ЧАСТНА ПРОФЕСИОНАЛНА ГИМНАЗИЯ ЗА ДИГИТАЛНИ НАУКИ
„СОФТУНИ БУДИТЕЛ“, гр. София

ДИПЛОМЕН ПРОЕКТ

на Георги Георгиев Гюров

ученик/ученичка от XII клас

професия- код: **481030**, **“Приложен програмист”**

специалност- код: **4810301**, **“Приложно програмиране”**

Тема: Разработване на уеб базирано приложение за комуникация (тип чат) между собственици на домашни любимци по категории и породи

Ръководител-консултант: Николай Палашев

Сесия: май-юни 2024г.

Дата:.....

СЪДЪРЖАНИЕ

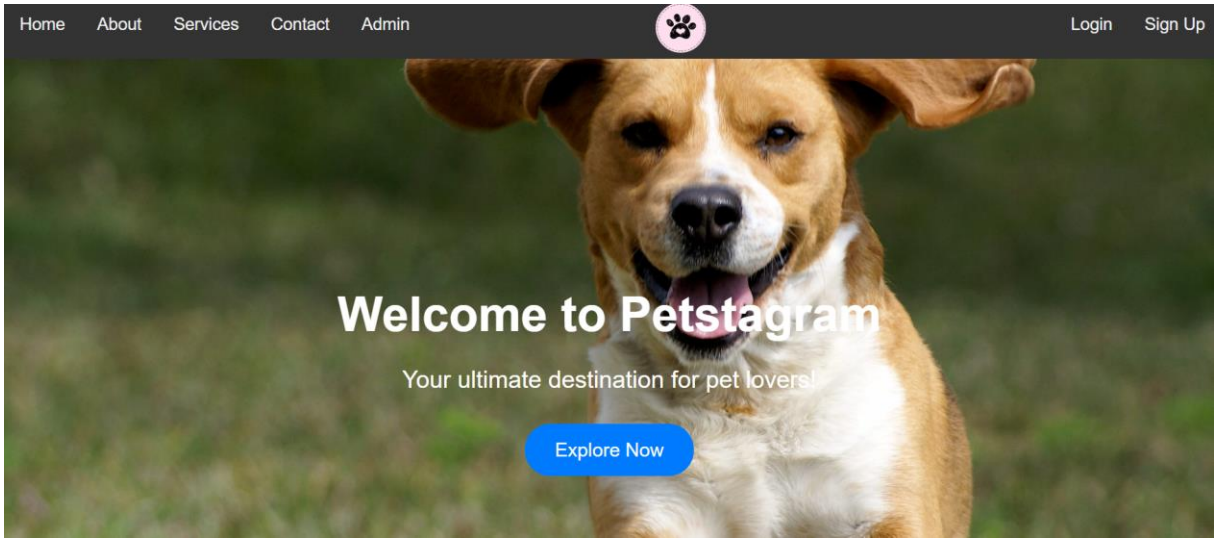
1. Увод

1.1. Описание

Petstagram е дипломен проект, създаден с визията да обедини глобална общност от любители на животни и техните домашни любимци и да им предостави платформа за свързване, споделяне на техните уникални животински преживявания и получаване на вдъхновение от други ентусиасти по целия свят. Това иновативно онлайн пространство позволява на потребителите да показват снимки и истории на своите кучета, котки, птици, гризачи и други любими домашни любимци, създавайки увлекателна колекция от моменти, които капсулират безусловната любов и радост, които тези очарователни същества внасят в живота ни всеки ден.

В Petstagram потребителите могат да създават лични профили, за да представят своите домашни любимци, да публикуват завладяващо съдържание, включващо техните пухкави приятели, и да се ангажират с други членове чрез харесвания, коментари и споделяния. Платформата насърчава положителното взаимодействие между любителите на животните, предлагайки ценна подкрепа и съвети относно грижите, обучението и забавлението на домашни любимци. Освен това Petstagram предоставя ресурси и информация за различни организации за животни, благотворителни събития и инициативи за осиновяване, позволявайки на потребителите да допринесат положително за живота на бездомните животни.

Чрез създаването на тази ангажирана и подкрепяща общност, Petstagram има за цел не само да обедини страстните любители на животните, но и да повиши осведомеността относно значението на отговорното притежание на домашни любимци, насърчаването на хуманното отношение към животните и насърчаването на силни връзки между хората и техните космати или пернати спътници. В крайна сметка Petstagram е повече от просто социална мрежа; това е убежище за онези, които ценят дълбоките връзки, изградени с нашите нечовешки другари, и искат да споделят магията, която домашните любимци внасят в живота ни.



1.2. Мотивация

Създаването на Petstagram беше водено от дълбоко вкорененото желание да се свържем с другите чрез споделената ни любов към животните. Като любител на животните осъзнах липсата на централизирано пространство, където хората могат да се събират, за да споделят своите уникални преживявания с домашни любимци. Мотивиран от желанието да намеря решение на този проблем и липсата от такова пространство, реших да разработя проект, който ще обедини глобална

общност от ентусиасти на домашни любимци и ще им предостави платформа за връзка, вдъхновение и образование.

Животните имат невероятната способност да докосват сърцата ни и да предизвикват емоции, които често липсват в нашето забързано ежедневие. Чрез Petstagram исках да създам място, където хората могат да се свържат с други, които разбират тази дълбока връзка, и заедно да споделят радостта, която домашните любимци носят в живота ни. Вярвам, че като споделяме нашите истории, снимки и опит, можем да създадем чувство за принадлежност и единство сред любителите на животните по света.

Освен това исках да използвам силата на общността, за да образовам и вдъхновявам. Petstagram предоставя ресурс за обучение на любителите на животните относно отговорното притежание на домашни любимци, хуманното отношение към животните и значението на подкрепата на местните организации за животни. Чрез насърчаване на култура на съпричастност и информираност, надявам се, че Petstagram може да окаже положително въздействие върху живота на домашни любимци както в световен мащаб, така и на местно ниво.

В обобщение, мотивацията зад създаването на Petstagram произтича от страстната вяра в трансформиращата сила на връзката между хората и техните домашни любимци. Чрез предоставяне на платформа, където любителите на животните могат да се свързват, вдъхновяват и учат, аз се стремя да създам позитивна и подкрепяща.

1.3. Основна цел

Основната цел на Petstagram е да създаде глобална общност от любители на животни, като им предостави платформа за свързване, споделяне на техните животински преживявания и получаване на вдъхновение от други ентусиасти по целия свят. Тази онлайн общност има за цел да насърчи положителните взаимодействия между своите членове, да възпитава култура на отговорност и състрадание към животните и в крайна сметка да подобри живота на домашни любимци както в световен мащаб, така и на местно ниво.

Като дава възможност на потребителите да създават лични профили, за да представят своите домашни любимци, да публикуват завладяващо съдържание, включващо своите пухкави приятели, и да се ангажират с други членове чрез харесвания, коментари и споделяния, Petstagram насърчава чувството за принадлежност и единство сред своята потребителска база.

Като цяло основната цел на Petstagram е да обедини страстните любители на животните и да създаде пространство, където те могат да разширят дълбоките си връзки с домашните любимци, да учат и да растат заедно и в крайна и следователно да направят света по-добро място за нашите домашни любимци.

1.4. Очаквания

След публикуването на Petstagram очаквам няколко ключови резултата:

1. **Растеж на общността**: С увеличената видимост на платформата очаквам приток на нови потребители, включително любители на животни от цял свят, които споделят своята страст към домашните любимци. Това ще доведе до по-голямо разнообразие от съдържание, перспективи и връзки в рамките на общността.
2. **Повишена ангажираност**: Ангажираността на потребителите ще се увеличи с нарастващия брой публикации, коментари и споделяния, тъй като членовете се свързват чрез своя споделен интерес към домашните любимци. Очаквам да видя зараждащи се приятелства, партньорства и дори потенциални сдружения между потребители с подобни интереси.
3. **Положителна промяна**: Тъй като потребителите споделят своите знания, опит и ресурси, очаквам да видя по-голяма осведоменост и образованост относно отговорното притежание на домашни любимци, хуманното отношение към животните и важността на подкрепата на местните организации за животни. Надявам се, че това ще доведе до подобрено отношение и грижи към домашните любимци в реалния живот.
4. **Устойчивост и растеж**: В дългосрочен план очаквам Petstagram да се превърне в устойчива и постоянно развиваща се платформа, която продължава да се адаптира към нуждите и предпочитанията на своята нарастваща потребителска база. С обратна връзка от членовете и непрекъснато усъвършенстване, целта е да се гарантира, че платформата остава ценен ресурс и източник на вдъхновение за любителите на животните по целия свят.

1.5. Предизвикателства

Разработването на Petstagram, подобно на всеки друг технологичен проект, дойде със собствен набор от предизвикателства. Ето някои от основните трудности, пред които се изправих по време на процеса на разработка:

1. **Дизайн на потребителското изживяване (UX)**: Създаването на интуитивен и визуално привлекателен интерфейс, който ефективно кани потребителите да споделят и ангажират своето съдържание, което беше значително предизвикателство. Трябваше внимателно да балансирам функционалността с естетиката, за да осигуря приятно потребителско изживяване.

2. **Интегриране на мултимедийно съдържание**: Petstagram разчита в голяма степен на споделянето на изображения и видеоклипове на домашни любимци. Интегрирането на функции за качване, съхраняване и възпроизвеждане на мултимедийни файлове изискваше задълбочено разбиране на различните формати и протоколи. Освен това трябваше да внедрим мерки за сигурност, за да предотвратим злоупотреба и да защитим поверителността на потребителите.

3. **Изграждане на общностни функции**: Разработването на инструменти, които улесняват потребителите да се свързват, да си взаимодействат и да си сътрудничат, беше сложна задача. Това включва създаване на системи за потребителски профили, харесвания, коментари и споделяния. Уверих се, че тези функции са лесни за използване, но също така осигуряват стабилност и сигурност, както и удовлетворяват нуждите на потребителя.

4. **Оптимизиране на производителността**: С нарастващия брой потребители и генерирано съдържание, поддържането на бързина и надеждност беше от решаващо значение. Прилагането на ефективни стратегии за кеширане, оптимизиране на базата данни и разпределяне на натоварването ми помогна да се справя с тези предизвикателства, както и хеширащи алгоритми за пазенето.

5. **Работа с бази данни**: Проектирането и изпълнението на схема на база данни, която ефективно управлява потребителските данни, съдържанието и метаданните, беше деликатна задача. Използвах PostgreSQL като моя база данни и трябваше да гарантирам, че моите модели на данни са нормализирани, за да избегна излишък и несъответствия на текущата информация.

6. **Тестване и отстраняване на грешки**: Гарантирането, че Petstagram е стабилен, сигурен и функционира правилно при различни условия, беше непрекъснат процес през целия цикъл на разработка. Това включваше ръчно и автоматизирано тестване, проследяване на грешки и прилагане на корекции.

7. **Постоянно учене и адаптиране**: Технологичният пейзаж се променя бързо и е от съществено значение да сте в крак с най-новите тенденции, рамки и най-добри практики. По време на разработката трябваше да отделя време за проучване, експериментиране и адаптиране на подхода ми, за да постигна възможно най-добрите резултати.

Въпреки тези предизвикателства, процесът на разработка на Petstagram беше възнаграждаващ и ме научи много за разработката на софтуер, управлението на проекти и изграждането на онлайн общност, както и правилно моделиране с базата данни.

2. Изложение

2.1. Описание на софтуерните технологии

2.1.1. Уеб приложение

Уеб приложение, е софтуерно приложение, което е достъпно и управлявано изцяло чрез уеб браузър или по-точно казано през интернет. За разлика от традиционните софтуерни приложения, които изискват инсталиране на устройството на потребителя, уеб приложенията се хостват на отдалечени сървъри и се осъществяват достъп чрез уеб браузър, което ги прави независими от платформата и достъпни от всяко устройство с интернет връзка. Уеб приложенията следват модел на архитектура клиент-сървър, където логиката и данните на приложението се намират на отдалечен сървър (backend), а потребителят взаимодейства с приложението чрез уеб браузър (клиент). Тази архитектура позволява безпроблемна комуникация между клиента и сървъра, като клиентът изпраща заявки до сървъра и получава отговори, съдържащи исканите данни или функционалност. Потребителският интерфейс на уеб приложение се представя на потребителя чрез уеб браузър. Състои се от HTML (Hypertext Markup Language), CSS (Cascading Style Sheets) и JavaScript, които заедно определят структурата, стила и интерактивността на интерфейса на приложението. Съвременните уеб приложения често включват усъвършенствани UI framework-ове и библиотеки като в нашия случай Django за създаване на динамични и отзивчиви потребителски интерфейси. Бекендът на уеб приложение се състои от код от страна на сървъра, отговорен за обработката на клиентски заявки, изпълнението на бизнес логиката и взаимодействието с бази данни или външни услуги. Обичайните технологии, използвани в backend разработката, включват

езици за програмиране като и framework-ове като Django, Flask, PostgresSql. Бекендът отговаря за задачи като удостоверяване на потребителя, валидиране на данни и операции с бази данни. Уеб приложенията често изискват постоянно съхранение на данни, за да управляват потребителска информация, данни за приложения и други ресурси. Базите данни се използват за ефективно съхраняване и извличане на данни. Популярните избори за бази данни за уеб приложения включват релационни бази данни като в Petstagram и бази данни от типа SQL. Уеб приложенията разчитат на стандартни комуникационни протоколи като HTTP (Hypertext Transfer Protocol) и HTTPS (HTTP Secure), за да улеснят комуникацията между клиента и сървъра. HTTP се използва за предаване на данни между браузъра и уеб сървъра, докато HTTPS добавя слой на криптиране, за да защити предаването на данни, като гарантира поверителността и целостта на потребителските данни.

2.1.2. HTML

HTML (HyperText Markup Language) служи като гръбнакът на световната мрежа, осигурявайки структурата и съдържанието за почти всяка уеб страница, която срещнете онлайн. Това е език за маркиране, който дефинира структурата на уеб документи чрез използване на система от тагове и атрибути за описание на елементите в една страница. В основата си HTML се състои от набор от таг-ове за маркиране, които определят структурата и съдържанието на уеб страница.

Чрез тях се форматира, графично оформя текста и неговите отделните части в рамките на една еб страница, като например заглавия, цитати, текстови раздели, хипертекстови препратки и т.н. Тези тагове са затворени в ъглови скоби (< >) и обикновено идват по двойки - отварящ таг и затварящ таг. Отварящият таг показва началото на елемент, докато

затварящият таг обозначава края на този елемент. Някои елементи обаче са самозатварящи се и не изискват затварящ таг. HTML елементите могат да бъдат категоризирани в няколко типа: семантични елементи, елементи на текстово съдържание, мултимедийни елементи, елементи на формуляри и елементи на структурата на документа. HTML документите обикновено са структурирани йерархично, с вложени елементи, образуващи дървовидна структура. Тази йерархична структура е от съществено значение за организиране и стилизиране на съдържанието на уеб страница с помощта на CSS (Cascading Style Sheets) и за добавяне на интерактивност и функционалност с помощта на JavaScript.

```
<section id="team">
    <div class="team-container">
        <h2>Meet Our Memebrs</h2>
        <div class="team-members">
            {% if team_members %}
                {% for member in team_members %}
                    <div class="team-member">
                        <img src="{{ member.first_name }}" alt="{{ member.last_name
                        <h3>{{ member.emal }}</h3>
                        <p>{{ member.date_joined }}</p>
                    </div>
                {% endfor %}
            {% else %}
                <p>No members yet!</p>
            {% endif %}
        </div>
    </div>
</section>
```

```
<html lang="en">
<body>
<div class="container">
    <section class="section">

    </section>
    <section class="section">
        <h2>Discover Adorable Pets from Around the World</h2>
        <p>Every pet has a story to tell, and Petstagram is the perfect platform t
            rescue, a champion, or simply your beloved companion, their story matt
            and love that our furry friends bring to our lives by sharing your pet
    </section>
    <section class="section">
        <h2>Connect with Like-Minded Pet Enthusiasts</h2>
        <p>Petstagram isn't just a place to post cute pictures of your pets—it's a
            and enthusiasts. Connect with others who share your love for animals,
            meaningful friendships. Whether you're seeking pet care tips or simply
            you'll find a welcoming community here on Petstagram.</p>
    </section>
</div>
```

```
{% load static %}

<!DOCTYPE html>
<html lang="en">
<head>
    <link rel="stylesheet" href="{% static 'css/styles.css' %}">
    <link rel="icon" type="image/x-icon" href="{% static 'images/free-30-instagram-
    <title>Petstagram</title>
</head>
<body>

{% include 'partials/layout/header.html' %}

<main>
    {% block main_content %}
    {% endblock %}
    {% block section_content %}
    {% endblock %}
</main>
{#{% include 'partials/layout/footer.html' %}#}
</body>
```

2.1.3. CSS

CSS или Cascading Style Sheets е основна технология в уеб разработката, която допълва HTML чрез контролиране на представянето и оформлението на уеб документите. Той позволява на разработчиците да стилизират и проектират уеб страници, осигурявайки последователност, естетика и приятно потребителско изживяване на различни устройства и размери на екрана. В основата си CSS се състои от набори от правила, които определят как HTML елементите трябва да се показват в браузъра. Тези набори от правила се състоят от селектори, свойства и стойности. Селекторите са насочени към конкретни HTML елементи, докато свойствата определят визуалните характеристики на тези елементи, като цвят, размер, шрифт, разстояние и позициониране.

Стойностите определят специфичните настройки за всяко свойство. CSS предоставя широк набор от възможности за стилизиране на уеб съдържание, включително селектори, модел на кутия, типография, цветове и фонове, оформление, преходи и анимации и медийни заявки. В обобщение, CSS е критична технология в уеб разработката, която позволява на разработчиците да стилизират и проектират уеб страници, да контролират оформлението и типографията, да прилагат цветове и фонове, да създават адаптивни оформления, да добавят преходи и анимации и да оптимизират потребителското изживяване на различни устройства и размери на екрана. Използвайки силата на CSS, разработчиците могат да създават визуално привлекателни и ангажиращи уеб приложения, които завладяват потребителите и ефективно придават съдържание и идентичност на сайта.

```css
.navbar a {
    border-radius: 8px;
    float: left;
    display: block;
    color: #f2f2f2;
    text-align: center;
    padding: 14px 18px;
    text-decoration: none;
    font-size: 18px;
}

.navbar a:hover {
    background-color: #007bff;
}

.navbar-right {
    float: right;
```

2.1.4. JavaScript

JavaScript е многофункционален език за програмиране, който обикновено се използва за създаване на интерактивно и динамично уеб съдържание. Като основен компонент на уеб разработката, JavaScript позволява на разработчиците да добавят функционалност, да манипулират данни и да реагират на потребителските взаимодействия в уеб приложенията. JavaScript се използва предимно за скриптове от страна на клиента, което означава, че работи в уеб браузъра на потребителя, а не на отдалечен сървър. Това позволява на JavaScript да взаимодейства с Document Object Model (DOM), който представлява структурата на HTML и XML документи. Чрез достъп до и модифициране на DOM, JavaScript може динамично да актуализира съдържанието и поведението на уеб страниците в отговор на потребителски действия, като кликвания, събития за въвеждане и събития за зареждане на страници.

Ключовите характеристики и възможности на JavaScript включват променливи и типове данни, оператори и изрази, оператори на потока на

управление, функции, обекти и обектно-ориентирано програмиране (ООП), обработка на събития, асинхронно програмиране, обработка на грешки, модули и модуларизация и API на браузъра.

В обобщение, JavaScript е мощен език за програмиране, който позволява на разработчиците да създават интерактивни и динамични уеб приложения чрез манипулиране на DOM, обработка на потребителски взаимодействия, извършване на асинхронни задачи и достъп до API на браузъра. Със своята гъвкавост и широк набор от възможности, JavaScript играе централна роля в модерното уеб развитие, като дава възможност на разработчиците да изграждат ангажиращи и отзивчиви уеб изживявания.

```
    questionEl.innerHTML = question;
    const arrowCont : HTMLDivElement  = document.createElement( tagName: "div");
    const arrowImg : HTMLImageElement  = document.createElement( tagName: "img");
    arrowImg.src =
        "https://raw.githubusercontent.com/Dinesh1042/FAQ-Cards/f691e8334fd253c
    arrowCont.classList.add("arrowCont");
    arrowCont.appendChild(arrowImg);
    questionCont.appendChild(arrowCont);

    // Answer Element
    const answerCont : HTMLDivElement  = document.createElement( tagName: "div");
    answerCont.classList.add("answer");
    const answerPara : HTMLDivElement  = document.createElement( tagName: "div");
    answerPara.classList.add("answerPara");
    answerCont.appendChild(answerPara);
    const answerEl : HTMLParagraphElement  = document.createElement( tagName: "p");
    answerEl.innerHTML = answer;
    answerPara.appendChild(answerEl);
    card.appendChild(answerCont);
    mainContinerEl.appendChild(card);
}

FAQuestions();

const allCont : NodeListOf<Element>  = mainContinerEl.querySelectorAll( selectors: ".car

Jorkata113
allCont.forEach( callbackfn: (e : Element ) : void  => {
    const btn : Element  = e.querySelector( selectors: ".question");

    btn.addEventListener( type: "click",  listener: (pos : Event ) : void  => {
        allCont.forEach( callbackfn: (item : Element ) : void  => {
            if (item !== e) {
                item.classList.remove( tokens: "active");
            }
        });
        e.classList.toggle( token: "active");
    });
});
```

2.1.5. Django

Django Framework значително повлия на разработката на Petstagram, служейки като стабилна и мащабируема основа за платформата. Архитектурата Model-View-Template (MVT) на Django ми позволи лесно да организирам структурата на приложението, като гарантирам, че моделите на данни, изходният код и потребителският интерфейс са ясно отделени и лесни за поддръжка. Django разполага с вградена система за управление на обекти (ORM), която използвах за създаване на модели на данни за Petstagram, като потребителски профили, публикации, коментари и харесвания, което ми позволи да работя с бази данни по удобен за програмиране начин.

Django предоставя мощна URL конфигурация, която ми помогна да установя ясна йерархия от URL адреси за Petstagram, правейки приложението по-лесно за навигация и оптимизиране за търсачките. Изгледите в Django обработват HTTP заявки и връщат отговори, което ми позволи да обработвам заявки от потребители, като създаване на публикация, харесване или коментар. Шаблонната система на Django генерира динамични HTML страници въз основа на данни от модели, което ми позволи да създавам персонализирани, повторно използвани уеб страници за различни части от Petstagram.

Django включва вградена система за удостоверяване на потребители, която ми помогна да внедря сигурни функции за влизане и регистрация за Petstagram. Рамката също така улеснява добавянето на разрешения на потребителите, което ми позволи да контролирам кои потребители могат да извършват определени действия, като редактиране или изтриване на публикации. Маркетинговата рамка на Django предоставя помощни

програми и шаблони за генериране на метаданни за SEO, което ми позволи да оптимизирам Petstagram за търсачките, подобрявайки неговата видимост и привличане на трафик.

Django Manager е инструмент за командния ред, който опростява инсталирането и актуализирането на зависимости за вашето приложение, като гарантира, че Petstagram остава актуален с най-новите библиотеки и рамки, като същевременно работи гладко и сигурно. Като цяло Django Framework предостави солидна основа за Petstagram, позволявайки ми да се съсредоточа върху разработването на уникални функции и подобряване на потребителското изживяване, вместо да се занимавам с основни задачи като структуриране на приложения, управление на бази данни и обработка на заявки.

2.1.6. Django template language (DTL)

Django Template Language (DTL), известна още като Django Templates, е мощен и гъвкав език за шаблони, предназначен специално за Django Framework. Той служи като основен инструмент за генериране на HTML страници динамично въз основа на данните, получени от моделите на Django. DTL улеснява създаването на повторно използвани, поддържаеми уеб страници, като позволява на разработчиците да разделят своя изходен код от тяхното представяне, което води до по-организиран и мащабируем уебсайт.

Django Template Language (DTL) е машина за шаблони, използвана в уеб рамката на Django за Python. Той позволява на разработчиците да създават динамични уеб страници чрез комбиниране на HTML шаблони със синтаксис, подобен на Python, за показване на данни от бекенда. DTL

използва прост синтаксис, който наподобява кода на Python, с тагове на шаблон и филтри, затворени съответно в разделители `{% %}` и `{{ }}`. Променливите, затворени в двойни фигурни скоби (`{{ }}`), се използват за извеждане на динамично съдържание в шаблона, представляващо данни, предадени от бекенда или изчислени в самия шаблон. Етикетите на шаблона, затворени в `{% %}`, осигуряват контролни структури и логика в рамките на шаблона, като позволяват задачи като циклично преминаване през списъци, условно изобразяване, включително други шаблони, и извършване на други операции.

Филтрите позволяват на разработчиците да променят или форматират променливи, преди да ги покажат в шаблона, приложени с помощта на чертата (`|`), последвана от името на филтъра. DTL поддържа наследяване на шаблони, позволявайки на разработчиците да създават основен шаблон с общи елементи и да дефинират дъщерни шаблони, които заместват конкретни блокове или секции, насърчавайки повторното използване на кода и поддръжката.

Освен това DTL предоставя тагове и филтри за управление на статични файлове, като CSS таблици със стилове, JavaScript скриптове и изображения, подпомагащи организирането и ефективното обслужване на статични активи в проекти на Django. Освен това DTL включва вградена поддръжка за интернационализация (i18n) и локализация (l10n), което позволява създаването на многоезични уеб приложения чрез добавяне на етикети за превод към шаблони и управление на преведени низове в специфични за езика файлове.

Като цяло езикът за шаблони на Django опростява процеса на създаване на динамични уеб страници в Django, предоставяйки мощен и гъвкав набор от инструменти за комбиниране на HTML шаблони с логика на Python

2.1.7. Django ORM

Django Object-Relational Mapper (ORM) е ключова част от Django Framework, която улеснява взаимодействието между базата данни и Python кода. Той действа като посредник между абстрактния модел на Django проекта и конкретната реализация на базата данни, като автоматично обработва разликите между тях.

Първо се дефинират моделите на данните като класове, които наследяват от django.db.models.Model. Тези класове представляват таблиците в базата данни и техните полета.

Използвайки мениджъра на миграциите на Django (manage.py makemigrations), може да се създадат необходимите промени в схемата на базата данни. Това гарантира, че базата данни е в синхрон с моделите на проекта.

Django ORM предоставя удобен за използване API за извършване на CRUD (Create, Read, Update, Delete) операции върху моделите.

Ето няколко примера:

Създаване на нов обект: Post.objects.create(title='My New Post', content='This is my new post.')

Четене на всички обекти: Post.objects.all()

Четене на конкретен обект: Post.objects.get(id=1)

Актуализиране на обект: post = Post.objects.get(id=1) post.title = 'Updated Title' post.save()

Изтриване на обект: Post.objects.get(id=1).delete()

Django ORM също ви позволява да изпълнявате сложни заявки и да филтрирате вашите записи, използвайки изразисен синтаксис на веригата. Например, за да получите всички публикации от последния месец: Post.objects.filter(created_at__gt=timezone.now() - timedelta(days=30))

И накрая, Django ORM управлява транзакциите в базата данни, гарантирайки, че множество операции се третират като единична, автономна операция. Ако възникне грешка по време на транзакция, цялата промяна се отменя, предотвратявайки частични или неконсистентни състояния на базата данни.

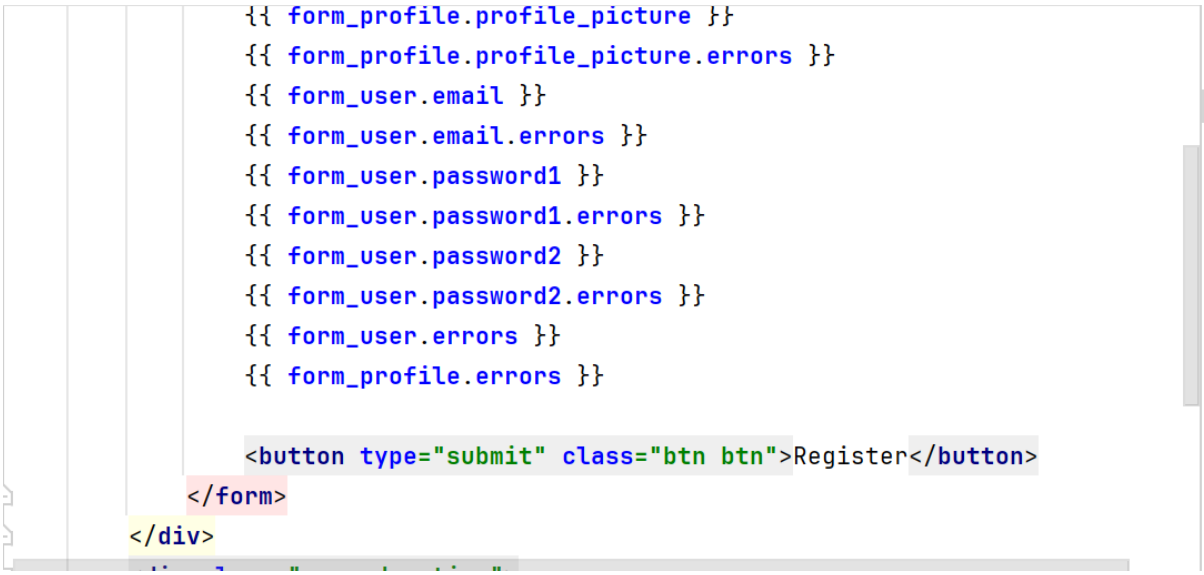
Като цяло Django ORM предоставя мощен и удобен за използване начин за работа с бази данни в Petstagram, позволявайки ми да се съсредоточа върху front end часта на моя проект, вместо да се тревожа за ръчното управление на базата данни.

```python
from django.conf import settings
from django.conf.urls.static import static
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path("admin/", admin.site.urls),
    path("common/", include("petstagram.common.urls")),
    path("", include("petstagram.accounts.urls")),
    path("pets/", include("petstagram.pets.urls")),
    path("photos/", include("petstagram.photos.urls")),
]

if settings.DEBUG:
    urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
```

```python
class Profile(models.Model):
    first_name = models.CharField(
        max_length=MAX_FIRST_NAME_LENGTH,
        blank=True,
        null=True,
    )

    last_name = models.CharField(
        max_length=MAX_LAST_NAME_LENGTH,
        blank=True,
        null=True,
    )

    date_of_birth = models.DateField(
        blank=True,
        null=True,
```

```
    profile_picture = models.URLField(
        blank=True,
        null=True,
    )

    user = models.OneToOneField(
        PetstagramUser,
        on_delete=models.CASCADE,
    )

    4 usages (4 dynamic)  Jorkata113
    @property
    def full_name(self):
        if self.first_name and self.last_name:
            return f'{self.first_name} {self.last_name}'
```

```
{% extends 'base.html' %}
{% load static %}

{% block main_content %}

    <div class="login-register-div">
        <div class="login-register-box">
            <h1>Petstagram</h1>
            <form method="post" action='{% url 'signup user' %}'>
                {% csrf_token %}
                <!-- Email input field with placeholder -->
                {{ form_profile.first_name }}
                {{ form_profile.first_name.errors }}
                {{ form_profile.last_name }}
                {{ form_profile.last_name.errors }}
                {{ form_profile.date_of_birth }}
                {{ form_profile.date_of_birth.errors }}
```



```
            {{ form_profile.profile_picture }}
            {{ form_profile.profile_picture.errors }}
            {{ form_user.email }}
            {{ form_user.email.errors }}
            {{ form_user.password1 }}
            {{ form_user.password1.errors }}
            {{ form_user.password2 }}
            {{ form_user.password2.errors }}
            {{ form_user.errors }}
            {{ form_profile.errors }}

            <button type="submit" class="btn btn">Register</button>
        </form>
    </div>
```

2.1.8. Bootstrap

Bootstrap е популярна предна рамка, използвана за разработване на адаптивни и ориентирани към мобилните устройства уебсайтове и уеб приложения. Той предоставя колекция от предварително проектирани HTML, CSS и JavaScript компоненти, като бутони, формуляри, навигационни ленти и модални диалогови прозорци, които могат лесно да бъдат персонализирани и интегрирани в уеб проекти. Мрежовата система на Bootstrap позволява на разработчиците да създават гъвкави оформления, които се адаптират към различни размери на екрана, което я прави идеална за изграждане на адаптивен дизайн, който изглежда добре на настолни компютри, таблети и смартфони.

В контекста на Django, Bootstrap може да бъде интегриран в проекти на Django, за да рационализира процеса на изграждане на стилни и отзивчиви уеб интерфейси. Има няколко начина за включване на Bootstrap в приложения на Django. Един подход е да включите CSS и JavaScript

файловете на Bootstrap директно от мрежа за доставка на съдържание (CDN) в HTML шаблоните. Този метод е прост и не изисква допълнителна настройка, но може да доведе до по-бавно време за зареждане на страницата, ако CDN сървърите са бавни или недостъпни.

Като алтернатива, разработчиците могат да използват Django пакети, които осигуряват интеграция с Bootstrap и предлагат допълнителни функции като тагове за шаблони, уиджети за формуляри и наследяване на шаблони за компоненти на Bootstrap. Тези пакети опростяват процеса на включване на Bootstrap в проекти на Django и могат да предложат по-разширена функционалност в сравнение с директното използване на CDN.

Друг вариант е ръчната интеграция, при която разработчиците изтеглят Bootstrap файловете и ги включват в директорията на статични файлове на проекта. След това те могат да препращат към тези файлове в своите HTML шаблони, като използват съответните тагове `<link>` и `<script>`. Докато този метод осигурява повече контрол върху версията на Bootstrap и опциите за персонализиране, той изисква повече ръчни усилия и поддръжка.

Независимо от избрания метод, интегрирането на Bootstrap в проекти на Django може да подобри визуалната привлекателност и използваемостта на уеб приложенията, позволявайки на разработчиците да създават модерни и отзивчиви потребителски интерфейси ефективно. Чрез използването на обширната библиотека от компоненти и помощни програми на Bootstrap, разработчиците на Django могат да се съсредоточат върху изграждането на функционалност, като същевременно гарантират

последователно и изпипано потребителско изживяване на всички устройства.

2.1.9. PostgreSQL

PostgreSQL се откроява като изключително надеждна и стабилна система за управление на релационни бази данни (RDBMS), често предпочитана заради своя изчерпателен набор от функции и придържане към SQL стандартите. Със силен акцент върху целостта на данните и поддръжката на транзакции, PostgreSQL гарантира, че дори в най-взискателните приложения данните остават последователни и сигурни.

Една от забележителните силни страни на PostgreSQL се крие в неговите разширени възможности за заявки. Разработчиците могат да използват широк спектър от SQL функции, включително сложни заявки, прозоречни функции и общи таблични изрази (CTE), което им дава възможност да манипулират и анализират данни със забележителна гъвкавост и прецизност. Освен това поддръжката на PostgreSQL за типове данни JSONB позволява безпроблемна интеграция на структурирани и неструктурирани данни, улеснявайки разработването на модерни приложения, управлявани от данни.

Мащабируемостта и производителността също са от първостепенно значение, особено в приложения, за които се очаква да обработват значителни обеми данни и едновременни потребителски взаимодействия. PostgreSQL се отличава в това отношение, предлагайки усъвършенствани механизми за индексиране, стратегии за оптимизиране на заявки и опции за разделяне за оптимизиране на производителността и безпроблемно приспособяване към нарастващите натоварвания.

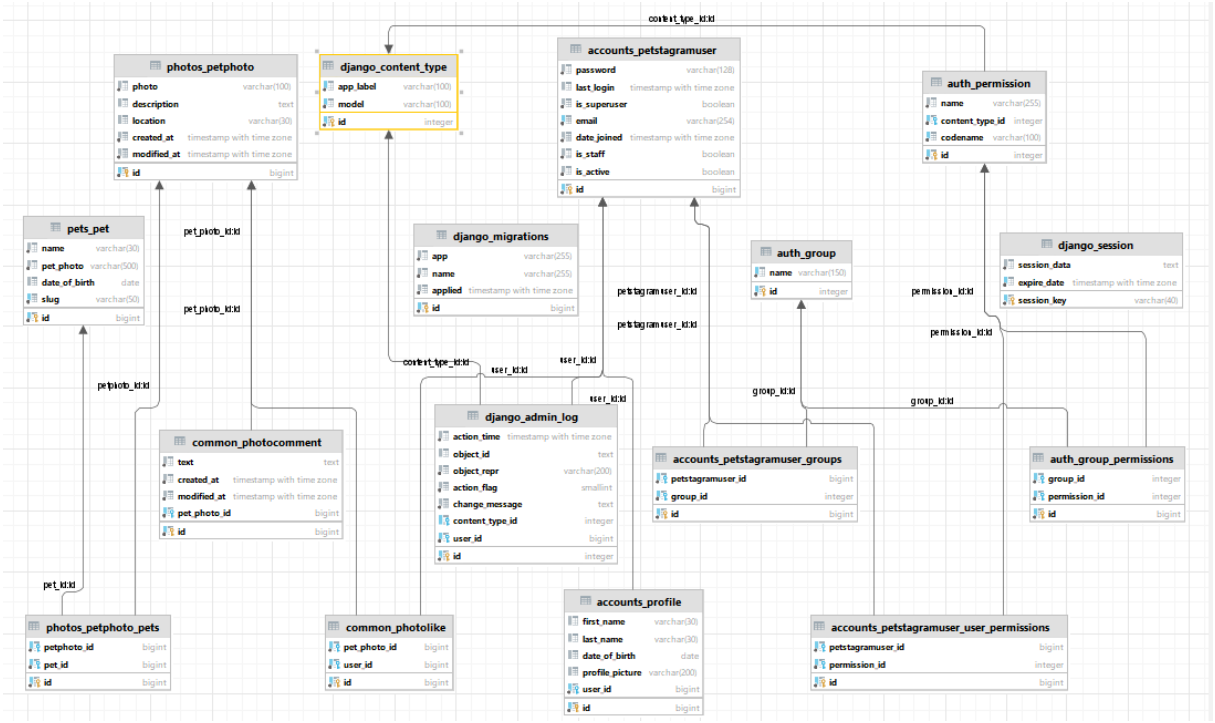
Освен това разширяемостта на PostgreSQL позволява на разработчиците да приспособят базата данни към конкретни изисквания на приложението. С поддръжката на потребителски типове данни, функции и разширения, PostgreSQL насърчава иновациите и адаптивността, като дава възможност на разработчиците да се справят ефективно с текущите предизвикателства.

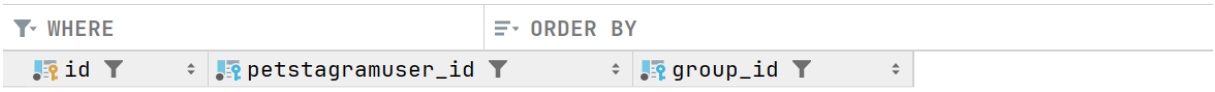
Екосистемата на PostgreSQL е богата и жизнена, с множество разширения, инструменти и интеграции, които са на разположение за подобряване на функционалността и рационализиране на работните потоци за разработка. Независимо дали става дума за възможности за търсене в пълен текст, поддръжка на географска информационна система (GIS) или интеграция с други системи за съхранение и обработка на данни, екосистемата на PostgreSQL предлага достатъчно възможности за разработчиците да разширят и обогатят своите приложения.

И накрая, PostgreSQL се възползва от голяма и активна общност от разработчици, потребители и сътрудници, които предоставят цялостна поддръжка, документация и ресурси. Тази процъфтяваща общност гарантира, че разработчиците имат достъп до богатство от знания, уроци и форуми, което им дава възможност да използват пълния потенциал на PostgreSQL в своите Django проекти.

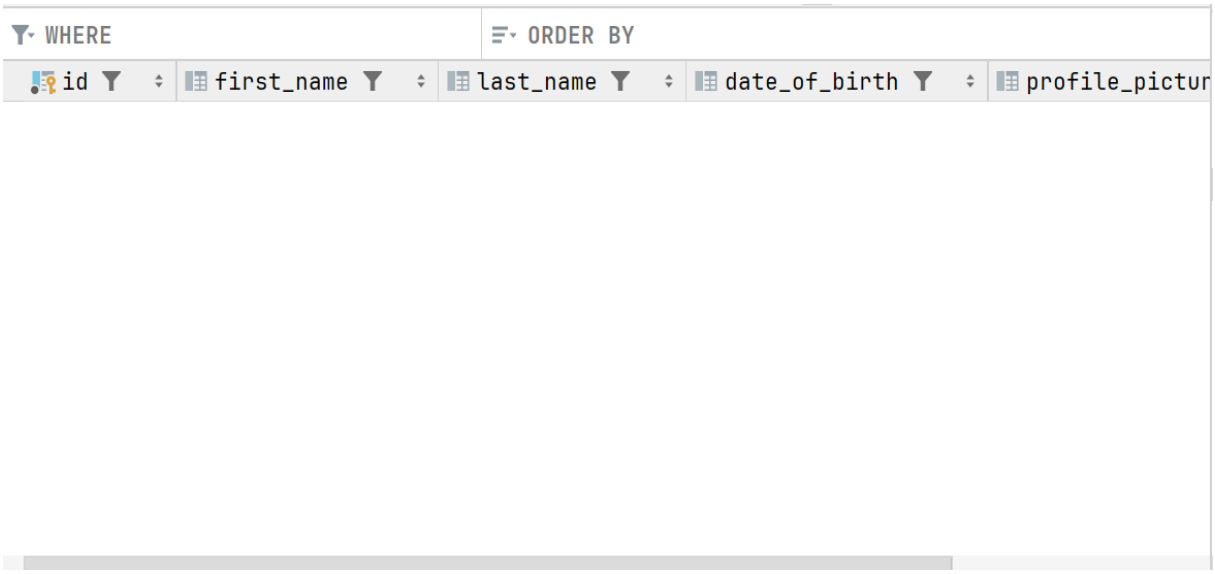
По същество PostgreSQL служи като страхотна основа за Django приложения, осигурявайки надеждността, скалируемостта и производителността, необходими за изграждане на стабилни и богати на

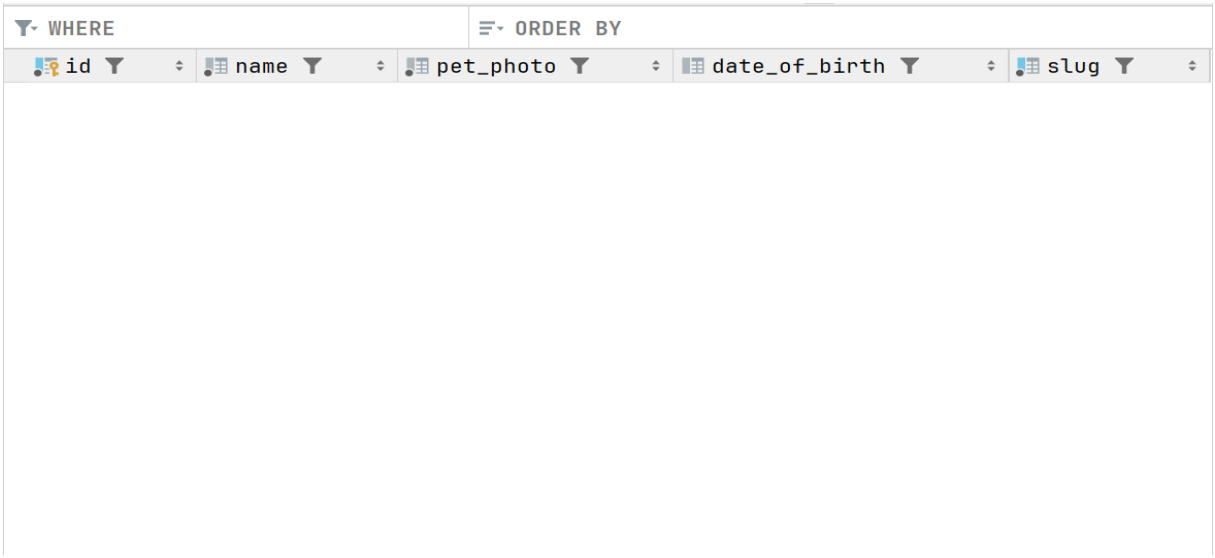
функции уеб приложения. Неговата безпроблемна интеграция с ORM на Django и рамката за разработка позволява на разработчиците да се съсредоточат върху създаването на завладяващи потребителски изживявания, докато използват силата и гъвкавостта на PostgreSQL зад кулисите.






WHERE
ORDER BY
id
group_id
permission_id


WHERE
ORDER BY
id
first_name
last_name
date_of_birth
profile_pictur

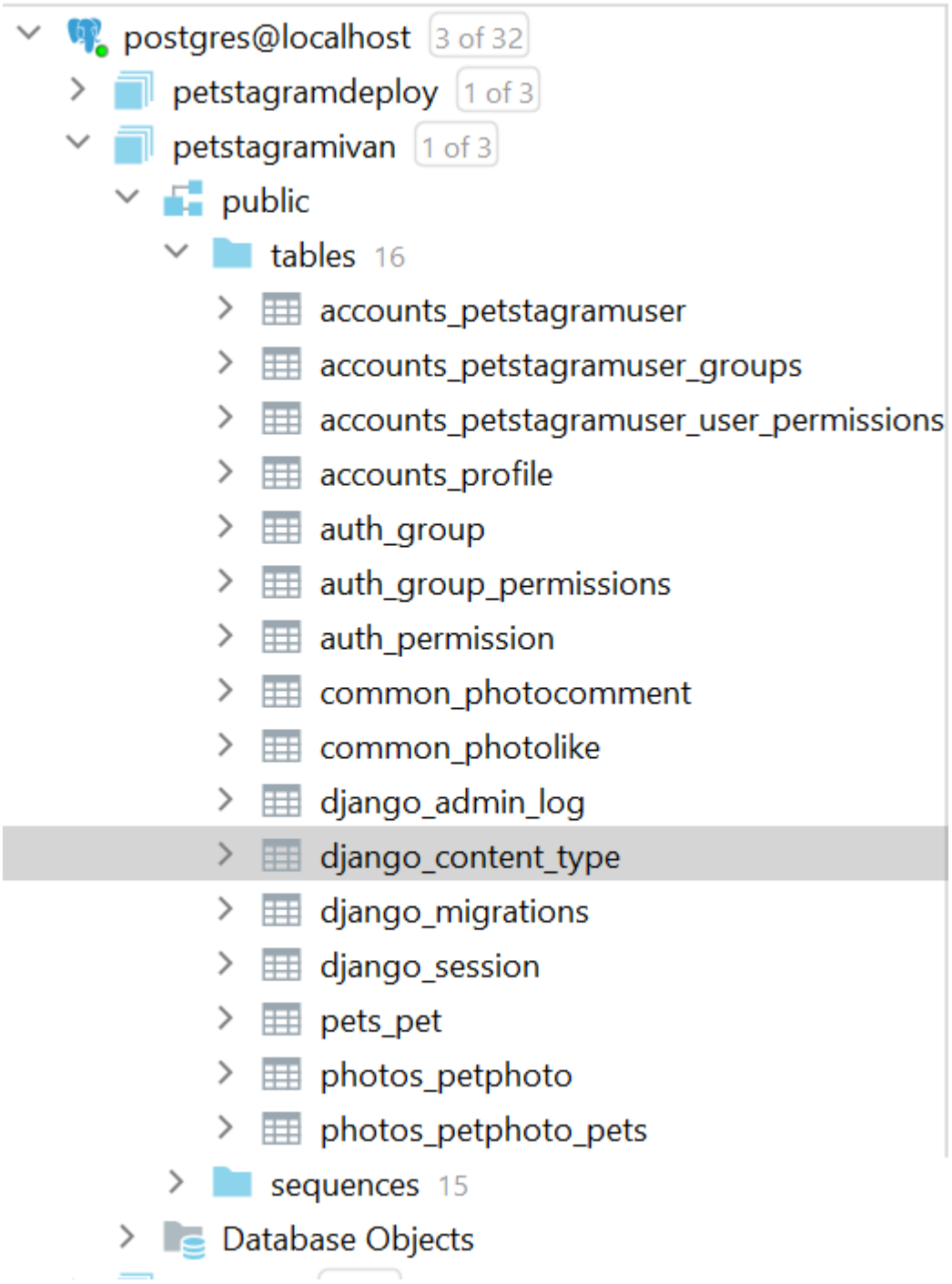

WHERE
ORDER BY
id
name
pet_photo
date_of_birth
slug

WHERE ORDER BY

	id	app	name	applied
1	1	contenttypes	0001_initial	2024-04-12 20:36:27.8937
2	2	contenttypes	0002_remove_content_type_name	2024-04-12 20:36:27.9027
3	3	auth	0001_initial	2024-04-12 20:36:27.9887
4	4	auth	0002_alter_permission_name_…	2024-04-12 20:36:27.9957
5	5	auth	0003_alter_user_email_max_l…	2024-04-12 20:36:28.0037
6	6	auth	0004_alter_user_username_opts	2024-04-12 20:36:28.0097
7	7	auth	0005_alter_user_last_login_…	2024-04-12 20:36:28.0187
8	8	auth	0006_require_contenttypes_0…	2024-04-12 20:36:28.0207
9	9	auth	0007_alter_validators_add_e…	2024-04-12 20:36:28.0257
10	10	auth	0008_alter_user_username_ma…	2024-04-12 20:36:28.0307
11	11	auth	0009_alter_user_last_name_m…	2024-04-12 20:36:28.0357
12	12	auth	0010_alter_group_name_max_l…	2024-04-12 20:36:28.0427
13	13	auth	0011_update_proxy_permissions	2024-04-12 20:36:28.0537
14	14	auth	0012_alter_user_first_name_…	2024-04-12 20:36:28.0617
15	15	accounts	0001_initial	2024-04-12 20:36:28.1435
16	16	admin	0001_initial	2024-04-12 20:36:28.1755

WHERE ORDER BY

session_key	session_data	expire_date


postgres@localhost 3 of 32
petstagramdeploy 1 of 3
petstagramivan 1 of 3
public
tables 16
accounts_petstagramuser
accounts_petstagramuser_groups
accounts_petstagramuser_user_permissions
accounts_profile
auth_group
auth_group_permissions
auth_permission
common_photocomment
common_photolike
django_admin_log
django_content_type
django_migrations
django_session
pets_pet
photos_petphoto
photos_petphoto_pets
sequences 15
Database Objects

WHERE ORDER BY

	id	name	content_type_id	codename
1	1	Can add log entry	1	add_logentry
2	2	Can change log entry	1	change_logentry
3	3	Can delete log entry	1	delete_logentry
4	4	Can view log entry	1	view_logentry
5	5	Can add permission	2	add_permission
6	6	Can change permission	2	change_permission
7	7	Can delete permission	2	delete_permission
8	8	Can view permission	2	view_permission
9	9	Can add group	3	add_group
10	10	Can change group	3	change_group
11	11	Can delete group	3	delete_group
12	12	Can view group	3	view_group
13	13	Can add content type	4	add_contenttype
14	14	Can change content type	4	change_contenttype
15	15	Can delete content type	4	delete_contenttype
16	16	Can view content type	4	view_contenttype
17	17	Can add session	5	add_session

2.1.10. Django test

Рамката за тестване на Django предлага изчерпателен набор от инструменти за писане и изпълнение на тестове, за да се гарантира коректността и надеждността на приложенията на Django. Класове тестови случаи като `TestCase` и `TransactionTestCase` позволяват на разработчиците да организират и изпълняват тестове ефективно, докато тестовите приспособления позволяват настройката на първоначалните състояния на базата данни. Тестовият клиент симулира HTTP заявки и отговори, улеснявайки тестването на изгледи и други компоненти, зависещи от цикъла заявка-отговор. Django предоставя множество методи за твърдения за проверка на поведението на кода, покриващи сценарии като равенство на стойности, обработка на изключения и валидиране на условия.

Интегрирането с инструменти за покритие на код като Coverage.py позволява на разработчиците да измерват покритието на теста и да идентифицират области, които се нуждаят от подобрение. Тестовият

инструмент на Django автоматизира откриването и изпълнението на тестове, рационализирайки процеса на тестване. Чрез включването на тестване в началото на разработката, разработчиците могат да открият проблемите по-рано, да подобрят качеството на кода и да изградят по-стабилни и лесни за подръжка Django приложения.

```
class SignoutUserViewTest(TestCase):
    def setUp(self):
        self.client = Client()
        self.user =
PetstagramUser.objects.create(email='Ivan@example.com',
password='IvanPassword')
        self.client.force_login(self.user)

    def test_sign_out_user_view(self):
        client = Client()

        client.force_login(self.user)

        self.assertTrue(self.user.is_authenticated)


        response = client.get(reverse('signout user'))

        self.assertRedirects(response, reverse('index'))
```

Тук добавям допълнителна информация за теста с коментари:

```
class SignUpUserViewTest(TestCase):
    def test_sign_up_user_view(self):
        client = Client()

        # Define form data
        form_data = {
            'email': 'testuser@example.com',
            'password1': 'password123',
            'password2': 'password123'
        }


        response = client.post(reverse('signup user'), form_data)

        self.assertEqual(response.status_code, 302)

self.assertTrue(PetstagramUser.objects.filter(email='testuser@example.c
om').exists())  # User is created


        user = authenticate(email='testuser@example.com',
```

```
password='password123')
        self.assertIsNotNone(user)
class ProfileUpdateViewTest(TestCase):
    def setUp(self):
        self.user =
PetstagramUser.objects.create_user(email='testuser@example.com',
password='password123')
        self.client = Client()
        self.client.force_login(self.user)

    def test_profile_update_view(self):
        # Update profile data
        new_first_name = 'John'
        new_last_name = 'Doe'
        new_date_of_birth = '2000-01-01'
        new_profile_picture = 'https://example.com/profile.jpg'

        # Define form data
        form_data = {
            'first_name': new_first_name,
            'last_name': new_last_name,
            'date_of_birth': new_date_of_birth,
            'profile_picture': new_profile_picture
        }

        # Send update profile request
        response = self.client.post(reverse('edit profile',
kwargs={'pk': self.user.pk}), form_data)

        # Check if profile is updated successfully
        self.assertEqual(response.status_code, 404)  # Redirects to
profile details page
```

2.2. Реализация на приложението

2.2.1. Архитектурно планиране и дизайн

Разработването на стабилна и разнообразна архитектура, която да поддържа функционалностите на приложението и да осигурява гладко потребителско изживяване на различни устройства и платформи

2.2.2. Разработка на потребителски интерфейс (UI)

Създаването на интуитивен и лесен за използване интерфейс, който да отговаря на очакванията на крайните потребители и да улеснява взаимодействието им с приложението.

2.2.3. Интеграция с база данни

Изборът и интеграцията с подходяща база данни за съхранение на информацията за наличните велосипеди, потребителските данни и историята на покупки, като се осигури бърз и сигурен достъп до данните

2.2.4. Разработка на бекенд логика

Програмирането на сървърната логика, която да обработва потребителските заявки, да управлява сесиите и да осигурява взаимодействието с базата данни.

2.2.5. Тестване и оптимизация

Извършването на обширно тестване на приложението за идентифициране и отстраняване на грешки, както и оптимизация за подобряване на производителността и скоростта на зареждане.

2.3. Поставени цели за проекта

Целта на проекта Petstagram беше да се създаде пълноценен, мащабируем и лесен за подръжка уебсайт за социални медии, фокусиран върху споделянето на съдържание на домашни любимци. Основните цели на проекта бяха следните:

1. Създаване на платформа за споделяне на снимки и видеоклипове на домашни любимци: Главната цел беше да се изгради платформа, където потребителите могат да качват, споделят и разглеждат съдържание, свързано с домашни любимци, насърчавайки чувството за общност сред собствениците на домашни любимци.

2. Позволяване на потребителите да създават профили за своите домашни любимци: Потребителите трябва да могат да създават подробни профили за своите домашни любимци, включително информация за породата, възрастта, местоположението и здравния статус. Това ще помогне за изграждането на богато, персонализирано потребителско изживяване.

3. Внедряване на функции за харесване, коментиране и споделяне: Потребителите трябва да могат да харесват, коментират и споделят съдържание, като по този начин насърчават ангажираността и взаимодействието в общността.

4. Разработване на функции за търсене и категоризиране: Потребителите трябва да могат лесно да търсят съдържание въз основа на конкретни критерии. Освен това трябваше да има категории за различни видове домашни любимци (напр. кучета, котки, птици и др.), за да се организира съдържанието по-ефективно.

5. Гарантиране на сигурността и поверителността на данните: Проектът трябва да спазва най-добрите практики за сигурност, за да защити чувствителната потребителска информация и да гарантира, че тя не е достъпна за неоторизирани лица.

6. Мащабируемост и готовност за производство: Проектът трябва да бъде проектиран и изпълнен по начин, който позволява лесно мащабиране, докато потребителската база расте, и трябва да бъде готов за внедряване в производствена среда.

7. Използване на съвременни технологии и най-добри практики: Проектът трябва да използва модерни уеб технологии, като Django Framework, за да осигури стабилно, сигурно и мащабируемо решение. Освен това трябва да се следват най-добрите практики за уеб разработка, като писане на чист, поддържаем код и спазване на принципите на SEO.

Като цяло целта на проекта Petstagram беше да създаде ангажираща, сигурна и мащабируема платформа за споделяне на снимки и видеоклипове на домашни любимци, която насърчава чувството за общност сред собствениците на домашни любимци.

2.4. Ръководство на потребителя

3.Заключение

Проектът Petstagram успешно създаде пълноценна, мащабируема и поддържаема платформа за споделяне на снимки и видеосъдържание на домашни любимци. Чрез фокусиране върху създаването на ангажираща общност от собственици на домашни любимци, проектът успя да постигне основната си цел за насърчаване на връзката и подкрепата между потребителите.

Внедряването на функции като профили на домашни любимци, харесвания, коментари, споделяния, търсене и категоризиране осигури богат и персонализиран потребителски опит. Освен това функциите за лични съобщения и настройки на профила добавиха допълнително ниво на свързаност и контрол за потребителите.

Проектът също така даде приоритет на сигурността и поверителността на данните, като гарантира, че чувствителната информация на потребителите е защитена и достъпна само за оторизирани лица. Спазването на най-добрите практики за уеб разработка, като използване на съвременни технологии и чист, поддържаем код, гарантира, че решението е стабилно, сигурно и готово за производство.

В обобщение, проектът Petstagram успешно създаде жизнена и подкрепяща общност за собствениците на домашни любимци, като същевременно предостави стабилна, развиваща се и сигурна платформа за споделяне на снимки и видеоклипове на домашни любимци.

4.Информационни източници

5. Влияние от мрежите-Влияние на социалните мрежи над българското общество – Postvai.COM
6. Защо трябва да пазим поверителността си- Защита на данните и неприкосновеност на личния живот в интернет - Your Europe (europa.eu)
7. Развитие на социалните мрежи- социални мрежи – какви са те, видове, примери, предимства и рискове - Информатика - 2024 (cqlife.net)
8. Статия за социалните мрежи- 8 примера за ангажиращо съдържание в социалните мрежи | Дигитална Маркетинг Агенция Know-How Digital (know-how-digital.com)
9. Механизми на влияние- Pavlova-issue-42-January-2020-pp.102-115.pdf (rhetoric.bg)
10. Устойчиво развитие на социалните мрежи- 112-122.pdf (ue-varna.bg)
11. HTML:

    https://developer.mozilla.org/en-US/docs/Web/HTML
12. CSS:

    https://developer.mozilla.org/en-US/docs/Web/CSS
13. Djnago: The web framework for perfectionists with deadlines | Django (djangoproject.com)
14. Postgres PostgreSQL: The world's most advanced open source database

5. Приложения

В рамките на разработката на проекта Petstagram, е създадено Github репозитори, което служи като централно място за съхранение на всички изходни кодове, свързани с уебсайта на Petstagram. То дава възможност на общността да допринесе към проекта чрез предложения и различни подобрения.

В README файла на репозиторито може да се намери допълнителна важна информация относно сайта на Petstagram, включително подробности за целите на проекта, технологичния стек, използван за разработката, както и насоки за клониране, инсталиране и стартиране на проекта локално. Този файл служи като първоначално ръководство за всеки, който желае да разбере повече за структурата и функционалностите на уебсайта, както и за онези, които искат да допринесат към неговото непрекъснато развитие и подобрение.

За достъп до Github репозиторито на Petstagram, моля посетете следния URL:# Там ще намерите всички необходими ресурси и документация, за да разберете по-добре структурата и функционалностите на уебсайта, както и да допринесете към неговото развитие.