Project Presentation Report

Our group was tasked with assisting in the development of a Veterinary Information and Management System (VIMS) for a new client, Northwood Animal Care. Our objective was to cover the first three phases of the SDLC (Planning, Analysis, & Design).

Our client is a local veterinary clinic that offers services such as routine checkups ,vaccinations, surgeries, and emergency care. Their current system consists of a combination of various types of business software and spreadsheets. The client reported that a lack of integration in the current system is leading to data management errors which in turn are impacting customer service and inventory tracking. They are looking to implement an integrated system with high maintainability that will help resolve these issues.

Interview Questions

To understand the client's current appointment management process and identify pain points for designing a new system, we asked a plethora of questions, the main ones are as follows:

- **Question 1:** Can you walk us through your entire appointment management process, including the tools and systems used, and any challenges faced?
- **Question 2:** What are the most time-consuming or repetitive tasks?
- **Question 3:** How do you currently communicate with pet owners about appointments, and which methods are most effective?

The current appointment management process relies heavily on manual efforts, using phone calls, in-person visits, and physical records like spreadsheets and appointment books. This manual approach leads to inefficiencies and challenges, including potential errors in scheduling and difficulties in locating up-to-date documentation. Staff spend significant time on repetitive tasks such as searching for information and manually verifying details, which can cause delays and miscommunication.

Phone calls are the primary and most effective method for communicating with pet owners regarding appointments, followed by manual email follow-ups. There's an interest in exploring SMS services to potentially improve communication and efficiency. However, the existing reliance on manual processes and physical records creates bottlenecks and increases the risk of errors, highlighting the need for a more streamlined and automated system.

These questions aimed to reveal pain points and inefficiencies in the existing system, such as manual processes, potential errors, and communication challenges. The answers will guide the design of a new system that streamlines operations, improves efficiency, and enhances communication with pet owners.

System Requirements

The system is designed to meet Northwood Animal Care’s specific needs. The requirements are divided into functional and non-functional categories:

Functional Requirements

- Pet Record Management: Add, update, and view detailed medical records, including vaccination history and medication details.
- Appointment Scheduling: Online booking with automated reminders for pet owners.
- Billing and Invoicing: Automated invoice generation and payment tracking.
- Inventory Management: Real-time inventory tracking with alerts for low stock.
- Customer Communication: Support for reminders via phone, SMS, and email.

Non-Functional Requirements

- Accessibility: Easy-to-use navigation with assistive technology support.
- Reliability: Regular backups and consistent error recovery to ensure data availability.
- Security: Role-based access control and compliance with data privacy standards.

These requirements directly address the client's needs for an integrated system to improve efficiency and accuracy in managing operations.

Initial Use Case Diagram

The initial use case diagram shows the main actors and their interactions with the system:

1. Actors:
    - Pet Owner: Books appointments, views bills, and accesses pet records.
    - Vet: Manages pet medical records.
    - Staff: Handles tasks like billing, scheduling, and inventory.
2. Use Cases:
    - Manage Pet Records: Staff and vets manage pet health information.
    - Book Appointments: Pet owners book appointments, and staff organize schedules.
    - Bill Customer: Staff generate invoices, and pet owners pay bills through the portal.
    - Manage Inventory: Staff track inventory and get alerts for low stock.
    - Manage Appointments: Helps staff organize and track appointments.

The diagram also uses generalization for Manage Medication Record and Manage Vaccination Record under Manage Pet Records. This reduces redundancy and keeps the system organized. Relationships like include and extend were used for shared tasks, such as reminders for appointments and billing.

Initial Class Diagram

The class diagram outlines the system’s main components and how they work together:

1. Key Classes:
    - PetRecords: Stores medical details for pets.
    - Appointments: Manages appointment scheduling and reminders.

    - Bill: Handles invoices and payment tracking.
    - Inventory: Tracks supplies and sends low-stock alerts.
    - CustomerPortal: Lets pet owners book appointments, view records, and pay bills.
    - Employees: Divided into Vet and Staff for their specific roles.
2. Core Interactions:
    - Appointments connect to CustomerPortal for booking and Staff for managing schedules.
    - Bill interacts with CustomerPortal for payments and Staff for creating invoices.
    - PetRecords interact with Staff for updating and CustomerPortal for view-only access.
3. Multiplicity:
    - Inventory to Employees: Many-to-Many, as multiple staff manage supplies.
    - PetOwner to CustomerPortal: One-to-One, ensuring each pet owner has a secure account.

These diagrams were designed to address the client's needs and ensure all parts of the system work together smoothly.

Team Contribution Summary:

Overall, each member of the group was able to evenly contribute to the project. Each deliverable for the project was structured in a way that allowed for the work to be distributed between 3-4 individuals in an even manner.

Additional contributions were as follows:

- Alish & Angie: Coordinated and set up the powerpoint presentation rehearsal
- Carlos: Facilitated contact with Client group for project requirements and interview
- Rohan: Updated and revised use case and class diagram deliverables

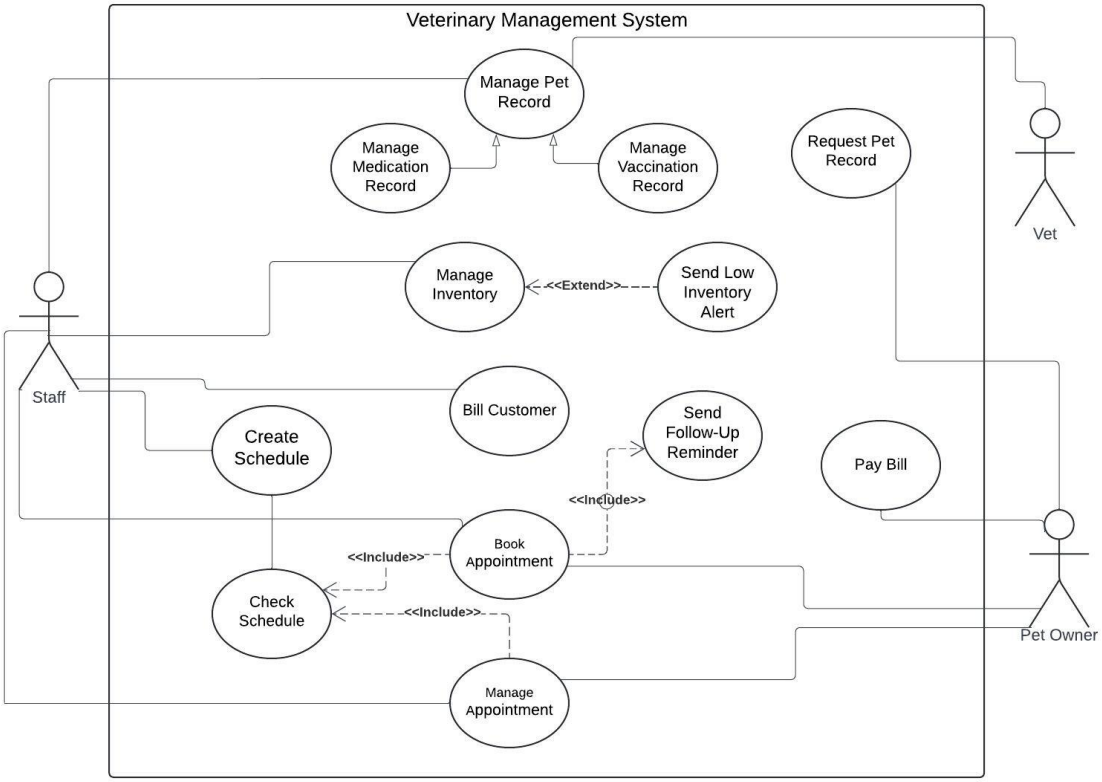
Final Diagram Updates:

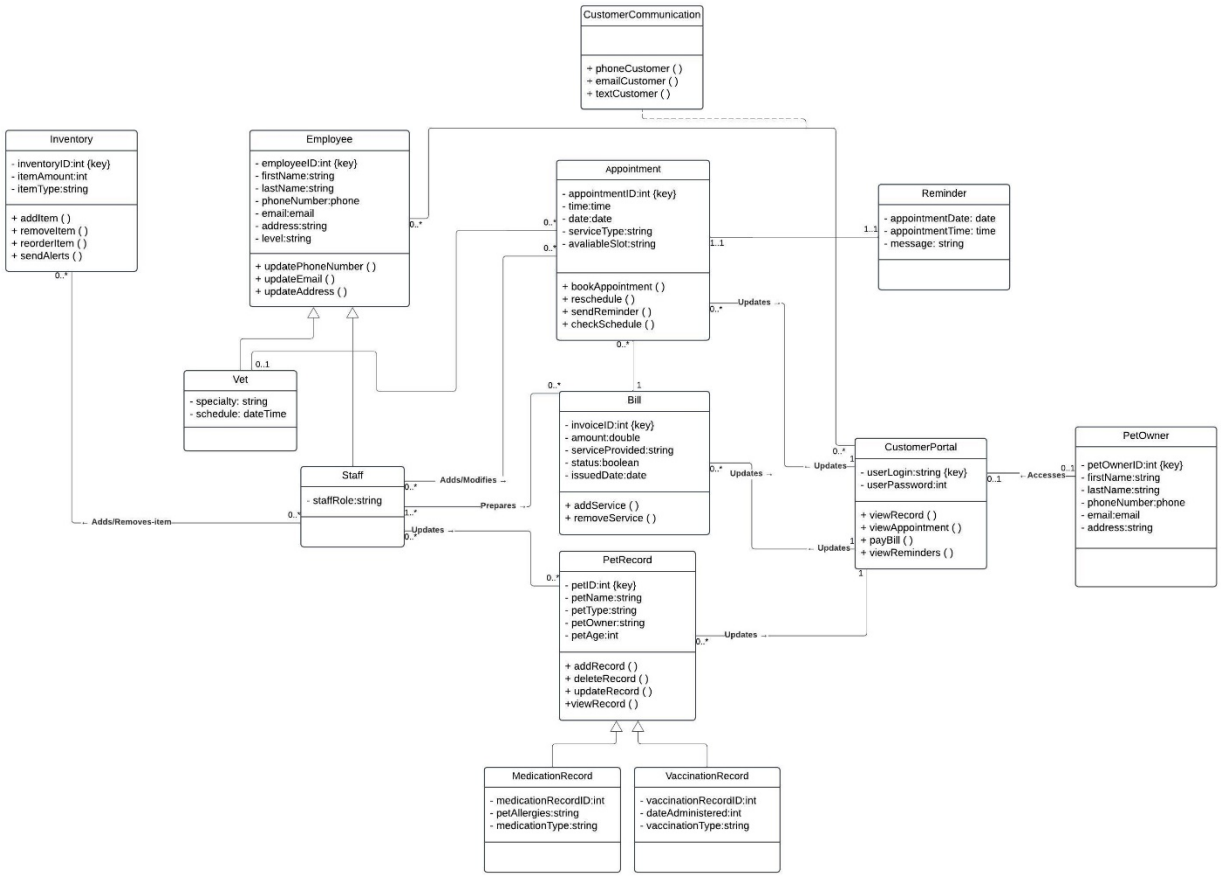
Use Case

- Changed direction of extends relationship for Send Low Inventory Alert use case
- Added a Create Schedule use case

Class

- Changed Inventory association with Employee, to association with Staff
- Reminder association with CustomerPortal changed to association with Appointment
- CustomerCommunication associative class now associated with Employee and CustomerPortal classes


Veterinary Management System
Manage Pet Record
Manage Medication Record
Manage Vaccination Record
Request Pet Record
Vet
Manage Inventory
<<Extend>>
Send Low Inventory Alert
Staff
Bill Customer
Send Follow-Up Reminder
Create Schedule
Pay Bill
<<Include>>
Book Appointment
<<Include>>
Check Schedule
<<Include>>
Pet Owner
Manage Appointment


CustomerCommunication
+ phoneCustomer ( )
+ emailCustomer ( )
+ textCustomer ( )
Inventory
- inventoryID:int {key}
- itemAmount:int
- itemType:string
+ addItem ( )
+ removeItem ( )
+ reorderItem ( )
+ sendAlerts ( )
Employee
- employeeID:int {key}
- firstName:string
- lastName:string
- phoneNumber:phone
- email:email
- address:string
- level:string
+ updatePhoneNumber ( )
+ updateEmail ( )
+ updateAddress ( )
Appointment
- appointmentID:int {key}
- time:time
- date:date
- serviceType:string
- avaliableSlot:string
+ bookAppointment ( )
+ reschedule ( )
+ sendReminder ( )
+ checkSchedule ( )
Reminder
- appointmentDate: date
- appointmentTime: time
- message: string
Vet
- specialty: string
- schedule: dateTime
Staff
- staffRole:string
Bill
- invoiceID:int {key}
- amount:double
- serviceProvided:string
- status:boolean
- issuedDate:date
+ addService ( )
+ removeService ( )
CustomerPortal
- userLogin:string {key}
- userPassword:int
+ viewRecord ( )
+ viewAppointment ( )
+ payBill ( )
+ viewReminders ( )
PetOwner
- petOwnerID:int {key}
- firstName:string
- lastName:string
- phoneNumber:phone
- email:email
- address:string
PetRecord
- petID:int {key}
- petName:string
- petType:string
- petOwner:string
- petAge:int
+ addRecord ( )
+ deleteRecord ( )
+ updateRecord ( )
+viewRecord ( )
MedicationRecord
- medicationRecordID:int
- petAllergies:string
- medicationType:string
VaccinationRecord
- vaccinationRecordID:int
- dateAdministered:int
- vaccinationType:string
Adds/Modifies
Prepares
Updates
Adds/Removes-item
Accesses