Информационный веб-сайт

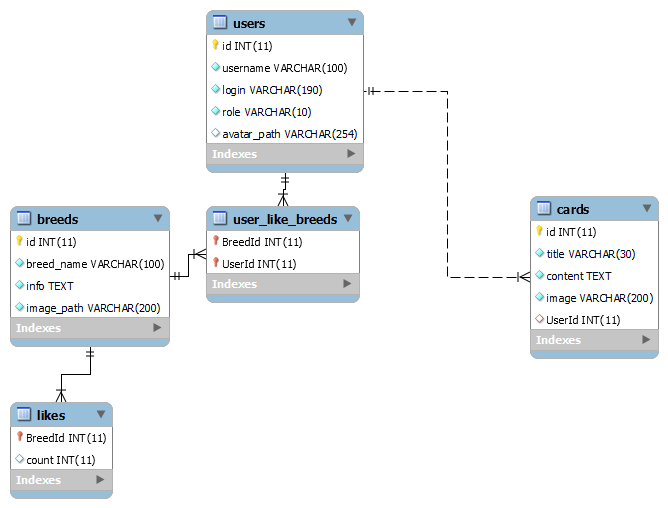
**Описание**

Web-сайт, в котором будет собрана и в удобном формате представлена информация о различных породах собак. Сайт будет состоять из 2 главных страниц:   
- Главная страница с различными породами. Породы будут представлены в виде карточек с описанием породы.  
- Вторая страница с пользовательскими карточками. Пользовательские карточки будут включать в себя информацию о конкретном питомце конкретного пользователя.  
  
**Функциональность**  
- возможность авторизоваться на сайте (для создания/просмотра пользовательских карточек)  
- возможность поиска по названию породы  
- возможность оценить понравившуюся породу  
- возможность отсортировать породы по количеству лайков  
- каждый пользователь имеет свой профиль  
- возможность зайти на профиль пользователя и посмотреть его карточки **Пользовательские роли  
Конечные пользователи** – те, кто будут непосредственно пользоваться сайтом, авторизовываться, добавлять свои карточки (сюда относятся как авторизованные пользователи, так и не авторизованные)  
**Неавторизованные пользователи** (VISITOR) – пользователи, которые имеют доступ к сайту, но могут просматривать ТОЛЬКО основную публичную страницу (public). Доступа к сообществу/к оцениванию/просмотру чужих профилей у неавторизованных пользователей нет.   
**Авторизованные пользователи** (USER) – пользователи, которые успешно авторизовались на сайте и имеют возможность просматривать/создавать свои карточки.  
**Модератор пользовательского контента (MODERATOR) –** тот, кто будет следить за пользовательским контентом. В его обязанности будет входить проверка пользовательских карточек на соблюдение определенного формата и определенных требований. Модератор по своему усмотрению может удалять пользовательские карточки (те, которые противоречат тематике сайта или являются источниками неположительной информации).  
**Контент-менеджер (MANAGER)** – тот, кто будет наполнять систему актуальной информацией. (То есть будет вносить определенные данные в БД).

**Архитектор системы/администратор данных/менеджер проекта** – проектировщик системы. Выдвигает основные требования к системе и следит за их соблюдением.   
**Разработчики системы** – прикладные программисты, использующие технологии для разработки описываемого продукта. (Сюда входят программисты всех уровней, то есть администратор баз данных тоже входит в эту группу)  
**Тестировщик** – специалист, который будет проверять корректность работы системы и соблюдение выдвигаемых требований посредством тестирования.

**Данные**  
Для хранения информации будет задействована база данных. Она будет состоять из 5 таблиц:  
- **Breeds**:   
**id** – INT,  
**breed\_name** – VARCHAR(100),   
**info** – TEXT,   
**image\_path** – VARCHAR(200))  
Таблица содержит информацию, которая показывается на карточках на главной странице (породы)

- **Likes**:   
**BreedId** – INT,   
**count** – INT  
Таблица содержит информацию о количестве лайков у каждой породы (подробне см. транзакции)

- **Users**:   
**id** – INT,   
**username** – VARCHAR(100),  
**login** – VARCHAR(190),  
**role** – VARCHAR(10),  
**avatar\_path** – VARCHAR(254)  
Таблица содержит информацию о пользователях, которые авторизовывались на сайте  
  
- **User\_like\_breeds**:   
**BreedId** – INT,   
**UserId** – INT  
Таблица содержит информацию о том, каким породам поставил лайк каждый юзер (см. подробнее связи между таблицами ниже)  
  
- **Cards**:   
**id** – INT,   
**title** – VARCHAR(30),  
**content** – TEXT,  
**image** – VARCHAR(200),  
**UserId** – INT)  
Таблица содержит информацию, которая показывается на странице сообщества (пользовательские карточки)  
  
EER диаграмма описанной базы данных:  


Как можно заметить, между таблицами действуют различные связи:  
- Между таблицами **Breeds** и **Likes** построена связь «Один к одному» (см. транзакции), причем связь identifying (запись в Likes не может существовать, если в графе BreedId указан id несуществующей породы)  
- Между таблицами **Users** и **Cards** построена связь«Один ко многим», причем non-identifying (одному пользователю могут соответствовать несколько карточек, при этом каждой карточке соответствует один пользователь)  
- Между таблица **Users** и **Likes** построена связь «Многие ко многим», поскольку одному и тому же пользователю могут понравиться разные породы и в то же время одна порода может понравиться разным пользователям (связь non-identifying, т.к. потенциально порода может никому не понравиться)  
  
**Транзакции**В ходе разработки было принято решение использовать транзакции там, где это было уместно. Основная причина такого подхода заключается в том, что транзакции дают возможность выполнить последовательный набор инструкций как атомарное действие. При этом, если возникнет ошибка в ходе выполнения одной из внутренних инструкции транзакции, то в этом случае транзакция откатит систему до состояния, в котором она была до начала выполнения транзакции. Этот подход был удобен при проектировании исходной базы данных:  
1) При создании новой породы выполняются две инструкции:  
CREATE для таблицы **Breeds** (создание самой породы непосредственно)  
CREATE для таблицы **Likes** (по умолчанию каждая новая порода имеет 0 лайков)  
Таким образом, данная транзакция позволяет автоматически создавать запись в подчиненной таблице.  
  
2) Когда пользователь хочет поставить/убрать лайк:  
SELECT в таблице **User\_likes\_breeds** (если пользователь уже ставил лайк данной породе, то запись в таблице будет найдена)  
В зависимости от результата SELECT:  
 - если запись не была найдена, то CREATE в таблице **User\_like\_breeds** и UPDATE поля count в таблице **Likes** (увеличения счетчика лайков для той породы, лайк которой был поставлен)  
 - если запись была найдена, то DELETE в таблице **User\_like\_breeds** и UPDATE в таблице **Likes** (уменьшение счетчика лайков для той породы, лайк который был убран)  
  
**Технологии разработки**В качестве СУБД будет использоваться MySQL   
Для проектирования баз данных будет использоваться MySQL Workbench  
Для создания оболочки сайта (frontend часть) будет использоваться React  
Для создания серверной части сайта (backend часть) будет использоваться NodeJS с фреймворком Express  
Язык программирования - JavaScript  
Для поддержки совместной разработки будет использоваться GitHub.  
  
**Команда**Адамович Анатолий, Егор Кудряшов (группа 331)