
18. Payment: Records a financial transaction for an invoice.
    - Attributes: paymentId (unique), paymentDate, amount, paymentMethod, transactionRef.
    - Relationships: Associated with an Invoice.
19. Discount: Represents various types of price reductions.
    - Attributes: discountId (unique), code (for promotional codes), type (e.g., Percentage, Fixed Amount, Loyalty Tier), value, startDate, endDate, maxUses, usesCount.
    - Relationships: Can be applied to an Invoice. Promotions and LoyaltyTiers grant Discounts.
20. Promotion: Specific marketing campaigns offering discounts.
    - Attributes: promotionId (unique), name, description, discountCode (if applicable), startDate, endDate.
    - Relationships: Grants Discounts.
21. LoyaltyTier: Defines levels within the fidelity program for PetOwners.
    - Attributes: tierName, requiredPoints, discountPercentage, benefitsDescription.
    - Relationships: A PetOwner belongs to one LoyaltyTier. Grants Discounts.
22. Administrator (inherits from Person): Staff managing the system (e.g., clinic managers, receptionists).
    - Attributes: role (e.g., "Receptionist", "Clinic Manager").
    - Permissions: Manages appointments, invoices, patron/pet registration, discount/promotion configuration.

Use Cases for Veterinary Clinic Management System

We will divide the use cases into the following three parts: (1) Schedule Visits, (2) Track Visits & Treatments, and (3) Invoicing & Sales.

Part 1: Schedule Visits

These use cases focus on the booking and management of appointments.

1. UC1.1: Schedule New Visit
* Description: A receptionist or an owner (via a portal) schedules a new appointment for a pet with a specific veterinarian.
* Actors: Receptionist, PetOwner (if self-service portal exists).
* Pre-conditions: Pet and PetOwner are registered. Veterinarian has available slots.
* Post-conditions: A new Visit record is created with a 'Scheduled' status.
* Flow:

    1. User initiates "Schedule New Visit."
    2. User selects/searches for PetOwner and Pet.
    3. User selects desired Veterinarian or speciality (optional, can be assigned later) and preferred date/time range.
    4. System displays available time slots for the chosen veterinarian (or any available if no vet specified).
    5. User selects a time slot and enters reasonForVisit.
    6. System creates the Visit record and sends a confirmation (email/SMS) to the PetOwner.
    7. The initial duration of a visit is 15 minutes, but can be extended

2. UC1.2: View Veterinarian's Schedule
* Description: A receptionist or veterinarian views a veterinarian's daily or weekly appointment schedule.
* Actors: Receptionist, Veterinarian.
* Pre-conditions: Veterinarians and appointments exist.
* Post-conditions: The system displays the chosen veterinarian's schedule.
* Flow:

    1. User selects "View Schedule."
    2. User chooses a Veterinarian and a date range.
    3. System retrieves and displays all Visits for that veterinarian within the specified period along with his/her available time slots.

3. UC1.3: Reschedule Visit
* Description: A receptionist or owner changes the date and/or time of an existing scheduled visit.
* Actors: Receptionist, PetOwner.
* Pre-conditions: Visit exists with 'Scheduled' status. New slot is available.
* Post-conditions: The Visit record's dateTime is updated and change recorded in a Log
* Flow:

    1. User finds the Visit to be rescheduled.
    2. User selects "Reschedule."

3. System displays available time slots (similar to UC1.1).
4. User selects a new slot.
5. System updates the Visit record and also stores the rescheduling event in a History or Audit Log class, and sends confirmation/notification.

4. UC1.4: Cancel Scheduled Visit
* Description: A receptionist or owner cancels a scheduled visit.
* Actors: Receptionist, PetOwner.
* Pre-conditions: Visit exists with 'Scheduled' status.
* Post-conditions: The Visit record's status is updated to 'Cancelled', and the event is stored in a History or Audit Log class.
* Flow:

1. User finds the Scheduled Visit to be cancelled.
2. User selects "Cancel Visit."
3. System prompts for confirmation and (optional) reason for cancellation.
4. System updates the Visit status to 'Cancelled' and frees up the time slot.
5. System stores the cancelation event to a History or Audit Log class

5. UC1.5: Register Walk-in Visit
* Description: A receptionist records an unscheduled, immediate visit for a pet.
* Actors: Receptionist.
* Pre-conditions: A veterinarian is available. Pet and PetOwner are registered (or new ones will be registered).
* Post-conditions: A new Visit record is created, possibly with an 'In Progress' status.
* Flow:

1. Receptionist selects "Register Walk-in."
2. Receptionist identifies/registers PetOwner and Pet.
3. Receptionist assigns an available Veterinarian.
4. System creates a Schedule record for the current time.

6. UC1.6: Owner didn’t show up
* Description: the owner does not show up with his pet to a visit
* Actors: Receptionist / system
* Pre-conditions: The Visit is scheduled and 10 minutes after the due dateTime the owner has not showed up with his pet
* Post-conditions: The Visit record’s state is updated to “Not Showed Up”
* Flow:

1. Receptionist selects the Visit the owner of which has not showed up
2. Receptionist updates the Visit status to “Not Showed Up”

7. UC1.7 Manage Veterinarian Availability
* Description: The administrator defines a veterinarian's regular working hours for each day of the week and can add specific date-based exceptions (e.g., holidays, personal leave, conferences, or specific extended hours) that override the general availability for a given day or period. This ensures the system accurately reflects when a veterinarian is available for appointments.
* Actors: Receptionist / administrator
* Pre-conditions: The veterinarian is registered in the system

* Post-conditions: The scheduling system accurately reflects the Veterinarian's availability for booking new appointments.
* Flow:
    1. Administrator specifies the veterinarian
    2. The system shows the veterinarian’s availability and exceptions
    3. Define availabilities: for each day of the week define periods of time the veterinarian is available
    4. Define exceptions: for a calendar day, define hours the veterinarian is not available. Also can define a reason.
    5. Steps 3 and 4 can be performed to update or delete availabilities and exceptions

8. UC1.8: List specialities ordered by demand in a given period
* Description: The administrator can list the specialities ordered by demand during a given period. Demand is calculated based in the number of scheduled visits. The number is shown in the list.
* Actors: administrator
* Pre-conditions: schedule data exists in the system
* Post-conditions: administrator is aware of the demand of the specialities
* Flow:
    1. Administrator accesses the “Specialities’ demand” list
    2. Administrator introduces the period
    3. The system generates the list

9. UC1.9: List veterinarians ordered by demand in a given period
* Description: The administrator can list the veterinarians ordered by demand during a given period. Demand is calculated based in the number of scheduled visits. The number is shown in the list.
* Actors: administrator
* Pre-conditions: schedule data exists in the system
* Post-conditions: administrator is aware of the demand of the veterinarians
* Flow:
    1. Administrator accesses the “Veterinarians demand” list
    2. Administrator introduces the period
    3. The system generates the list

Part 2: Keep Track of Visits and Treatments

These use cases focus on recording the medical aspects of a visit.

1. UC2.1: Start/Complete Visit Consultation
* Description: A veterinarian or assistant marks a visit as 'In Progress' when the consultation begins and 'Completed' when it ends, allowing for medical records to be attached.
* Actors: Veterinarian, Vet Assistant.
* Pre-conditions: Visit exists with 'Scheduled' or 'In Progress' status.
* Post-conditions: Visit status is updated. Time of start/completion may be recorded.
* Flow:
    1. Veterinarian accesses their schedule.

2. Veterinarian selects a Visit and marks it "Start Consultation."
3. (After consultation) Veterinarian marks the Visit "Complete Consultation."

2. UC2.2: Record Visit Diagnosis and Notes

* Description: A veterinarian enters the diagnosis, examination findings, and general notes for a pet's visit.
* Actors: Veterinarian.
* Pre-conditions: Visit is 'In Progress' or 'Completed'.
* Post-conditions: Diagnosis and Notes attributes of the Visit are updated.
* Flow:
    1. Veterinarian selects an 'In Progress' or 'Completed' Visit.
    2. Veterinarian enters diagnosis (e.g., "Canine Distemper"), observations, and any relevant notes.
    3. System saves the medical record associated with the Visit.

3. UC2.3: Prescribe Medication

* Description: A veterinarian prescribes one or more medications to a pet during a visit.
* Actors: Veterinarian.
* Pre-conditions: Visit is 'In Progress' or 'Completed'. Medications exist in stock.
* Post-conditions: MedicationPrescription records are created, linked to the Visit and Medication.
* Flow:
    1. Veterinarian accesses an In Progress or Completed Visit.
    2. Veterinarian selects "Prescribe Medication."
    3. Veterinarian searches for and selects desired Medication from available stock.
    4. Veterinarian enters quantityPrescribed, dosageInstructions, and duration.
    5. System updates stock (see UC2.8).
    6. System creates a MedicationPrescription record.
    7. (Repeat for multiple medications).

4. UC2.4: Prescribe/Record Treatment

* Description: A veterinarian records that a specific treatment was performed or is to be performed for a pet during a visit.
* Actors: Veterinarian.
* Pre-conditions: Visit is 'In Progress' or 'Completed'. Treatments are defined in the system.
* Post-conditions: Treatment record is associated with the Visit.
* Flow:
    1. Veterinarian accesses an 'In Progress' or 'Completed' Visit.
    2. Veterinarian selects "Add Treatment."
    3. Veterinarian searches for and selects the Treatment (e.g., "Vaccination," "Suture Removal").
    4. System associates the chosen Treatment with the Visit.
    5. (Repeat for multiple treatments).

5. UC2.5: View Pet's Medical History

* Description: A veterinarian or receptionist views a pet's complete medical history, including all past visits, diagnoses, treatments, and prescribed medications.
* Actors: Veterinarian, Receptionist.
* Pre-conditions: Pet exists and has past visits.
* Post-conditions: System displays a chronological list of the pet's medical events.
* Flow:

    1. User searches for a Pet.
    2. User selects "View Medical History."
    3. System retrieves all associated Visits, Treatments, and MedicationPrescriptions for the pet.
    4. System displays the history in an organized, chronological manner.

6. UC2.6: List medication ordered by prescription in a given period

* Description: The administrator can list the medication ordered by the number of times it is prescribed during a given period. The number of prescriptions is shown in the list.
* Actors: administrator
* Pre-conditions: prescription data exists in the system
* Post-conditions: administrator is aware of the most prescribed medication
* Flow:

    1. Administrator accesses the “Medication prescription” list
    2. Administrator introduces the period
    3. The system generates the list

7. UC2.7: List veterinarians ordered by prescription of a medication in a given period

* Description: The administrator can list the veterinarians ordered by the number of times he or she prescribed a given medication during a given period. The number of prescriptions is shown in the list.
* Actors: administrator
* Pre-conditions: prescription data exists in the system
* Post-conditions: administrator is aware of the veterinarians’ prescriptions
* Flow:

    1. Administrator accesses the “Veterinarians’ prescription” list
    2. Administrator introduces the medication
    2. Administrator introduces the period
    3. The system generates the list

8. UC2.8: Dispense Medication from Stock

* Description: A veterinarian or receptionist records the dispensing or sale of medication to a pet owner, reducing the quantity from the appropriate MedicationBatch (preferably the one expiring soonest). We may not need a REST entry point for this UC
* Actors: Veterinarian, Receptionist.
* Pre-conditions: Medication is being prescribed (UC2.3) or sold directly (UC3.2). Sufficient MedicationBatch currentQuantity is available and not expired.
* Post-conditions: The currentQuantity of the selected MedicationBatch is decreased. An InvoiceItem is created.
* Flow (Integrated with UC2.3/UC3.2, but detailing stock aspect):

1. (From UC2.3 Prescribe Medication or UC3.2 Sell Medication) User selects the Medication and quantity.
2. System retrieves available MedicationBatches for that Medication type, sorted by expiryDate (earliest first).
3. System prompts user to select a batchId to dispense from, or automatically selects the earliest expiring non-empty batch.
4. System verifies if currentQuantity in the chosen MedicationBatch is sufficient.
5. If sufficient, System reduces the currentQuantity of the selected MedicationBatch by the dispensed quantity.
6. (If currentQuantity of a batch becomes 0, mark it as 'depleted').
7. System creates the MedicationPrescription or InvoiceItem as per the main use case.

9. UC2.9: Receive New Medication Batch

* Description: An administrator or inventory manager records the receipt of a new delivery of medication, updating stock levels by adding a new batch with its specific expiry date.
* Actors: Administrator, Inventory Manager.
* Pre-conditions: The Medication type already exists in the system. The delivery has arrived physically.
* Post-conditions: A new MedicationBatch record is created, increasing the total available stock for that medication type.
* Flow:

1. User selects "Receive New Medication Batch."
2. User searches for and selects the Medication type (e.g., "Amoxicillin 250mg Tablets").
3. User inputs batch details: lotNumber, expiryDate, initialQuantity (and currentQuantity set to this value), purchasePricePerUnit, receivedDate.
4. User confirms the entry.
5. System validates data (e.g., expiryDate is in the future).
6. System creates the new MedicationBatch record and adds it to the inventory.

10. UC2.10: Receive Low Stock Alert

* Description: The system automatically notifies relevant personnel (e.g., Inventory Manager, Administrator) when the total currentQuantity of a Medication type falls below a predefined reorder threshold.
* Actors: System.
* Pre-conditions: Reorder thresholds are configured for Medications. currentQuantity is updated regularly (e.g., after dispensing/sales).
* Post-conditions: Notification is sent.
* Flow:

1. (Scheduled Task or triggered by UC2.8) System periodically checks or immediately checks Medication's total currentQuantity.
2. If total currentQuantity for a Medication falls below its configured reorder threshold.
3. System generates a low stock alert (e.g., email, in-app notification) to the designated user(s).

4. System logs the alert to prevent repeated notifications for the same event.

11. UC2.11: Manage Medication incompatibilities
* Description: The administrator defines incompatibilities between medications when they cannot be taken at the same time. An incompatibility can persist for a specified period.
* Actors: Administrator or veterinary
* Pre-conditions: Medications must exist in the system
* Post-Conditions: a new medication incompatibility is registered in the system. Only one incompatibility can be defined for a given pair of medications.
* Flow:

1. Administrator lists available medications
2. Administrator selects two medications
3. Administrator states a persisting period if applicable
4. Administrator can delete or modify incompatibilities

12. UC2.12: Medication incompatibility alert
* Description: given a pet and a medication, the system returns whether this medication has any incompatibilities with the ones the pet is taking or has taken recently.
* Actors: Veterinary
* Pre-conditions: pet and medication are on the system
* Post-conditions: an alarm is signaled when there exists an incompatibility. You may need to call this use case from 2.3 (prescribe medication)
* Flow:

1. When a veterinarian prescribes a medication or makes a consultation
2. Pet and medication are introduced
3. The system returns true if there is an incompatibility, or false otherwise

Part 3: Generation of Invoices, Sales, Discounts, Promotions, Fidelity Points

These use cases cover all financial aspects, from billing to loyalty programs.

1. UC3.1: Generate Invoice from Visit
* Description: A receptionist or system automatically generates an invoice for a completed visit, including all associated services, treatments, and prescribed medications.
* Actors: Receptionist, System.
* Pre-conditions: Visit is 'Completed'. ClinicServices, Treatments, and Medications have defined costs/prices.
* Post-conditions: A new Invoice record is created, pre-populated with InvoiceItems.
* Flow:

1. Receptionist selects a 'Completed' Visit and chooses "Generate Invoice."
2. System retrieves the ClinicService (e.g., "Consultation fee"), all Treatments performed, and MedicationPrescriptions.
3. System creates InvoiceItems for each, calculating their cost.

4. System sums up totalAmount and creates the Invoice record with 'Unpaid' status.

2. UC3.2: Sell Medication/Products (Non-Visit Sale)
* Description: A receptionist sells over-the-counter medication or other clinic products directly to a pet owner, independent of a visit.
* Actors: Receptionist.
* Pre-conditions: Medication or other Products exist in stock with defined prices.
* Post-conditions: An Invoice is generated for the sale. Medication stockQuantity is reduced.
* Flow:

1. Receptionist initiates "New Sale."
2. Receptionist searches for and adds Medications/Products to a virtual cart, specifying quantities.
3. System calculates the subtotal.
4. Receptionist can associate the sale with a PetOwner (optional for loyalty points).
5. System updates stock (see UC2.8)
6. System generates an Invoice for these items and updates stockQuantity.

3. UC3.3: Apply Discount to Invoice (Manual/Code)
* Description: A receptionist manually applies a specific Discount (e.g., staff discount, promotional code) to an invoice.
* Actors: Receptionist.
* Pre-conditions: Invoice has been generated. Discount code/reason is known.
* Post-conditions: Invoice discountAmount and finalAmount are updated.
* Flow:

1. Receptionist views an Invoice.
2. Receptionist selects "Apply Discount" and enters a Discount code or selects a manual discount type.
3. System validates the Discount (existence, validity, usage limits).
4. If valid, system calculates the discount, updates the Invoice's amounts, and (if code-based) increments usesCount for the Discount. A new InvoiceItem is created with the discountAmount. There may be rules limiting the quantity of discounts or Loyalty Tier can be applied or a maximum of quantity/percentage can be applied.

4. UC3.4: Apply Loyalty Tier Discount (Automatic)
* Description: The system automatically applies a discount to an invoice based on the PetOwner's current LoyaltyTier.
* Actors: System.
* Pre-conditions: Invoice is being generated or viewed. PetOwner has an assigned LoyaltyTier with an associated Discount.
* Post-conditions: Invoice discountAmount and finalAmount are automatically adjusted.
* Flow:

1. System generates an Invoice for a PetOwner (UC11).
2. System retrieves the PetOwner's loyaltyTier.
3. System checks if the LoyaltyTier has an automatic discountPercentage.

4. If yes, system automatically applies this percentage to the totalAmount and sets discountAmount, updating finalAmount. A new InvoiceItem is created with the discountAmount. There may be rules limiting the quantity of discounts or Loyalty Tier can be applied or a maximum of quantity/percentage can be applied.

5. UC3.5: Earn Fidelity Points (automatic)
* Description: When an Invoice is paid, PetOwners earn fidelity points based on the finalAmount spent.
* Actors: System.
* Pre-conditions: Invoice is marked as 'Paid'. PetOwner is enrolled in the fidelity program. Fidelity rules are configured (e.g., 1 point per €10 spent).
* Post-conditions: PetOwner's fidelityPoints balance is updated. LoyaltyTier might be re-evaluated.
* Flow:

    1. System detects an Invoice has been paid (UC16).
    2. System retrieves the finalAmount and the associated PetOwner.
    3. System calculates points earned based on configured rules.
    4. System adds points to the PetOwner's fidelityPoints balance.
    5. System re-evaluates the PetOwner's loyaltyTier based on new point total.
    6. (Optional) System notifies PetOwner of points earned.

6. UC3.6: Process Invoice Payment
* Description: A receptionist records the payment received for an invoice, updating its status.
* Actors: Receptionist.
* Pre-conditions: Invoice has an 'Unpaid' or 'Partially Paid' status.
* Post-conditions: Invoice status is updated to 'Paid' or 'Partially Paid'. A Payment record is created. Triggers UC3.5.
* Flow:

    1. Receptionist selects an Invoice to be paid.
    2. Receptionist enters amount paid and paymentMethod (e.g., cash, credit card, bank transfer).
    3. System creates a Payment record linked to the Invoice.
    4. System updates the Invoice status ('Paid' if finalAmount is covered, 'Partially Paid' otherwise).
    5. System triggers UC3.5 (Earn Fidelity Points).

7. UC3.7: Redeem Fidelity Points for Discount
* Description: A PetOwner (or receptionist on their behalf) uses accumulated fidelity points to get a discount on a new service or an outstanding invoice.
* Actors: PetOwner, Receptionist.
* Pre-conditions: PetOwner has sufficient fidelityPoints. Redemption rules are configured (e.g., 100 points = €5 discount).
* Post-conditions: Invoice finalAmount is reduced. PetOwner's fidelityPoints are deducted.
* Flow:

    1. User selects an Invoice or is about to pay for a new service.
    2. User selects "Redeem Points."

3. System displays available redemption options based on PetOwner's fidelityPoints.
4. User selects the desired point redemption amount.
5. System calculates the corresponding discount, applies it to the Invoice (similar to UC13).
6. System deducts the points from the PetOwner's fidelityPoints balance.

8. UC3.8: Configure Promotions & Discounts

* Description: An administrator creates and manages various Promotions and Discounts (e.g., "First Visit 10% Off," "Vaccination Drive Discount," specific product sale).
* Actors: Administrator.
* Pre-conditions: User has administrative privileges.
* Post-conditions: New Promotion and Discount records are created and active in the system.
* Flow:

1. Administrator accesses "Promotions & Discounts Management."
2. Administrator creates a new Promotion (e.g., "Summer Wellness Check").
3. Administrator defines associated Discounts (e.g., 15% off Wellness Check ClinicService).
4. Administrator sets validity period (startDate, endDate), usage limits (maxUses), and code (if applicable).
5. System saves the configuration.

9. UC3.9: Define Loyalty Tiers/Levels

* Description: The system allows administrators to define different loyalty tiers (e.g., Bronze, Silver, Gold) based on accumulated points or spending, with associated benefits.
* Actors: Administrator.
* Pre-conditions: None.
* Post-conditions: Loyalty tiers and their rules are configured in the system.
* Flow:

1. Administrator accesses "Loyalty Program Configuration."
2. Administrator defines tier names, point thresholds, and associated benefits (e.g., % discount, free check-up).
3. System saves the tier configurations

10. UC3.10: View Invoice History

* Description: A PetOwner or administrator can view a history of all generated invoices.
* Actors: PetOwner, Administrator/Receptionist.
* Pre-conditions: Invoices exist in the system.
* Post-conditions: User can see a list of past invoices with their status (paid/unpaid).
* Flow:

1. User navigates to "Invoice History" section.
2. System displays a list of invoices, possibly with filters (date range, status, patient).

3. User can select an invoice to view its details.

11. UC3.11: View Fidelity Point Balance

* Description: A PetOwner can view their current fidelity point balance.
* Actors: PetOwner.
* Pre-conditions: Patient has an account and points may have been accumulated.
* Post-conditions: Patient is aware of their current point balance.
* Flow:
    1. Patient logs into their account.
    2. Patient navigates to their "Fidelity Program" or "Profile" section.
    3. System displays the current fidelity point balance.

12. UC3.12: Analyze Discount Utilization

* Description: The system provides analytics on which discounts are most frequently used, their impact on revenue, and popular discount codes.
* Actors: Administrator.
* Pre-conditions: Discount data exists in the system.
* Post-conditions: Administrator gains insights into discount effectiveness.
* Flow:
    1. Administrator accesses "Discount Analytics."
    2. Administrator specifies criteria (e.g., discount type, time frame).
    3. System analyzes discount applications and their financial impact.
    4. System displays trends, most used discounts, and revenue impact.

Separation in 4 Sprints

Sprint 1

- 1.7 Manage Vet availability

- 2.9 Add medication batch
- 2.10 Receive new low stock alert

- 3.8 Configure promotions & discounts
- 3.9 Define loyalty tiers

Sprint 2

- 1.1 Schedule visits
- 1.2 View Vet schedule

- 2.1 Start/complete visit consultation
- 2.2 Record visit diagnosis
- 2.3 Prescribe medication
- 2.4 Prescribe treatment

- 3.2 Sell medication and generate invoice
- 3.4 Loyalty Tier discount
- 3.5 Earn fidelity points

Sprint 3

- 1.3 Reschedule visit
- 1.4 Cancel Scheduled visit
- 1.5 Walk-in visit
- 1.6 Owner didn't show up

- 2.8: Dispense medication from stock
- 2.11: Manage medication incompatibilities
- 2.12: Medication incompatibility alert

- 3.1 Generate invoice from visit
- 3.3 Apply discount to invoice
- 3.6 Process invoice payment
- 3.7 Redeem Fidelity points for a discount

Sprint 4

- 1.8 List specialities ordered by demand in a given period
- 1.9 List veterinarians ordered by demand in a given period

- 2.5 View pet's medical history
- 2.6 List medication ordered by prescription in a period
- 2.7 List veterinarians ordered by prescription of a medication in a period

- 3.10 View invoice history
- 3.11View fidelity point balance
- 3.12 Analyze discount utilization