Veterinary Clinic Management System

Problem Statement

This veterinary clinic management system is designed to streamline operations and ensure pets receive prompt and effective care. The system allows different user roles—Pet Owners, Veterinarians, and Administrators—to manage appointments, medical records, and pet information, providing a cohesive platform for clinic management.

Pet Owners can log into the system to schedule appointments, view their pets' medical histories, and manage their own information. Veterinarians use the system to log in, access and update medical records after consultations, and manage their schedule of appointments. Administrators handle adding or updating pets and associating them with their respective owners, as well as managing veterinarian and owner details. However, they cannot modify medical records.

Each pet is uniquely named by a pet ID and linked to an owner. Pet information includes the pet's name, species, breed, age, and medical history. Medical histories store detailed records of treatments, vaccinations, surgeries, and prescriptions. Pet owners also have unique owner IDs and handle managing their pets' appointments, with their contact information stored in the system.

Veterinarians, identified by a unique vet ID, provide medical care and maintain appointment schedules. These schedules record details such as the pet being treated, the appointment date and time, the veterinarian assigned, and the reason for the visit (e.g., check-up or vaccination).
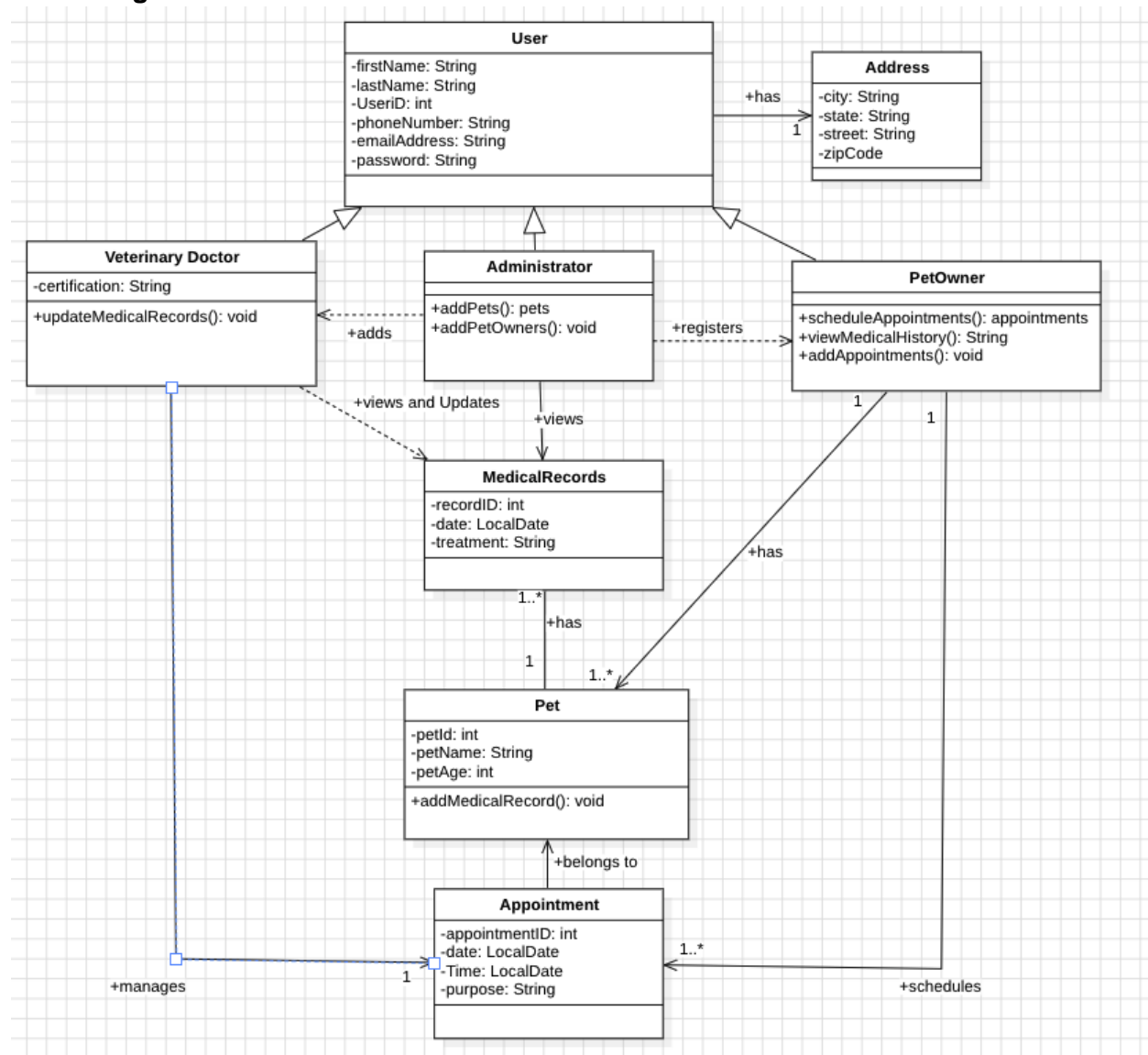
By centralizing clinic operations, the system improves the coordination between pet owners, veterinarians, and administrators, ultimately enhancing the quality of care for pets.
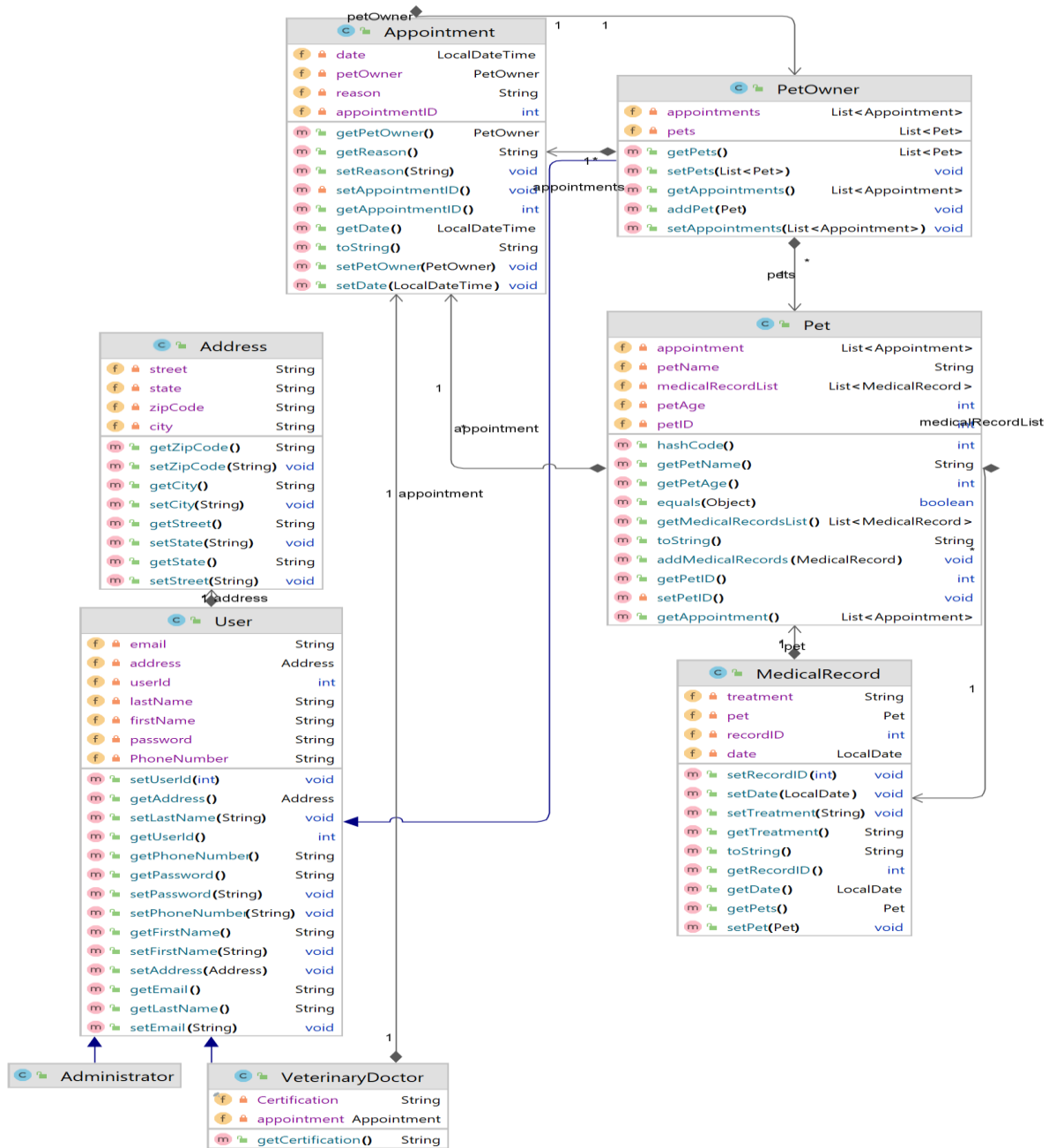
Domain Classes, Attributes, and Relationships
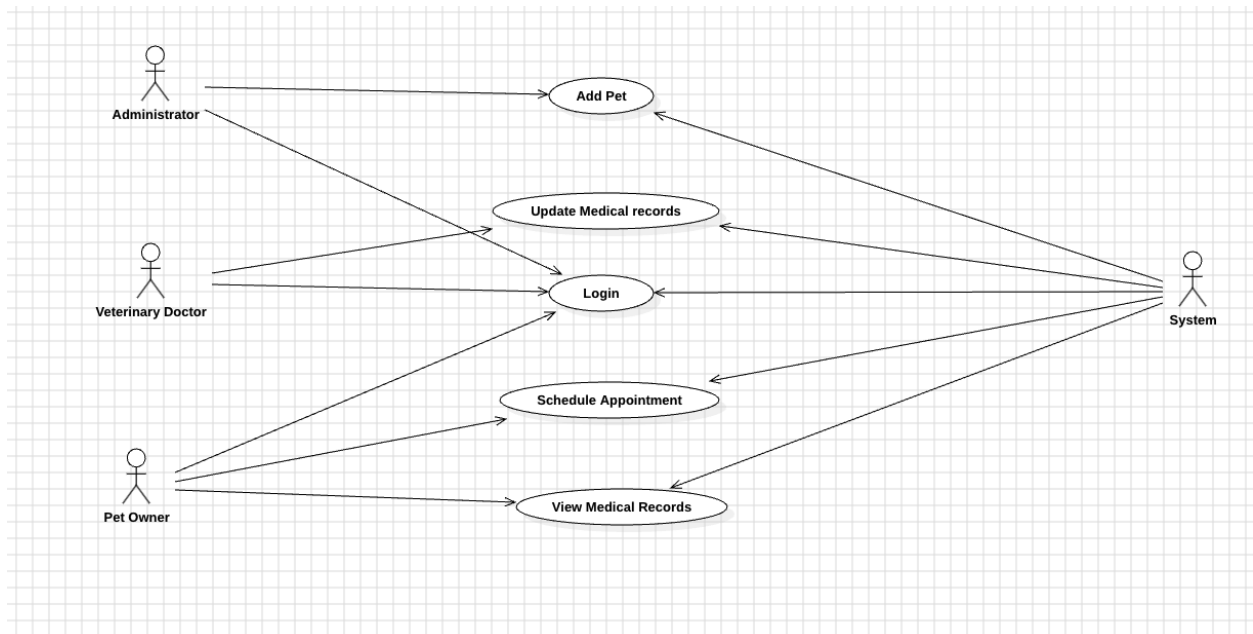
Actors/Domain Classes

- Pet
- Pet Owner
- Veterinary Doctor
- Appointment
- Administrator
- Address
- Medical Records

Class Diagram.

Appointment

petOwner

f	date	LocalDateTime
f	petOwner	PetOwner
f	reason	String
f	appointmentID	int
m	getPetOwner()	PetOwner
m	getReason()	String
m	setReason(String)	void
m	setAppointmentID()	void
m	getAppointmentID()	int
m	getDate()	LocalDateTime
m	toString()	String
m	setPetOwner(PetOwner)	void
m	setDate(LocalDateTime)	void

PetOwner

f	appointments	List<Appointment>
f	pets	List<Pet>
m	getPets()	List<Pet>
m	setPets(List<Pet>)	void
m	getAppointments()	List<Appointment>
m	addPet(Pet)	void
m	setAppointments(List<Appointment>)	void

appointments

pets *

Address

f	street	String
f	state	String
f	zipCode	String
f	city	String
m	getZipCode()	String
m	setZipCode(String)	void
m	getCity()	String
m	setCity(String)	void
m	getStreet()	String
m	setState(String)	void
m	getState()	String
m	setStreet(String)	void

1 address

Pet

f	appointment	List<Appointment>
f	petName	String
f	medicalRecordList	List<MedicalRecord >
f	petAge	int
f	petID	int
m	hashCode()	int
m	getPetName()	String
m	getPetAge()	int
m	equals(Object)	boolean
m	getMedicalRecordsList()	List<MedicalRecord >
m	toString()	String
m	addMedicalRecords(MedicalRecord)	void
m	getPetID()	int
m	setPetID()	void
m	getAppointment()	List<Appointment>

medicalRecordList

appointment

1 appointment

1 pet

User

f	email	String
f	address	Address
f	userId	int
f	lastName	String
f	firstName	String
f	password	String
f	PhoneNumber	String
m	setUserId(int)	void
m	getAddress()	Address
m	setLastName(String)	void
m	getUserId()	int
m	getPhoneNumber()	String
m	getPassword()	String
m	setPassword(String)	void
m	setPhoneNumber(String)	void
m	getFirstName()	String
m	setFirstName(String)	void
m	setAddress(Address)	void
m	getEmail()	String
m	getLastName()	String
m	setEmail(String)	void

MedicalRecord

f	treatment	String
f	pet	Pet
f	recordID	int
f	date	LocalDate
m	setRecordID(int)	void
m	setDate(LocalDate)	void
m	setTreatment(String)	void
m	getTreatment()	String
m	toString()	String
m	getRecordID()	int
m	getDate()	LocalDate
m	getPets()	Pet
m	setPet(Pet)	void

Administrator

VeterinaryDoctor

f	Certification	String
f	appointment	Appointment
m	getCertification()	String

\

Use Case Diagram

Use Cases

Use Case 1 - Login to the Veterinary Clinic Management System
Pet Owners, Veterinary Doctors, and Administrators log into the system by providing valid credentials. Upon successful authentication, they are redirected to their role-specific dashboard. The user must be registered with valid login credentials to gain access. If credentials are invalid, the system displays an error message.

Use Case 2 - Schedule a Veterinary Appointment
A pet owner logs in to schedule an appointment for their pet by selecting a time slot and specifying the purpose (e.g., check-up or vaccination). Once confirmed, the appointment is added successfully. The owner must be registered and logged in to book an appointment.

Use Case 3 - View Pet's Medical History
A pet owner logs in and views their pet's medical history like treatments. The pet must have an existing medical history in the system, and the owner must be logged in to access this information.

Use Case 4 - Add a New Pet to the System
An administrator logs into the system and adds a new pet by entering the pet's details, such as name, age, and the pet owner's information. Once added, the pet is linked to the correct pet owner in the system.

Use Case 5 - Update Pet Medical Record
After a consultation, the veterinarian updates the pet's medical record with treatment details. The veterinarian must be logged in to update the records. Medical history is updated with this new information.

1. **Add a new Pet to the system.**

User Action	em Response
Administrator logins into the system	System checks validity of the user login credentials and presents options for the user to choose
The Administrator selects "Add Pet" Option	The systems displays a screen to add Pet Owner ID
The Administrator inputs the PetOwner ID.	The system validates if the PetOwner ID entered exists and presents another window to add Pet Name.
The Administrator adds the pet 's name.	The system Accepts the name and presents another screen to add Pet age.
The Administrator adds the pet's age.	The system accepts the age detail and adds Pet in the system successfully.

2. User Login

User Action	em Response
The User chooses the role they are assigned too.	System processes request and presents a window for user to input their Email Address.
The User enters their Email Address.	The system validates if the email address entered is of correct format and presents another window for the user to input a password.
The User inputs a password.	The system validates if the Email Address and the password provided by the user exist in the system and if they match, the system successfully logs the user in the system and displays the different rights they have in the system.

3. Schedule Appointment

User Action	System Response
The user chooses Pet Owner Login	System processes request and presents another window to Enter Email Address.
The User enters their Email Address	The system validates if the email address entered is of correct format and presents another window for the user to input a password.
The User inputs a password	The system validates if the Email Address and the password provided by the user exist in the system and if they match, the system successfully logs the user in the system and displays the different rights they have in the system as Pet Owner Role.
The user selects "Schedule an Appointment" option	The system processes request and presents another window to enter the Appointment date.
The user enters the Date	The system validates if the date from the user is of the right format and presents another window to enter appointment time.
The user enters the time for the appointment	The system validates if the time from the user is of the right format and schedules an appointment for the pet owner.

4. View Pet Medical Records

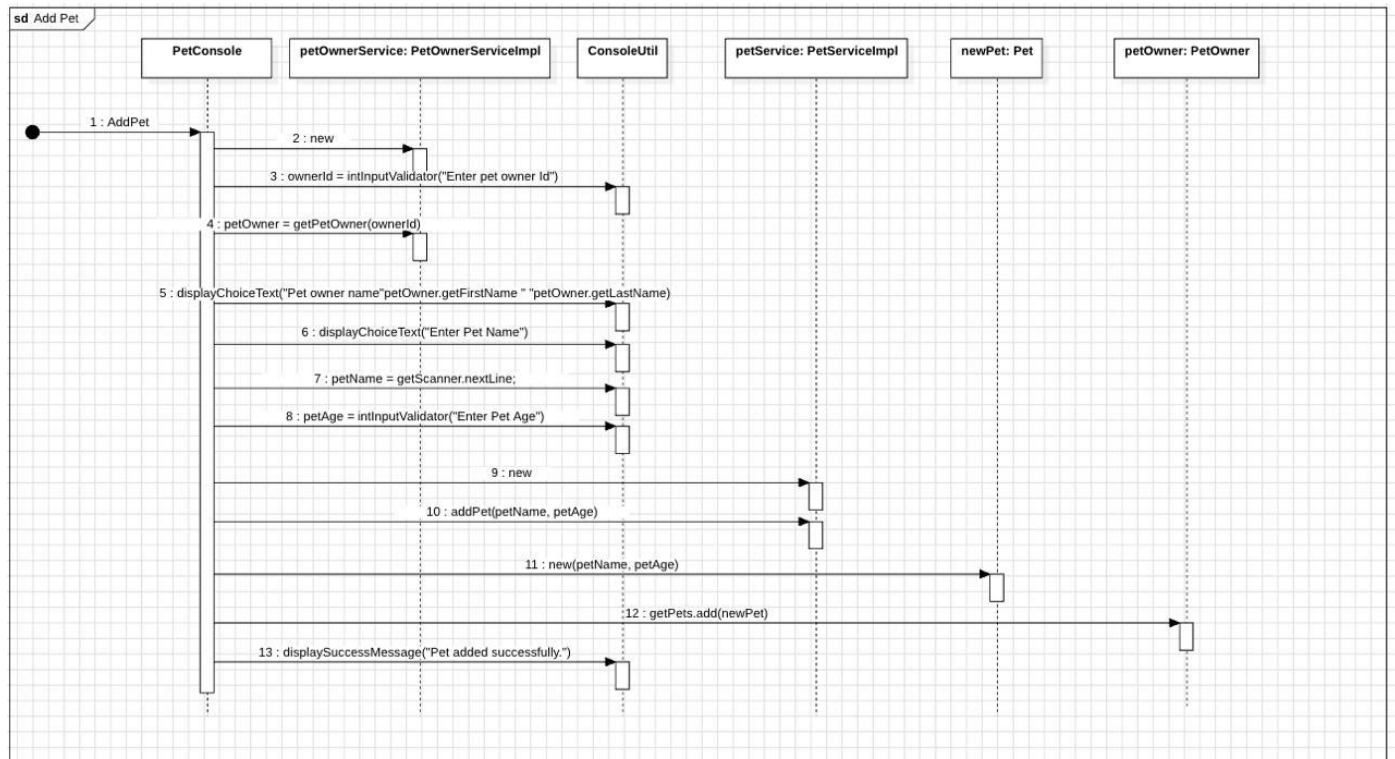| User Action	System Response
The user chooses Pet Owner Login.	System processes request and presents another window to Enter Email Address.
The User enters their Email Address.	The system validates if the email address entered is of correct format and presents another window for the user to input a password.
The User inputs a password.	The system validates if the Email Address and the password provided by the user exist in the system and if they match, the system successfully logs the user in the system and displays the different rights they have in the system as Pet Owner Role.
The user selects the "View Pets Medical History" option.	The system processes request and presents a list of pets belonging to the logged in user who is the pet Owner.
The user selects the pet whose history they want to see from the list.	The system processes request and displays the Medical history of the selected pet.

5. Update Pet Medical History

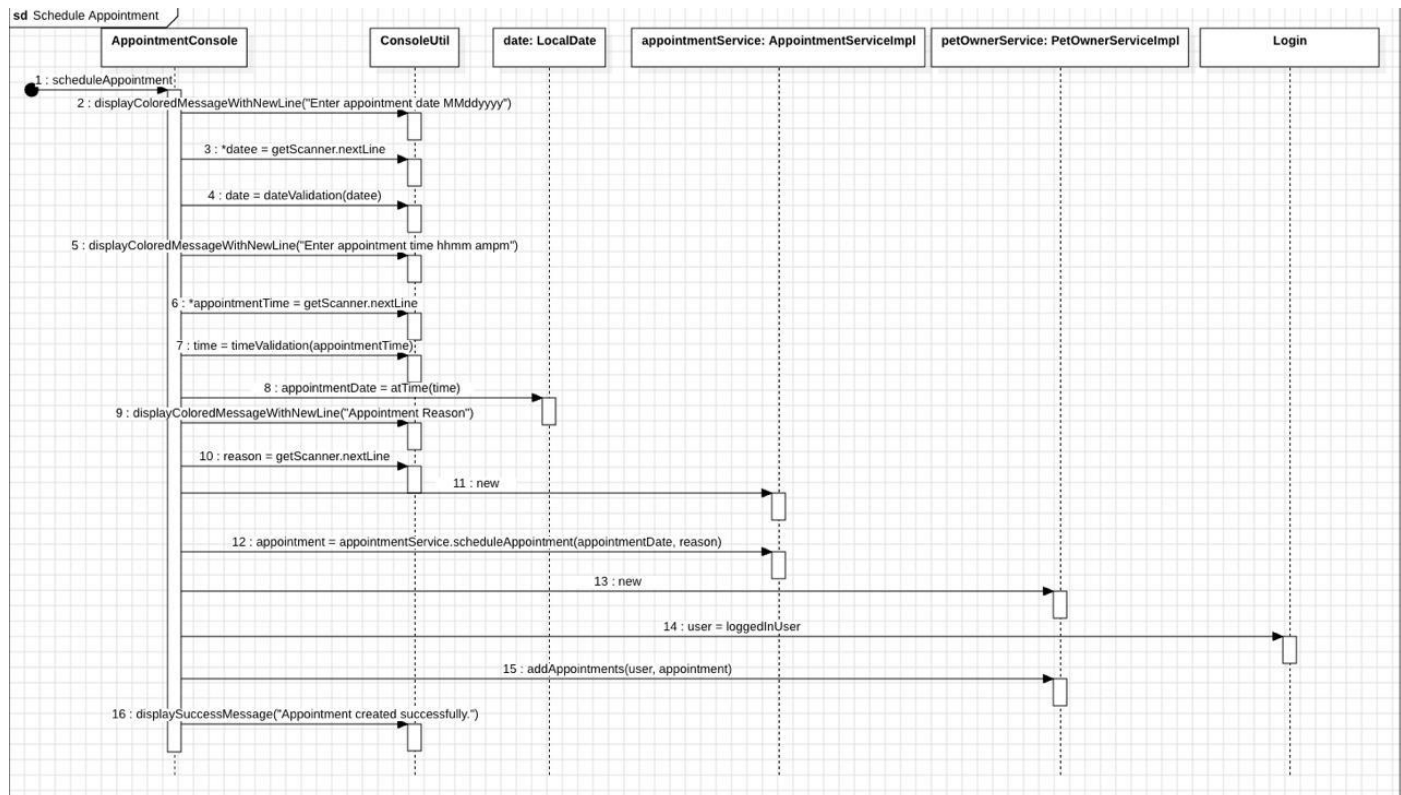| User Action	System Response
User chooses the Veterinary Doctor Login.	System processes request and presents a window for users to input their Email Address.
The User enters their Email Address.	The system validates if the email address entered is of correct format and presents another window for the user to input a password.
The User inputs a password.	The system validates if the Email Address and the password provided by the user exist in the system and if they match, the system successfully logs the user in the system and displays the different rights they have in the system .
The user chooses "Update Medical Records" .	The system processes request and presents another window displaying a list of pets in the system.
The user selects the Pet whose Medical history is to be updated.	The system processes the request and displays the list of medical records belonging to the pet.
The user selects the record ID of the medical record to be updated.	The system processes the request and prompts the user to enter the new treatment.

The user inputs a new treatment .	The system processes the update of the new treatment and the record is updated successfully.

Sequence Diagrams

Add Pet

Schedule Appointment

View Medical Records

sd View Medical Records Use Case

MedicalRecordsConsole **ConsoleUtil** **Login** **owner: PetOwner** **Pet**

1 : viewMedicalRecords

2 : displayChoiceText("Pets")

3 : owner = loggedInUser

4 : getPets

5 : *getPetName

6 : input = intInputValidator("Choose Pet")

7 : getPets

8 : get(input - 1)

9 : md = getMedicalRecordsList

10 : *displaySuccessMessage(md.toString)