Министерство науки и высшего образования Российской Федерации

федеральное государственное автономное   
образовательное учреждение высшего образования   
«Самарский национальный исследовательский университет   
имени академика С.П. Королева»

Институт информатики и кибернетики

Кафедра технической кибернетики

**Краткая документация по проекту   
«FluffyTailClinic»**

Выполнили:

Рыбалко Ф. А. 6305-010302D

Мурзина А. Е. 6305-010302D

Самара, 2025

**Введение**

FluffyTailClinic — это система управления ветеринарной клиникой, предназначенная для автоматизации работы с клиентами, записями на прием, медицинскими записями животных и ветеринарными специалистами. Данное программное обеспечение разработано для облегчения взаимодействия между владельцами животных, администрацией клиники и ветеринарами, а также для улучшения качества обслуживания.

**Основные функции системы**:

* Управление данными о животных и их владельцах.
* Запись на прием к ветеринарам.
* Ведение медицинской истории животных.
* Управление специалистами клиники.
* Обеспечение безопасности и управления доступом через систему аутентификации.

**1. Проектирование архитектуры**

**Схема взаимодействия компонентов:**

* Клиент (Frontend): Веб-приложение, через которое пользователи (владельцы животных, администраторы, ветеринары) взаимодействуют с системой.
* Сервер (Backend): Обрабатывает запросы от клиента, выполняет бизнес-логику и взаимодействует с базой данных.
* База данных (DB): Хранит информацию о животных, владельцах, записях на прием, медицинских записях и т.д.

Схема взаимодействия:

Клиент (Frontend) -> Сервер (Backend) -> База данных (DB)

**2. Логическая схема базы данных**

Сущности:

**1. Animal (Животное):**

* `id` (Primary Key)
* `name` (Имя животного)
* `type` (Вид животного)
* `owner\_id` (Foreign Key к таблице User)

**2. User (Пользователь):**

* `id` (Primary Key)
* `name` (Имя владельца)
* `phone` (Телефон)
* `email` (Электронная почта)
* `role ID` (Foreign Key)
* `description` (Описание)

**3. Appointment (Запись на прием):**

* `id` (Primary Key)
* `animal\_id` (Foreign Key к таблице Animal)
* `vet\_id` (Foreign Key к таблице User)
* `date` (Дата и время приема)
* `description` (Описание проблемы)

**5. Diagnosis (Диагноз):**

* ‘id’ (Primary Key)
* `name` (Имя)
* ‘description’ (Описание)

**6. Role (Роль) :**

* ‘id’ (Primary Key)
* ‘description’ (Описание)

**6. DiagnosisAppointment (Диагноз в записи) :**

* ‘appointment\_id’ (Foreign Key к таблице Appointment)
* ‘diagnosis\_id’ (Foreign Key к таблице Diagnosis)

Связь:

* Many-to-Many: Связь между ‘Diagnosis’ и ‘Appointment’ с использованием третьей таблицы для связи.

**3. Структура API**

**1. Животные (Animals)**

**Получить список всех животных**

- Метод: `GET`

- URL: `/api/animals`

- Ответ:

  [

    {

      "id": 1,

      "name": "Барсик",

      "type": "Кошка",

      "owner\_id": 1

    },

    {

      "id": 2,

      "name": "Шарик",

      "type": "Собака",

      "owner\_id": 2

    }

  ]

**Получить информацию о конкретном животном**

- Метод: `GET`

- URL: `/api/animals/{id}`

- Ответ:

  {

    "id": 1,

    "name": "Барсик",

    "type": "Кошка",

    "owner\_id": 1

  }

**Добавить новое животное**

- Метод: `POST`

- URL: `/api/animals`

- Запрос:

  {

    "name": "Барсик",

    "type": "Кошка",

    "owner\_id": 1

  }

- Ответ:

  {

    "id": 1,

    "name": "Барсик",

    "type": "Кошка",

    "owner\_id": 1

  }

**Обновить информацию о животном**

- Метод: `PUT`

- URL: `/api/animals/{id}`

- Запрос:

  {

    "name": "Барсик",

    "type": "Кошка",

    "owner\_id": 1

  }

- Ответ:

  {

    "id": 1,

    "name": "Барсик",

    "type": "Кошка",

    "owner\_id": 1

  }

**Удалить животное**

- Метод: `DELETE`

- URL: `/api/animals/{id}`

- Ответ: `204 No Content`

**2. Владельцы (Owners)**

**Получить список всех владельцев**

- Метод: `GET`

- URL: `/api/owners`

- Ответ:

  [

    {

      "id": 1,

      "name": "Иван Иванов",

      "phone": "+7 999 123 45 67"

    },

    {

      "id": 2,

      "name": "Мария Петрова",

      "phone": "+7 999 765 43 21"

    }

  ]

**Получить информацию о конкретном владельце**

- Метод: `GET`

- URL: `/api/owners/{id}`

- Ответ:

  {

    "id": 1,

    "name": "Иван Иванов",

    "phone": "+7 999 123 45 67"

  }

**Добавить нового владельца**

- Метод: `POST`

- URL: `/api/owners`

- Запрос:

  {

    "name": "Иван Иванов",

    "phone": "+7 999 123 45 67"

  }

- Ответ:

  {

    "id": 1,

    "name": "Иван Иванов",

    "phone": "+7 999 123 45 67"

  }

**Обновить информацию о владельце**

- Метод: `PUT`

- URL: `/api/owners/{id}`

- Запрос:

  {

    "name": "Иван Иванов",

    "phone": "+7 999 123 45 67"

  }

- Ответ:

  {

    "id": 1,

    "name": "Иван Иванов",

    "phone": "+7 999 123 45 67"

  }

**Удалить владельца**

- Метод: `DELETE`

- URL: `/api/owners/{id}`

- Ответ: `204 No Content`

**3. Записи на прием (Appointments)**

**Получить список всех записей**

- Метод: `GET`

- URL: `/api/appointments`

- Ответ:

  [

    {

      "id": 1,

      "animal\_id": 1,

      "vet\_id": 1,

      "date": "2025-10-15T10:00:00Z"

    },

    {

      "id": 2,

      "animal\_id": 2,

      "vet\_id": 2,

      "date": "2025-10-16T11:00:00Z"

    }

  ]

**Получить информацию о конкретной записи**

- Метод: `GET`

- URL: `/api/appointments/{id}`

- Ответ:

  {

    "id": 1,

    "animal\_id": 1,

    "vet\_id": 1,

    "date": "2025-10-15T10:00:00Z"

  }

**Добавить новую запись**

- Метод: `POST`

- URL: `/api/appointments`

- Запрос:

  {

    "animal\_id": 1,

    "vet\_id": 1,

    "date": "2025-10-15T10:00:00Z"

  }

- Ответ:

  {

    "id": 1,

    "animal\_id": 1,

    "vet\_id": 1,

    "date": "2025-10-15T10:00:00Z"

  }

**Обновить информацию о записи**

- Метод: `PUT`

- URL: `/api/appointments/{id}`

- Запрос:

  {

    "animal\_id": 1,

    "vet\_id": 1,

    "date": "2025-10-15T10:00:00Z"

  }

- Ответ:

  {

    "id": 1,

    "animal\_id": 1,

    "vet\_id": 1,

    "date": "2025-10-15T10:00:00Z"

  }

**Удалить запись**

- Метод: `DELETE`

- URL: `/api/appointments/{id}`

- Ответ: `204 No Content`

**4. Ветеринары (Vets)**

**Получить список всех ветеринаров**

- Метод:`GET`

- URL: `/api/vets`

- Ответ:

  [

    {

      "id": 1,

      "name": "Алексей Смирнов",

      "specialization": "Хирургия"

    },

    {

      "id": 2,

      "name": "Ольга Иванова",

      "specialization": "Терапия"

    }

  ]

**Получить информацию о конкретном ветеринаре**

- Метод: `GET`

- URL: `/api/vets/{id}`

- Ответ:

  {

    "id": 1,

    "name": "Алексей Смирнов",

    "specialization": "Хирургия"

  }

**Добавить нового ветеринара**

- Метод: `POST`

- URL: `/api/vets`

- Запрос:

  {

    "name": "Алексей Смирнов",

    "specialization": "Хирургия"

  }

- Ответ:

  {

    "id": 1,

    "name": "Алексей Смирнов",

    "specialization": "Хирургия"

  }

**Обновить информацию о ветеринаре**

- Метод: `PUT`

- URL: `/api/vets/{id}`

- Запрос:

  {

    "name": "Алексей Смирнов",

    "specialization": "Хирургия"

  }

- Ответ:

  {

    "id": 1,

    "name": "Алексей Смирнов",

    "specialization": "Хирургия"

  }

**Удалить ветеринара**

- Метод: `DELETE`

- URL: `/api/vets/{id}`

- Ответ: `204 No Content`

**5. Диагнозы (Diagnoses)**

**Получить список всех диагнозов**

- Метод: `GET`

- URL: `/api/diagnoses`

- Ответ:

  [

    {

      "id": 1,

      "animal\_id": 1,

      "diagnosis": "Гастрит",

      "date": "2025-10-15T10:00:00Z"

    },

    {

      "id": 2,

      "animal\_id": 2,

      "diagnosis": "Перелом",

      "date": "2025-10-16T11:00:00Z"

    }

  ]

**Получить информацию о конкретном диагнозе**

- Метод: `GET`

- URL: `/api/diagnoses/{id}`

- Ответ:

  {

    "id": 1,

    "animal\_id": 1,

    "diagnosis": "Гастрит",

    "date": "2025-10-15T10:00:00Z"

  }

**Добавить новый диагноз**

- Метод: `POST`

- URL: `/api/diagnoses`

- Запрос:

  {

    "animal\_id": 1,

    "diagnosis": "Гастрит",

    "date": "2025-10-15T10:00:00Z"

  }

- Ответ:

  {

    "id": 1,

    "animal\_id": 1,

    "diagnosis": "Гастрит",

    "date": "2025-10-15T10:00:00Z"

  }

**Обновить информацию о диагнозе**

- Метод: `PUT`

- URL: `/api/diagnoses/{id}`

- Запрос:

  {

    "animal\_id": 1,

    "diagnosis": "Гастрит",

    "date": "2025-10-15T10:00:00Z"

  }

- Ответ:

  {

    "id": 1,

    "animal\_id": 1,

    "diagnosis": "Гастрит",

    "date": "2025-10-15T10:00:00Z"

  }

**Удалить диагноз**

- Метод: `DELETE`

- URL: `/api/diagnoses/{id}`

- Ответ: `204 No Content`

**Общие параметры:**

- Формат запросов и ответов: JSON.

- Коды ответов:

  - `200 OK` — успешный запрос.

  - `201 Created` — успешное создание ресурса.

  - `204 No Content` — успешное удаление или обновление без возврата данных.

  - `400 Bad Request` — неверный запрос.

  - `404 Not Found` — ресурс не найден.

  - `500 Internal Server Error` — ошибка на сервере.

**4. Стек технологий**

* Frontend: Thymeleaf (для рендеринга HTML на стороне сервера) + HTML/CSS (для статики и интерактивности).
* Backend: Spring Boot (Java).
* База данных: PostgreSQL.
* API: RESTful API. Дополнительные технологии:
* ORM: Hibernate (для работы с базой данных в Spring Boot).
* Аутентификация: Spring Security (для управления доступом и аутентификацией).
* Шаблонизатор: Thymeleaf (для динамической генерации HTML на стороне сервера).
* Документация API: OpenAPI.
* Документация в GitHub: Markdown API

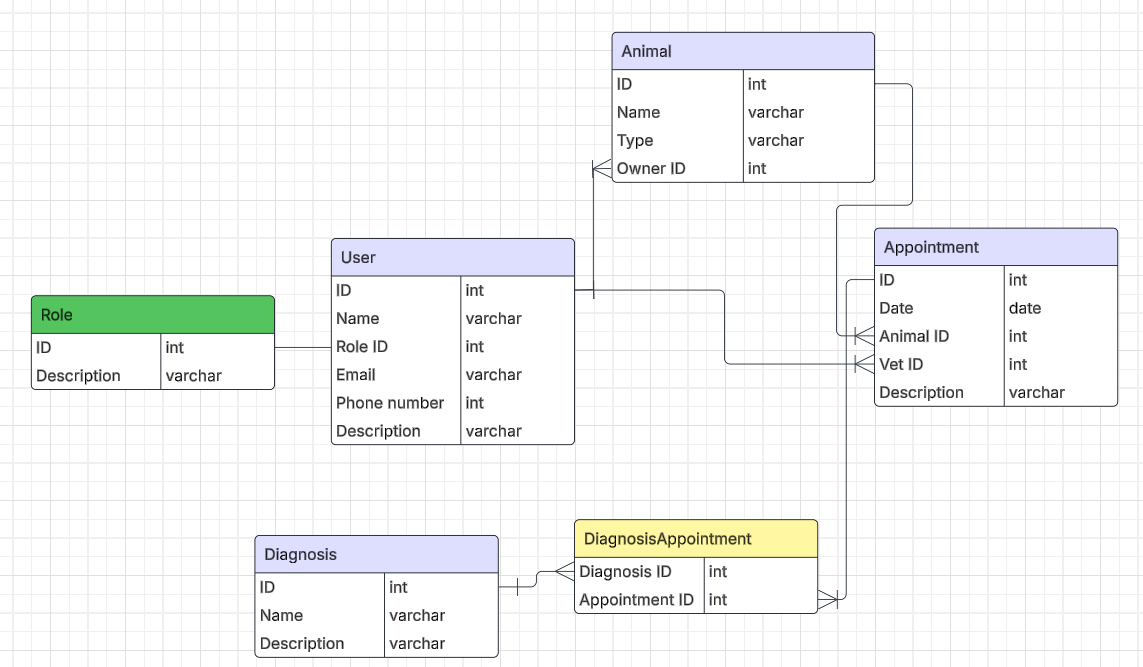


Рисунок 1 – логическая схема базы данных