Animal-Care Veterinary Clinic Management System

Part 3 - Technical documentation

Lasalle College - Fall 2025

Web Server Application

Professor: Houria Hamel

Team: Negar Pirasteh - Betty Dang - Hope Jeanine Ukundimana - Ngoc Yen Nhi Pham

Contents

1. Project overview

The Animal-Care Veterinary Clinic system is a centralized ASP.NET Core MVC web application that replaces manual paper-based processes used in the clinic.

The application allows the clinic to:

- Manage **owners** and their **animals**
- Manage **veterinarians** and their **work schedules**
- Create and manage **appointments**
- Record **visit histories** for completed appointments
- Authenticate users and restrict access based on **roles** (Admin, Secretary, Veterinarian)
- Generate **monthly activity reports** for administrators

The goal is to provide a secure, modular, and user-friendly system that follows the MVC pattern and the requirements described in the final exam document.

---

2. Technologies used

- **Framework:** ASP.NET Core 8.0 (MVC)
- **Language:** C#
- **Database:** Microsoft SQL Server 2022
- **ORM:** Entity Framework Core 8 (Database-First, Scaffolded Models)
- **Authentication:** Cookie Authentication (Forms login)
- **Front-end:** HTML5, CSS3, Bootstrap, jQuery, ASP.NET Core Tag Helpers
- **Tools:** Visual Studio 2022, SQL Server Management Studio

---

3. Solution structure

Solution name: AnimalCareClinicSolution
Project name: AnimalCareClinic

Main directories in the project:

- **Controllers**
    - AccountController.cs

  - HomeController.cs
  - AnimalsController.cs
  - OwnersController.cs
  - VeterinariansController.cs
  - SchedulesController.cs
  - AppointmentsController.cs
  - VisitHistoriesController.cs
  - ReportsController.cs
- **Models**
  - Owner.cs
  - Animal.cs
  - Veterinarian.cs
  - Schedule.cs
  - Appointment.cs
  - VisitHistory.cs
  - UserAccount.cs
  - AnimalCareClinicContext.cs
- **ViewModels**
  - MonthlyReportVM.cs
  - VetWorkloadItem.cs
- **Views**
  - Shared (layout, validation scripts, error page, AccessDenied)
  - Home (Index, Privacy)
  - Account (Login)
  - Animals (CRUD views)
  - Owners (CRUD views)
  - Veterinarians (CRUD views)

  - Schedules (CRUD views)
  - Appointments (CRUD views)
  - VisitHistories (CRUD views)
  - Reports (Monthly report)
- **Database Schema**
  - Animal-ClinicDB_T1.sql (SQL script for database)
- **Documentation**
  - Technical Documentation.docx
  - Animal-Clinic_Test Plan.docx / Animal-Clinic_Test Plan.pdf

---

4. Database design

Tables and Fields (simplified):

1. **Owner**
   - OwnerId (PK, int)
   - FirstName (nvarchar)
   - LastName (nvarchar)
   - Address (nvarchar)
   - PhoneNumber (nvarchar)
   - Email (nvarchar)
2. **Animal**
   - AnimalId (PK, int)
   - OwnerId (FK → Owner)
   - Name (nvarchar)
   - Species (nvarchar)
   - Age (int)
   - Gender (nvarchar)
   - MedicalHistory (nvarchar, nullable)

3. **Veterinarian**
    - VeterinarianId (PK, int)
    - FirstName (nvarchar)
    - LastName (nvarchar)
    - Specialty (nvarchar)
    - PhoneNumber (nvarchar)
    - Email (nvarchar)
4. **Schedule**
    - ScheduleId (PK, int)
    - VeterinarianId (FK → Veterinarian)
    - Date (date)
    - TimeSlot (nvarchar, e.g., “09:00–10:00”)
    - Status (nvarchar, e.g., Available / Booked / Cancelled)
5. **Appointment**
    - AppointmentId (PK, int)
    - ScheduleId (FK → Schedule)
    - AnimalId (FK → Animal)
    - AppointmentDate (date)
    - AppointmentTime (time)
    - Reason (nvarchar, nullable)
    - Status (nvarchar, e.g., Booked / Cancelled / Completed)
6. **VisitHistory**
    - VisitId (PK, int)
    - AppointmentId (FK → Appointment)
    - AnimalId (FK → Animal)
    - VeterinarianId (FK → Veterinarian)
    - VisitDate (date)

- Diagnosis (nvarchar, nullable)
- Treatment (nvarchar, nullable)
- Prescription (nvarchar, nullable)

7. **UserAccount**
    - UserAccountId (PK, int)
    - Username (nvarchar)
    - Password (nvarchar – plain text only for course purposes)
    - Role (nvarchar: Admin / Secretary / Veterinarian)

Main relationships (verbal ERD):

- An **Owner** has many **Animals**.
- A **Veterinarian** has many **Schedules** and many **VisitHistories**.
- A **Schedule** has many **Appointments**.
- An **Appointment** belongs to one **Animal** and one **Schedule**, and can have one associated **VisitHistory**.

---

5. Main classes (models)

For each model, a short description and validation rules.

5.1 Owner

Represents a pet owner.

Main validation (Data Annotations):

- FirstName, LastName, Address, PhoneNumber, Email: required, with StringLength limits and EmailAddress for email.

5.2 Animal

Represents a patient (pet) belonging to an owner.

Validation:

- OwnerId: required, must reference an existing Owner.
- Name, Species, Gender: required, string length limited.
- Age: required, [Range(0, 100)].

- MedicalHistory: optional.

5.3 Veterinarian

Represents a clinic doctor.

Validation:

- FirstName, LastName, Specialty, PhoneNumber, Email: required.
- Email has [EmailAddress].
- Names and specialty have [StringLength].

5.4 Schedule

Represents availability of a veterinarian.

Validation:

- VeterinarianId: required.
- Date: required.
- TimeSlot: required, string pattern (e.g., “09:00 – 10:00”).
- Status: required (Available, Booked, Cancelled).

5.5 Appointment

Represents a booked meeting for an animal.

Validation:

- ScheduleId and AnimalId: required, must reference existing records.
- AppointmentDate: required.
- AppointmentTime: required.
- Reason: optional but length-limited.
- Status: required (Booked, Cancelled, Completed).

5.6 VisitHistory

Represents completed visit information.

Validation:

- AppointmentId, AnimalId, VeterinarianId, VisitDate: required.
- Diagnosis, Treatment, Prescription: optional text fields with StringLength.

5.7 UserAccount

Represents login credentials and permission.

Validation:

- Username: required, unique (enforced logically).
- Password: required, minimum length.
- Role: required (Admin / Secretary / Veterinarian).

---

6. Security and validation

- Role-based [Authorize] attributes on controllers and actions
- [ValidateAntiForgeryToken] on all POST actions
- Server-side validation via DataAnnotations (Required, Range, StringLength, EmailAddress)
- Client-side validation via _ValidationScriptsPartial in views
- Only authenticated users see navigation links for data management and reports

---

7. Client-side logic

- Bootstrap CSS used for responsive layout and navigation bar
- Validation messages displayed directly under invalid form fields
- <input type="date"> and <input type="time"> used for Schedule, Appointment, VisitDate/VisitTime
- Conditional navbar items: different menus for Admin, Secretary, Veterinarian, and anonymous users

---

8. Monthly reports feature

The **ReportsController.Monthly** action aggregates data by month:

- Total number of appointments
- Number of booked, cancelled, completed appointments
- Workload per veterinarian (number of visits in VisitHistory for that month)

The view allows the administrator to choose **Year** and **Month**, and displays:

- Summary cards (totals)
- A table listing each veterinarian and the number of visits handled

---

9. How to run the application

1. Restore the **Animal-ClinicDB_T1.sql** script in SQL Server 2022.
2. Update the connection string in appsettings.json to point to your SQL Server instance.
3. Open AnimalCareClinic.sln in Visual Studio 2022.
4. Restore NuGet packages.
5. Run the project (IIS Express or Kestrel).
6. Log in using one of the predefined accounts in the UserAccounts table (e.g., admin1 / admin123).

The application is now ready for demonstration.