|  |  |
| --- | --- |
| Термін | Визначення |
| Models (та пов’язані терміни) | |
| Post | Модель для публікацій на сайті. Включає заголовок, текст, категорію, посилання на пост, автора поста тощо. |
| Category | Модель для категорій, до яких належать пости. Наприклад, "Верстка", "Менеджмент". |
| UserProfile | Модель, що зберігає додаткову інформацію про користувача. Включає поля для імені, електронної пошти, професії, зображення профілю та унікального ідентифікатора. |
| |  | | --- | | LoginRequiredMixin |  |  | | --- | |  | | Міксін для забезпечення доступу до представлень лише авторизованим користувачам. |
| ForeignKey | Тип зв'язку один-до-багатьох між моделями: один об'єкт категорії може бути пов'язаний з багатьма постами. |
| Views (та пов’язані терміни) | |
| loginUser | Функція для аутентифікації користувача. Перевіряє ім’я користувача та пароль, і, якщо вони вірні, виконує вхід у систему. |
| logoutUser | Функція для виходу користувача з системи. Завершує сесію користувача та перенаправляє на сторінку входу. |
| registerUser | Функція для реєстрації нового користувача. Створює нового користувача через форму реєстрації та автоматично аутентифікує користувача після успішної реєстрації. |
| main  (обмежено LoginRequiredMixin) | Вид, який відображає головну сторінку для авторизованих користувачів. Включає категорії для фільтрації постів. |
| PostListMain  (обмежено LoginRequiredMixin) | Клас, який реалізує відображення списку постів із пагінацією та фільтрацією за пошуковим запитом. |
| CategoryListMain  (обмежено LoginRequiredMixin) | Клас, що відображає пости, що належать до вибраної категорії. Використовує пагінацію та контекст для категорій. |
| ShowPost  (обмежено LoginRequiredMixin) | Клас для відображення деталей поста. Включає методи для отримання поста за **slug** та додавання категорій до контексту. |
| AddPostView  (обмежено LoginRequiredMixin) | Клас для створення нового поста. Включає валідацію унікальності **slug** та збереження користувача як автора поста. |
| create\_post | Функція для створення нового поста. Включає форму для додавання посту та збереження в базі даних. |
| account | Функція для відображення акаунту користувача (видимий тільки самому користувачеві). Якщо профіль не знайдений, показується помилка. Також використовує пагінацію для постів. |
| profile | Функція для відображення профілю користувача (доступний для перегляду всіма користувачами). Використовує пагінацію для відображення постів користувача. |
| edit\_post | Функція для редагування посту користувача. Дозволяє змінювати дані поста, а також обробляти помилки. (доступна лише самому користувачеві із його акаунту) |
| delete\_post | |  | | --- | |  |  |  | | --- | | Функція для видалення посту користувача. Після видалення перенаправляє користувача на сторінку акаунту. (доступна лише самому користувачеві із його акаунту) | |
| UpdateProfile | |  | | --- | |  |  |  | | --- | | Функція для редагування профілю користувача. Оновлює дані профілю та дозволяє змінювати ім’я користувача. (доступна лише самому користувачеві із його акаунту) | |
| PostForm | |  | | --- | |  |  |  | | --- | | Форма для створення або редагування постів. Включає в себе валідацію та обробку даних поста. | |
| |  | | --- | | UpdateProfileForm |  |  | | --- | |  | | Форма для редагування профілю користувача. Дозволяє змінювати інформацію в профілі користувача. |
| Інші ключові поняття | |
| **Slug (або post\_slug)** | Частина URL, яка є унікальним ідентифікатором для поста і генерується користувачем при створенні поста. Зазвичай, це перетворена на маленькі літери і відокремлена дефісами назва поста або його заголовок, що дозволяє формувати читабельні та зручні для пошукових систем адреси. Наприклад, business\_manager чи markup\_developer. |
| **Paginator** | Об'єкт, який дозволяє реалізувати пагінацію (розбиття контенту на сторінки) в Django. Він використовується для розподілу великої кількості записів (наприклад, постів) на кілька сторінок, що покращує навігацію по сайту та оптимізує роботу з великими обсягами даних. |
| **Пагінація** | Процес розбиття великої кількості даних на сторінки. В Django для цього використовується клас **Paginator**. Він забезпечує поділ на сторінки і дозволяє зручно переміщатися між ними. Для кожного запиту, що має більше ніж одну сторінку, можна задати кількість елементів на сторінці і отримати відповідні дані. Пагінація зазвичай застосовується в представленнях, де відображаються списки постів, користувачів чи інших об'єктів.  Наприклад, у моєму проекті пагінація використовується в таких класах, як:   * PostListMain * CategoryListMain * ShowPost |