**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ**

**імені ІГОРЯ СІКОРСЬКОГО»**

факультет інформатики та обчислювальної техніки

(повна назва інституту/факультету)

кафедра інформаційних систем та технологій

(повна назва кафедри)

|  |  |
| --- | --- |
|  |  |

**Курсова робота**

з дисципліни «Програмування-2»

на тему: Сайт з подачі петицій

Виконав : студент \_\_1\_ курсу, групи \_\_\_\_ІС-31\_\_\_\_\_

(шифр групи)

Коваль Богдан Андрійович \_\_\_\_\_\_\_\_\_

(прізвище, ім’я, по батькові) (підпис)

Науковий керівник \_а. к. ІСТ Міщенко В.О.\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_

(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Члени комісії \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_

(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_

(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Засвідчую, що у цій курсовій роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент

Коваль Б.А. \_\_\_\_\_\_\_\_\_

(підпис)

Київ – 2024

ЗМІСТ

[ВСТУП 3](#_heading=h.1fob9te)

[1 ВИМОГИ ДО САЙТУ 4](#_heading=h.3znysh7)

[1.1 Функціональні вимоги до системи 4](#_heading=h.2et92p0)

[1.2 Нефункціональні вимоги до системи 4](#_heading=h.tyjcwt)

[2 СЦЕНАРІЇ ВИКОРИСТАННЯ САЙТУ 5](#_heading=h.3dy6vkm)

[2.1 Діаграма прецедентів 5](#_heading=h.1t3h5sf)

[2.2 Опис сценаріїв використання сайту 6](#_heading=h.4d34og8)

[3 АРХІТЕКТУРА СИСТЕМИ 10](#_heading=h.2s8eyo1)

[4 РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ 12](#_heading=h.17dp8vu)

[4.2 Компоненти рівня доступу до даних 14](#_heading=h.3rdcrjn)

[4.3 Компоненти рівня бізнес-логіки 18](#_heading=h.26in1rg)

[4.4 Компоненти рівня авторизації 22](#_heading=h.lnxbz9)

[4.5 Компоненти рівня інтерфейсу користувача 25](#_heading=h.35nkun2)

5 ВИСНОВКИ……………………………………………………………………..28  
 6 СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ……………………………………….30  
 ДОДАТКИ…………………………………………………………………………31

# ВСТУП

У сучасному світі стрімкого розвитку інформаційних технологій, роль Інтернету у формуванні громадської думки та впливі на політичні процеси є надзвичайно важливою. Громадяни все частіше використовують електронні платформи для висловлення своїх поглядів, обговорення суспільно важливих питань та ініціювання змін. Одним із найефективніших інструментів у цьому процесі є сайти подачі петицій. Вони надають можливість кожному громадянину безпосередньо звертатися до органів влади, організацій або компаній з конкретними вимогами чи пропозиціями.

Сайт подачі петицій являє собою веб-ресурс, який дозволяє користувачам створювати петиції, підписувати їх, а також поширювати інформацію про них серед ширшого кола осіб. Цей інструмент не лише сприяє активізації громадянської позиції, але й підвищує прозорість та підзвітність державних органів та інших інституцій. Основне завдання даної роботи є розробка базового функціоналу який буде складатися із наступних пунктах:

Реєстрація та авторизація користувачів:

* Створення облікового запису: Користувачі можуть зареєструватися, заповнивши форму, яка включає поля для введення логіну та пароля. Реєстрація забезпечує доступ до всіх функцій сайту та дозволяє користувачам створювати та підписувати петиції. Процес реєстрації є швидким і простим, щоб користувачі могли без зайвих зусиль почати користуватися сайтом.
* Авторизація: Вхід до системи здійснюється через введення логіну та пароля. Авторизація дозволяє користувачам зберігати свої дані та дії на сайті, включаючи підписані петиції та створені звернення. Це забезпечує персоналізований досвід використання сайту та дозволяє легко відслідковувати свої активності.

Створення петицій:

* Форма для створення: Інтерфейс із полями для введення заголовка та опису петиції. Заголовок має бути коротким і зрозумілим, щоб привернути увагу інших користувачів, а опис повинен детально пояснювати проблему або пропозицію, на яку спрямована петиція. Така структура дозволяє максимально ефективно доносити суть петиції до аудиторії.
* Функція видалення петиції: Якщо користувач є автором петиції, він має можливість видалити створену ним петицію. Це надає автору контроль над своїми публікаціями і дозволяє видаляти петиції у разі зміни обставин або за необхідності внесення коригувань. Видалення петиції є простим і зрозумілим процесом, доступним у будь-який час.

Підписання петицій:

* Інтерфейс підписання: Простий інтерфейс, де користувачі можуть підписати або прибрати підпис петиції одним натисканням кнопки. Це забезпечує легкість і доступність процесу підписання, що сприяє залученню більшої кількості підписантів. Користувачі можуть швидко і без зайвих зусиль виразити свою підтримку або відкликати її, якщо змінять свою думку.
* Відображення кількості підписів: Можливість побачити кількість зібраних підписів. Це дозволяє користувачам відстежувати прогрес петиції і бачити, наскільки велика підтримка отримана. Відображення кількості підписів також стимулює інших користувачів долучатися до підтримки популярних петицій, бачачи, що багато людей вже висловили свою думку.Актуальність створення такого сайту зумовлена потребою суспільства у нових формах взаємодії з владою та іншими важливими інституціями. В умовах цифрової ери громадяни прагнуть швидкого та ефективного способу комунікації, який дозволяє швидко реагувати на актуальні проблеми та долучатися до процесів прийняття рішень.

Метою даної курсової роботи є розробка проекту "Сайту подачі петицій", який буде відповідати сучасним вимогам користувачів, забезпечує високу ефективність та надійність, а також сприятиме підвищенню рівня громадської активності.

# 1 ВИМОГИ ДО САЙТУ

## Функціональні вимоги до системи

Розроблений сайт повинен включати наступний функціонал:

## Якщо користувач не зареєстрований в системі:

## Створення облікового запису з обов'язковим введенням логіну та пароля.

## Авторизація за допомогою введення коректних облікових даних.

## Можливість відвідати головну сторінку системи.

## Перегляд списку петицій.

## Можливість переглянути деталі петиції, включаючи заголовок, опис, автора та кількість підписів.

## Для зареєстрованих користувачів:

## Можливість вийти з особистого запису.

## Створення нової петиції з можливістю вводити текст та опис.

## Можливість видалити свою петицію, якщо користувач є її автором.

## Підписати або відкликати свій підпис з петиції іншого автора.1.2 Нефункціональні вимоги до системи

Нефункціональні вимоги до системи:

* Забезпечення відповідної швидкодії веб-сайту для зручного користування без зайвого очікування завантаження сторінок.
* Забезпечення зручного та інтуїтивно зрозумілого інтерфейсу користувача для полегшення навігації та взаємодії з системою.
* Забезпечення захисту конфіденційності користувачів та їхніх даних.

# 2 СЦЕНАРІЇ ВИКОРИСТАННЯ САЙТУ

## 2.1 Діаграма прецедентів

Діаграма прецедентів системи ілюструється на рисунку. 2.1.

У системі визначено два ключових типи користувачів: "Гість" (або незареєстрований користувач) та "Користувач" (який здійснив реєстрацію в системі). Зареєстрованим користувачам доступні всі функціональні можливості, як і для гостей.

Однак, вони мають додаткові привілеї, такі як можливість створювати, видаляти та віддавати свій голос за петиції. Детальніше кожен з цих сценаріїв буде розглянутий у наступному розділі, щоб краще зрозуміти функціонал та можливості цієї системи.

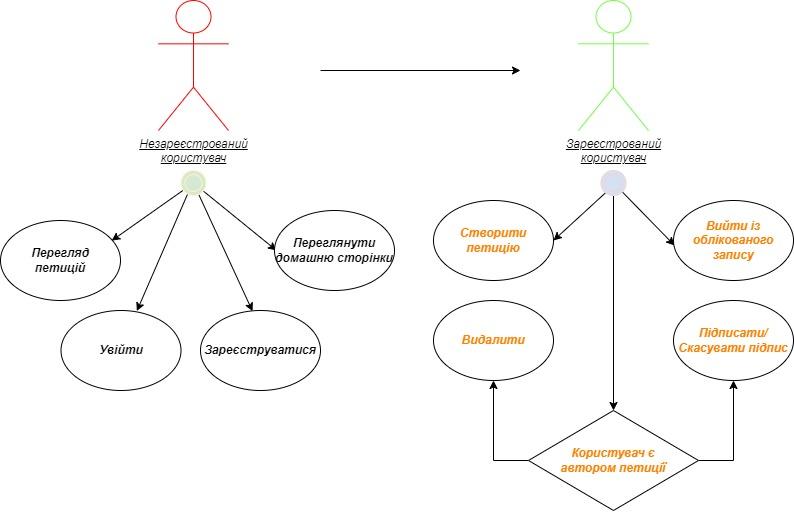


Рисунок 2.1 – Діаграма прецедентів

## 2.2 Опис сценаріїв використання сайту

Детальні описи сценаріїв використання наведено у таблицях 2.1 – 2.5

В таблиці 2.1 представлений сценарій використання «Перегляд головної сторінки»

Таблиця 2.1 – Сценарій використання «Перегляд сторінки головної сторінки»

|  |  |
| --- | --- |
| Назва | Перегляд сторінки головної сторінки |
| ID | 1 |
| Опис | Користувач може переглянути основну сторінку веб-сайту. Під час цього перегляду він може побачити заголовок та логотип, які ідентифікують сайт, а також навігаційне меню з посиланнями на різні розділи або сторінки сайту. |
| Актори | Користувач, Гість |
| Вигоди компанії | Головна сторінка містить загальну інформацію про сайт та його послуги, яка спрямована на те, щоб зацікавити користувача і надати йому зручний доступ до основного контенту. |
| Частота користування | Постійно |
| Тригери | Користувач натискає на відповідні кнопки на сторінці |
| Передумови | Користувач знаходиться на головній сторінці |
| Постумови | Користувач знаходиться на сторінці, на якій відображено перелік доступних петицій. |
| Основний розвиток | Користувач вибирає петицію та переходить на неї, клацнувши на відповідні елементи на сторінці. |
| Альтернативні розвитки | Користувач переходить на інші сторінки, натискаючи на відповідні кнопки у верхній частині веб-сайту (хедері). |
| Виняткові ситуації | – |

В таблиці 2.2 представлений сценарій використання «Перегляд сторінки петицій»

Таблиця 2.2 – Сценарій використання «Перегляд сторінки петицій»

|  |  |
| --- | --- |
| Назва | Перегляд сторінки петицій |
| ID | 2 |
| Опис | Користувач може відвідати сторінку "Перегляд сторінки петицій", щоб ознайомитися з актуальним списком петицій, які знаходяться на сайті. Переглядати заголовки таописи кожної петиції, а також побачити її автора. |
| Актори | Користувач, Гість |
| Вигоди компанії | Створення сторінки "Перегляд сторінки петицій" відкриває для користувачів можливість знаходити та висловлювати свою підтримку або незгоду з різноманітними ініціативами та проблемами, які є актуальними для них. Вона створює платформу для активного громадського участі, де люди можуть об'єднуватися для спільного досягнення мети та впливу на суспільні процеси. Ця сторінка стимулює взаємодію між різними групами та виявлення суспільних проблем |
| Частота користування | Постійно |
| Тригери | Користувач переходить на відповідні сторінки, клацнувши на відповідні кнопки, що знаходяться на екрані. |
| Передумови | Користувач перебуває на початковій сторінці. |
| Постумови | Користувач потрапляє на сторінку із переліком петицій |
| Основний розвиток | Користувач вибирає петицію та виконує відповідні дії, натискуючи на відповідні елементи на головній сторінці. Це призводить до переадресації користувача на сторінку самої петиції. |
| Альтернативні розвитки | Користувач виконує дії, натискуючи відповідні кнопки у верхній частині веб-сайту, що призводить до автоматичного переходу на сторінки, які відповідають цим елементам. |
| Виняткові ситуації | – |

Таблиця 2.3 – Сценарій використання «Створення петиції»

|  |  |
| --- | --- |
| Назва | Створення петиції |
| ID | 3 |
| Опис | Користувач ініціює процес створення петиції, заповнюючи необхідні поля з інформацією про петицію, таку як заголовок, опис. Після введення всіх даних вони натискають на кнопку "Публікувати", яка розміщена на відповідній сторінці. Після публікації петиція стає доступною для перегляду та підписування іншими користувачами веб-сайту. |
| Актори | Користувач |
| Вигоди компанії | Система подачі петицій, без відповідної сторінки для подачі петицій, втрачає свою цінність та ефективність. Мета сторінки полягає у тому, щоб забезпечити зручність та доступність процесу подачі петицій для користувачів. |
| Частота користування | Постійно |
| Тригери | Користувач активує опцію "Створити петицію", натиснувши на відповідну кнопку у верхній частині веб-сайту (хедері). |
| Передумови | Авторизація та створення петиції |
| Постумови | Публікація петиції |
| Основний розвиток | Після натискання на кнопку "Створити петицію", користувач вводить дані петиції. Після введення необхідної інформації, дані зберігаються на сервері для подальшої обробки. Петиція публікується та відображається на платформі, щоб інші користувачі могли знайти її та підписатися. |
| Альтернативні розвитки | \_ |
| Виняткові ситуації | – |

В таблиці 2.4 представлений сценарій використання «Перегляд сторінки конкретної петиції»

Таблиця 2.4 – Сценарій використання «Перегляд сторінки конкретної петиції»

|  |  |
| --- | --- |
| Назва | Перегляд сторінки конкретної петиції |
| ID | 4 |
| Опис | Користувач може переглянути дані конкретної петиції, такі як заголовок, кількість підписів та автор. Крім того, вони можуть виконати різні дії над петицією, такі як підписання (користувач), видалення (якщо користувач є автором). |
| Актори | Користувач, Гість |
| Вигоди компанії | Забезпечення легкості у створенні нових петицій може підтримати активність нашої користувальницької бази, привертаючи їхню увагу та зберігаючи інтерес до платформи. |
| Частота користування | Часто |
| Тригери | Користувач перейшов до конкретної петиції, перейшовши за посиланням або натиснувши на відповідне покликання. |
| Передумови | Користувач перебуває на сторінці, де відображається перелік різних петицій. |
| Постумови | Користувач знаходиться на сторінці конкретної петиції, де може ознайомитися з її вмістом та додати свій підпис або взяти участь у коментуванні. |
| Основний розвиток | У вікні з результатами пошуку користувач активує відповідні кнопки для висловлення свого голосу або видалення петиції. |
| Альтернативні розвитки | Користувач активує відповідні кнопки у верхній частині веб-сайту (хедері), що спричиняє переадресацію його на відповідні сторінки. |
| Виняткові ситуації | – |

В таблиці 2.5 представлений сценарій використання «Голосування за петицію та її видалення»

Таблиця 2.5 – Сценарій використання «Голосування за петицію та її видалення»

|  |  |
| --- | --- |
| Назва | Голосування за петицію та її видалення |
| ID | 5 |
| Опис | Користувач має можливість висловити свою підтримку або протест шляхом голосування за петицію, а також здійснити дію видалення у випадку, якщо він є автором цієї петиції. |
| Актори | Користувач |
| Вигоди компанії | Голосування є важливою функцією, що допомагає залучити та зацікавити наших поточних користувачів, створюючи основу для їхньої активності на платформі. |
| Частота користування | Часто |
| Тригери | Користувач активує відповідні елементи інтерфейсу. |
| Передумови | Користувач перебуває на веб-сторінці петиції та обирає бажану опцію. |
| Постумови | Петиція отримала новий голос. / Петиція була вилучена. |
| Основний розвиток | Користувач натискає на відповідні кнопки для голосування або видалення петиції |
| Альтернативні розвитки | Користувач активує відповідні елементи вверхній частині веб-сайту (хедері), що спричиняє його перенаправлення на відповідні сторінки. |
| Виняткові ситуації | \_ |

# 

# 3 АРХІТЕКТУРА СИСТЕМИ

Загальна архітектура системи наведена на рис. 3.1

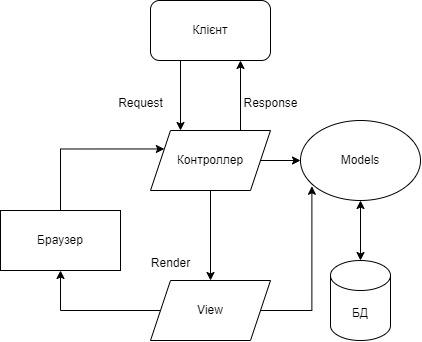


Рисунок 3.1 – Загальна архітектура системи

Система складається з трьох ключових компонентів:

* серверної частини
* бази даних
* графічного інтерфейсу

Графічний інтерфейс відповідає за забезпечення зручної взаємодії з користувачем. Він приймає введення від користувача та передає їх до серверної частини для подальшої обробки.

Серверна частина відповідає за обробку запитів, що надходять від графічного інтерфейсу. Тут реалізована основна логіка системи. Дані, які отримані з графічного інтерфейсу, перевіряються на відповідність та обробляються перед подальшою обробкою.

База даних виконує функцію збереження та надання даних, які були сформовані та оброблені на серверній частині. Вона забезпечує стабільний доступ до інформації для користувачів системи та графічного інтерфейсу.

Серверна частина системи включає наступні компоненти, які забезпечують його функціонування:

* Контролер: Відповідає за обробку вхідних запитів від користувача та взаємодію з іншими компонентами системи для виконання необхідних дій.
* Модель: Представляє собою структуру даних та бізнес-логіку системи. Модель забезпечує обробку та збереження інформації в базі даних.
* Вигляд: Відповідає за відображення даних та інтерфейсу користувача. Він перетворює дані з моделі в відповідність з графічним інтерфейсом користувача.
* Сервіс: Містить бізнес-логіку, яка виконується при взаємодії з іншими компонентами системи. Сервіси реалізують різноманітні операції та обробляють запити від контролера.
* Репозиторій: Відповідає за доступ до даних в базі даних. Репозиторій забезпечує спрощений та єдинообразний доступ до даних з різних частин системи, що використовуються для збереження та отримання інформації.Контролер отримує дані з графічного інтерфейсу. З контролера дані передаються в сервіс для виконання запиту до репозиторію. Репозиторій обробляє дані та взаємодіє з базою даних для збереження. Крім того, контролер формує вигляд, тобто об'єкт та його назву для відображення на графічному інтерфейсі.

# 4 РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ

4.1 Загальна структура проекту

Загальна структура проекту представлена на рисунках 4.1.1 та 4.1.2

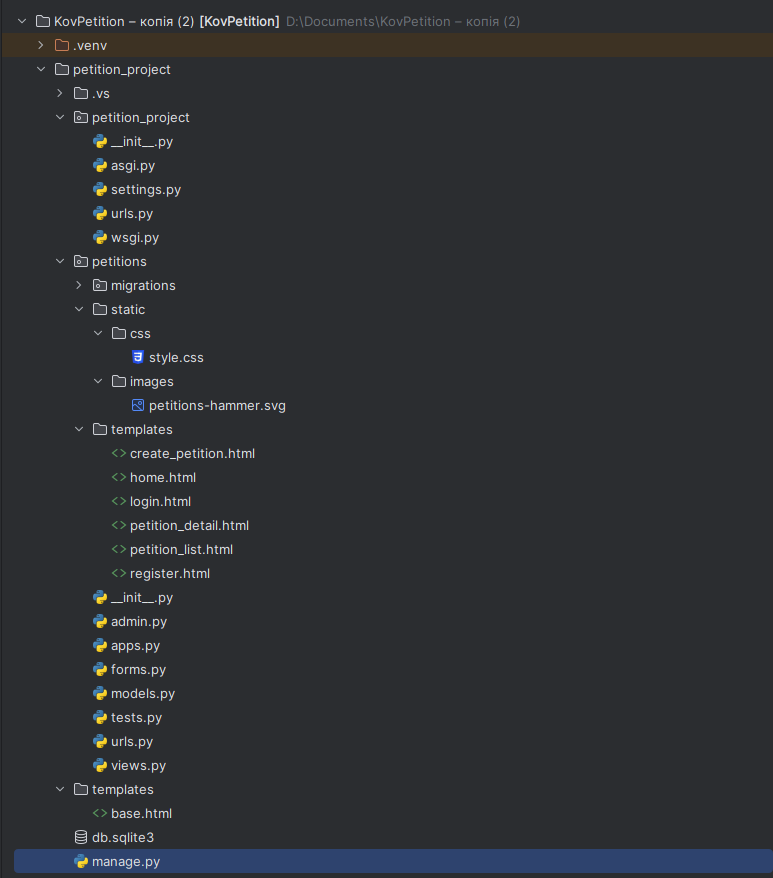


Рисунок 4.1.1 – Загальна структура проекту

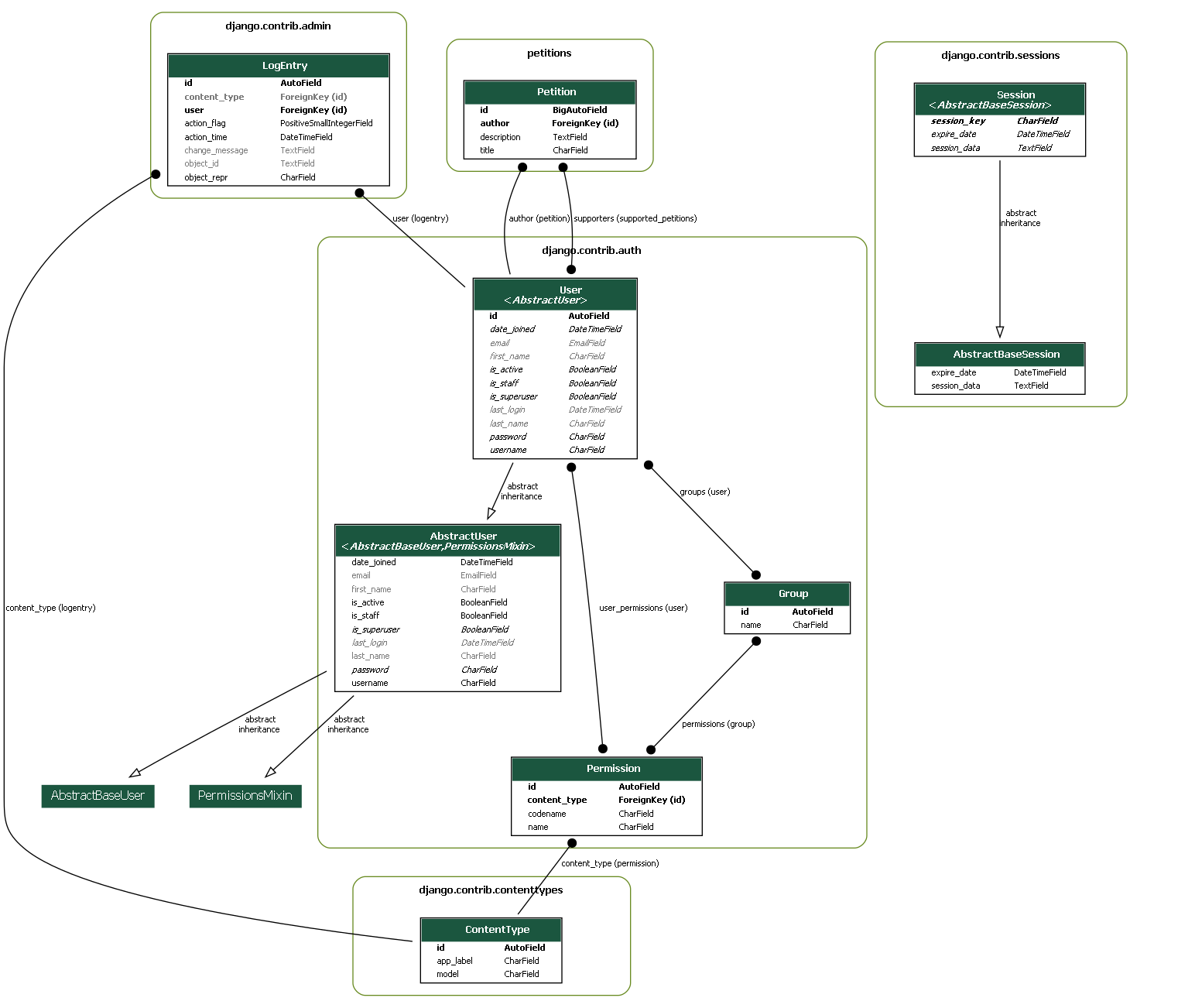


Рисунок 4.1.2 – Загальна структура проекту

Розглянемо загальну структу проєкта

**templates**: Ця тека містить HTML-шаблони вашого веб-додатку. Кожен шаблон відображає вміст сторінки та може містити дані, які передаються з представлення (views) для відображення користувачеві.

**static**: У цій теці зазвичай розміщуються статичні файли вашого веб-додатку, такі як CSS-стилі, JavaScript файли, зображення тощо. Ці файли не змінюються в залежності від запитів користувачів і можуть кешуватися браузером.

**migrations**: Тека міграцій містить файли, які відповідають за зміни в структурі бази даних вашого Django-проєкту. Кожна міграція описує зміни, які потрібно внести до бази даних для відповідності новій схемі.

**media**: У цій теці зазвичай зберігаються медіа-файли, які завантажуються користувачами, такі як зображення, відео тощо. Ці файли зазвичай пов'язані з моделями вашого Django-проєкту і завантажуються через форми.

**apps**: Деякі команди або розробники використовують цю теку для організації своїх додатків у Django-проєкті. Вона може містити підтеки для кожного окремого додатку, щоб легше управляти ними.

**settings**: Цей файл містить налаштування проєкту Django, такі як параметри бази даних, шляхи до статичних і медіа файлів, список встановлених додатків, налаштування безпеки та інші налаштування проєкту.

**urls**: У файлі *urls.py* визначаються URL-шаблони для вашого веб-додатку. Кожен URL-шаблон співставляється з відповідним представленням (view), яке обробляє відповідний HTTP-запит.

**wsgi**: Цей файл призначений для розгортання вашого Django-проєкту на веб-сервері, який підтримує *WSGI (Web Server Gateway Interface)*. Він ініціалізує об'єкт додатку, який використовується для обробки HTTP-запитів.

**asgi**: Цей файл потрібний для асинхронного розгортання Django-проєкту та використовується з асинхронними фреймворками через *ASGI (Asynchronous Server Gateway Interface)*.

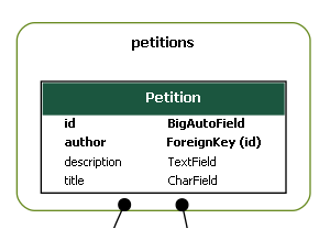
**models**: У цьому файлі описуються моделі даних вашого Django-проєкту. Кожна модель відображається в таблицю бази даних і визначається класом, який наслідується від класу Model

**admin**: Файл admin.py використовується для реєстрації моделей у панелі адміністратора Django. Це дозволяє вам керувати даними вашого сайту через адміністративний інтерфейс.

**forms**: У цьому файлі описуються форми, які використовуються у вашому Django-проєкті. Форми визначаються за допомогою класів, які наслідуються від класів Django Form.

**views**: Файл views.py містить представлення (views) вашого Django-проєкту. Представлення обробляють *HTTP-запити* та повертають відповіді, зазвичай через рендеринг шаблонів.

Далі детальний опис сутностей та інтерфейсів:



Основні класи та інтерфейси рівня інтерфейсу петиції наведені на рис. 4.2

Клас **Petition** представляє модель петиції і має наступні характеристики:

**title**: назва петиції (CharField(max\_length=200))

**description**: опис петиції (TextField())

**author**: автор петицій (User)

**supporters**: кількість голосів за петицію (User, related\_name='supported\_petitions')

Клас також має наступні методи:

**\_\_str\_\_():** повертає заголовок

**count\_supporters()**: повертає число підписів

## 4.1 Компоненти рівня авторизації

1. **register(request)** (реєстрація):

- Ця функція обробляє запити на реєстрацію нового користувача. Коли користувач відправляє форму реєстрації, вона отримує POST-запит.

- На початку перевіряється метод запиту. Якщо це POST-запит, тоді з форми, що надійшла, створюється екземпляр ***`UserCreationForm`***. Це форма, яка використовується для створення нового користувача в системі Django.

- Після отримання даних від користувача, форма перевіряється на валідність за допомогою методу ***`is\_valid()`***. Якщо форма є валідною, тобто всі необхідні дані введено правильно, то з цієї форми зберігається новий користувач за допомогою методу ***`save()`***. Це автоматично створює запис про нового користувача в базі даних.

- Після успішної реєстрації користувача відбувається автоматичний вхід в систему ***(`auth\_login`)***, після чого користувач перенаправляється на домашню сторінку.

- Якщо метод запиту - не ***POST*** (тобто ***GET***), то просто повертається пуста форма `UserCreationForm` для відображення на сторінці реєстрації.

2. **user\_login(request)** (вхід користувача):

- Ця функція обробляє запити на вхід користувача в систему. При отриманні ***POST***-запиту вона створює екземпляр форми ***`AuthenticationForm`***. Ця форма використовується для аутентифікації користувача в системі Django.

- Після отримання даних від користувача форма перевіряється на валідність. Якщо форма є валідною, отримується користувач за допомогою методу ***`get\_user()`***. Після чого користувач автоматично входить в систему ***(`auth\_login`).***

- Якщо метод запиту - не ***POST***, просто повертається пуста форма для входу користувача.

3. **user\_logout(request)** (вихід користувача):

- Ця функція обробляє запити на вихід користувача з системи. При виклику цієї функції користувач виходить з системи, і його сеанс знищується.

- Після виходу користувача відбувається автоматичне перенаправлення на домашню сторінку (***‘home’***).

## 

## 4.2 Компоненти рівня інтерфейсу користувача

Основні класи та інтерфейси рівня інтерфейсу користувача наведені на рис. 4.3

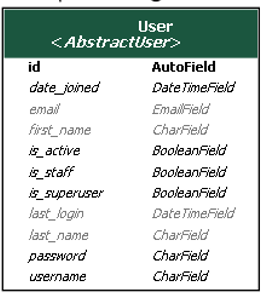


рис. 4.3

Рисунок 4.3 – Основні класи та інтерфейси рівня інтерфейсу користувача

Цей код включає функції для створення, перегляду, деталізації та видалення петицій.

1. **create\_petition(request)**:

- Ця функція обробляє запити на створення нової петиції. Якщо отриманий запит є POST-запитом, значить користувач намагається надіслати форму створення петиції.

- Форма *`PetitionForm`* створюється з даних, що надійшли в запиті.

- Якщо форма є валідною, то нова петиція зберігається у базі даних. Автором цієї петиції встановлюється поточний користувач.

- Після збереження петиції, користувач перенаправляється на домашню сторінку.

2. **petition\_list(request)**:

- Ця функція витягує усі петиції з бази даних та відображає їх у шаблоні *'petition\_list.html'*.

3. **petition\_detail(request, pk)**:

- Ця функція відображає деталі певної петиції. Вона отримує ідентифікатор петиції (pk) та витягує петицію з бази даних.

- Форма *`PetitionSignForm`* використовується для підписання на петицію.

- Якщо запит є POST-запитом, функція перевіряє, чи користувач намагається підписатися або відписатися від петиції. В залежності від цього змінюється стан підписки.

- На останок, деталі петиції та форма для підписання передаються у шаблон *'petition\_detail.html'*.

4. **delete\_petition(request, pk)**:

- Ця функція обробляє запити на видалення петиції. Якщо поточний користувач - автор петиції, то петиція видаляється з бази даних.

- Після видалення користувач перенаправляється на сторінку зі списком петицій.

5. **home(request)**

При отриманні запиту вона відображає шаблон *'home.html'*. Це зазвичай є основна сторінка додатку, яка може містити загальну інформацію, списки ресурсів або будь-який інший контент, який ви бажаєте показати користувачам при першому завантаженні додатку.

5 ВИСНОВКИ

Завершуючи написання курсової роботи на тему створення веб-платформи для подачі петицій, хочу поділитися з вами результатами роботи та відзначити важливі аспекти, які вплинули на процес розробки та функціонування системи. У цьому висновку підкреслимо ключові технології, які використовувалися під час розробки, а також загальну мету та значення проекту.

Розроблена система має важливе суспільне значення, сприяючи громадянському активізму та демократії. Платформа для подачі петицій створює можливість для громадян висловлювати свої погляди та долучатися до вирішення важливих суспільних питань. Враховуючи сучасні тенденції та потреби суспільства, проект спрямований на підвищення ефективності громадського впливу та підтримки громадянського суспільства.

У ході реалізації даної курсової роботи використовувалися передові технології веб-розробки, такі як Django, HTML та Jinja. Django, як відомий фреймворк Python, відіграв вирішальну роль у створенні серверної частини нашого веб-додатка. Цей фреймворк дозволяє швидко розробляти потужні та масштабовані веб-додатки за допомогою зручної системи маршрутизації, вбудованих інструментів аутентифікації та управління базами даних.

HTML використовувався для розробки клієнтської частини веб-додатка, а Jinja — для забезпечення динамічності та генерації веб-сторінок на базі шаблонів. Завдяки цим технологіям ми змогли створити зручний та естетичний користувацький інтерфейс, який дозволяє користувачам легко навігуватися та взаємодіяти з платформою подачі петицій.

Додатковою перевагою використання Django, HTML та Jinja є їхня велика спільнота та наявність багатофункціональних плагінів та інструментів, що полегшують процес розробки та підтримки веб-додатків.

Була використана база даних SQLite, яка відома своєю простотою в установці та використанні, а також легкістю налаштування. Ця локальна база даних ідеально підходить для невеликих проектів та прототипів, забезпечуючи достатній рівень продуктивності та надійності.

Використання ORM через методи Django дозволило нам зручно взаємодіяти з базою даних SQLite та забезпечити гнучкість у роботі з даними, зберігаючи при цьому простоту та зрозумілість коду.

Також варто відзначити, що ER-модель бази даних була ретельно спроектована, що спростило взаємодію між різними складовими системи та забезпечило її стабільну та ефективну роботу.

Отже, використання цих передових технологій у поєднанні з нашим розумінням потреб користувачів та правових аспектів дозволило створити не лише функціональний, але й ефективний та безпечний веб-додаток для подачі петицій, який може відігравати важливу роль у громадянському активізмі та сприяти вирішенню актуальних суспільних питань.

6 ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Django.fun [Електронний ресурс] – Режим доступу до ресурсу: - <https://django.fun/>
2. Google Fonts [Електронний ресурс] – Режим доступу до ресурсу: - <https://fonts.google.com/>
3. Djinja doc [Електронний ресурс] – Режим доступу до ресурсу: - <https://jinja.palletsprojects.com/en/3.1.x/>
4. Draw.io [Електронний ресурс] – Режим доступу до ресурсу: - https://app.diagrams.net/
5. Python doc [Електронний ресурс] – Режим доступу до ресурсу: - https://docs.python.org/3/
6. CSS doc [Електронний ресурс] – Режим доступу до ресурсу: - <https://developer.mozilla.org/en-US/docs/Web/CSS>
7. W3C [Електронний ресурс] – Режим доступу до ресурсу: - https://www.w3.org/2000/svg

7 ДОДАТКИ

**View**

**from django.shortcuts import render, redirect, get\_object\_or\_404**

**from django.contrib.auth.forms import UserCreationForm, AuthenticationForm**

**from django.contrib.auth import login as auth\_login, logout as auth\_logout**

**from django.contrib.auth.decorators import login\_required**

**from .forms import PetitionForm, PetitionSignForm**

**from .models import Petition**

**def home(request):**

**return render(request, 'home.html')**

**def register(request):**

**if request.method == 'POST':**

**form = UserCreationForm(request.POST)**

**if form.is\_valid():**

**user = form.save()**

**auth\_login(request, user)**

**return redirect('home')**

**else:**

**form = UserCreationForm()**

**return render(request, 'register.html', {'form': form})**

**def user\_login(request):**

**if request.method == 'POST':**

**form = AuthenticationForm(data=request.POST)**

**if form.is\_valid():**

**user = form.get\_user()**

**auth\_login(request, user)**

**return redirect('home')**

**else:**

**form = AuthenticationForm()**

**return render(request, 'login.html', {'form': form})**

**@login\_required**

**def user\_logout(request):**

**auth\_logout(request)**

**return redirect('home')**

**def create\_petition(request):**

**if request.method == 'POST':**

**form = PetitionForm(request.POST)**

**if form.is\_valid():**

**petition = form.save(commit=False)**

**petition.author = request.user**

**petition.save()**

**return redirect('home')**

**else:**

**form = PetitionForm()**

**return render(request, 'create\_petition.html', {'form': form})**

**def petition\_list(request):**

**petitions = Petition.objects.all()**

**return render(request, 'petition\_list.html', {'petitions': petitions})**

**def petition\_detail(request, pk):**

**petition = get\_object\_or\_404(Petition, pk=pk)**

**form = PetitionSignForm()**

**subscribed = petition.supporters.filter(**

**pk=request.user.pk).exists() # Перевірка, чи користувач підписаний на петицію**

**if request.method == 'POST':**

**if 'subscribe' in request.POST:**

**petition.supporters.add(request.user)**

**elif 'unsubscribe' in request.POST:**

**petition.supporters.remove(request.user)**

**return redirect('petition\_detail', pk=pk)**

**return render(request, 'petition\_detail.html', {'petition': petition, 'form': form, 'subscribed': subscribed})**

**def delete\_petition(request, pk):**

**petition = get\_object\_or\_404(Petition, pk=pk)**

**if request.user == petition.author:**

**petition.delete()**

**return redirect('petition\_list')**

**Urls**

**from django.urls import path**

**from . import views**

**urlpatterns = [**

**path('', views.home, name='home'),**

**path('register/', views.register, name='register'),**

**path('login/', views.user\_login, name='login'),**

**path('logout/', views.user\_logout, name='logout'),**

**path('create/', views.create\_petition, name='create\_petition'),**

**path('petitions/', views.petition\_list, name='petition\_list'),**

**path('petition/<int:pk>/', views.petition\_detail, name='petition\_detail'),**

**path('petition/<int:pk>/delete/', views.delete\_petition, name='delete\_petition'),**

**]**

**Models**

**from django.db import models**

**from django.contrib.auth.models import User**

**class Petition(models.Model):**

**title = models.CharField(max\_length=200)**

**description = models.TextField()**

**author = models.ForeignKey(User, on\_delete=models.CASCADE)**

**supporters = models.ManyToManyField(User, related\_name='supported\_petitions', blank=True)**

**def \_\_str\_\_(self):**

**return self.title**

**def count\_supporters(self):**

**return self.supporters.count()**

**forms**

**from django import forms**

**from .models import Petition**

**class PetitionForm(forms.ModelForm):**

**class Meta:**

**model = Petition**

**fields = ['title', 'description']**

**Templates:  
base.html**

**{% load static %}**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

**<meta name="viewport" content="width=device-width, initial-scale=1.0">**

**<meta name="author" content="Koval B."/>**

**<meta name="description" content="coursework website of petitions" />**

**<title>{% block title %}KovPetition{% endblock %}</title>**

**<link rel="stylesheet" href="{% static 'css/style.css' %}">**

**<!-- Montserrat, Nunito, Raleway -->**

**<link rel="preconnect" href="https://fonts.googleapis.com">**

**<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>**

**<link**

**href="https://fonts.googleapis.com/css2?family=Montserrat:wght@400;600;700&family=Nunito:wght@400;700&family=Open+Sans:wght@300;400&family=Raleway:wght@700;800&display=swap"**

**rel="stylesheet">**

**<!-- Birthstone -->**

**<link rel="preconnect" href="https://fonts.googleapis.com">**

**<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>**

**<link href="https://fonts.googleapis.com/css2?family=Birthstone&display=swap" rel="stylesheet">**

**<!-- Mohave -->**

**<link rel="preconnect" href="https://fonts.googleapis.com">**

**<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>**

**<link href="https://fonts.googleapis.com/css2?family=Mohave:wght@500&display=swap" rel="stylesheet">**

**</head>**

**<body>**

**<header class="header">**

**<div class="header\_\_container">**

**<div class="header\_\_inner">**

**<div class="header\_\_top">**

**<a class="logo\_\_link" href="#">KovPetition <img class="logo\_\_link--img"**

**src="{% static 'images/petitions-hammer.svg' %}" alt=""></a>**

**<nav class="header\_\_menu">**

**<ul class="header\_\_list">**

**<li class="header\_\_item"><a class="header\_\_link" href="{% url 'home' %}">Головна</a></li>**

**<li class="header\_\_item"><a class="header\_\_link" href="{% url 'petition\_list' %}">Переглянути петиції</a></li>**

**{% if user.is\_authenticated %}**

**<li class="header\_\_item"><a class="header\_\_link" href="{% url 'create\_petition' %}">Створити петицію</a>**

**</li>**

**<li class="header\_\_item"><span>Вітаю, {{ user.username }}!</span> <a**

**class="header\_\_link"**

**href="{% url 'logout' %}">Вийти</a></li>**

**{% else %}**

**<li class="header\_\_item"><a class="header\_\_link" href="{% url 'login' %}">Вхід</a></li>**

**<li class="header\_\_item"><a class="header\_\_link" href="{% url 'register' %}">Реєстрація</a></li>**

**{% endif %}**

**</ul>**

**</nav>**

**</div>**

**<div class="header\_\_content--inner ring">**

**<div class="ring-items">**

**<i style="--clr:#00ff0a;"></i>**

**<i style="--clr:#ff0057;"></i>**

**<i style="--clr:#fffd44;"></i>**

**</div>**

**<div class="header\_\_content">**

**{% block content %}**

**{% endblock %}**

**</div>**

**</div>**

**<div class="footer">**

**<div class="footer">**

**<p class="footer\_\_autor--text">**

**Курсова робота студента ІС-31 Коваля Богдана**

**</p>**

**</div>**

**</div>**

**</div>**

**</div>**

**</header>**

**</body>**

**</html>**

**home.html**

**{% extends 'base.html' %}**

**{% block content %}**

**<h1 class="header\_\_title">Вітаю, на сайт подачі петицій</h1>**

**{% endblock %}**

**login.html**

**{% extends 'base.html' %}**

**{% block title %}Вхід{% endblock %}**

**{% block content %}**

**<h2 class="content\_\_title">Вхід</h2>**

**<form class="form" method="post">**

**{% csrf\_token %}**

**{{ form.as\_p }}**

**<button class="form\_\_btn-green" type="submit">Ворватися</button>**

**</form>**

**{% endblock %}**

**register.html**

**{% extends 'base.html' %}**

**{% block content %}**

**<h2 class="content\_\_title">Реєстрація</h2>**

**<form class="form" method="post">**

**{% csrf\_token %}**

**{{ form.as\_p }}**

**<button class="form\_\_btn-green" type="submit">В тему</button>**

**</form>**

**{% endblock %}**

**petition\_list.html**

**{% extends 'base.html' %}**

**{% block title %}Список петицій{% endblock %}**

**{% block content %}**

**<h2 class="content\_\_title">Список петицій</h2>**

**<ol class="petition\_\_list">**

**{% for petition in petitions %}**

**<li class="petition\_\_list--item"><a class="petition\_\_list--link"**

**href="{% url 'petition\_detail' petition.id %}">{{ petition.title }}</a> - автор**

**{{ petition.author }}</li>**

**{% endfor %}**

**</ol>**

**{% endblock %}**

**petition\_detail.html**

**{% extends 'base.html' %}**

**{% block title %}{{ petition.title }}{% endblock %}**

**{% block content %}**

**<h2>Дедальніше про петицію:</h2>**

**<h3>{{ petition.title }}</h3>**

**<p>{{ petition.description }}</p>**

**<p>Автор: {{ petition.author }}</p>**

**<p>Кількість підписів: {{ petition.count\_supporters }}</p>**

**{% if user.is\_authenticated %}**

**{% if user != petition.author %}**

**<form class="form" method="post">**

**{% csrf\_token %}**

**{% if subscribed %}**

**<button class="form\_\_btn-red" type="submit" name="unsubscribe">Відписатися</button>**

**{% else %}**

**<button class="form\_\_btn-green" type="submit" name="subscribe">Підписатися</button>**

**{% endif %}**

**</form>**

**{% endif %}**

**{% if user == petition.author %}**

**<form class="form" method="post" action="{% url 'delete\_petition' petition.id %}">**

**{% csrf\_token %}**

**<button class="form\_\_btn-red" type="submit">Видалити петицію</button>**

**</form>**

**{% endif %}**

**{% else %}**

**<a href="{% url 'login' %}">Увійдіть, щоб підписатися</a>**

**{% endif %}**

**{% endblock %}**

**create\_petition.html**

**{% extends 'base.html' %}**

**{% block title %}Create Petition{% endblock %}**

**{% block content %}**

**<h2 class="content\_\_title">Створити петицію</h2>**

**<form class="form" method="post">**

**{% csrf\_token %}**

**{{ form.as\_p }}**

**<button class="form\_\_btn-green" type="submit">Додати</button>**

**</form>**

**{% endblock %}**

**Static:**

\*,  
\*::after,  
\*::before {  
 box-sizing: border-box;  
}  
  
a {  
 text-decoration: none;  
 color: inherit;  
}  
  
ul {  
 list-style: none;  
}  
  
button {  
 transition: all .5s;  
}  
  
.header\_\_container {  
 max-width: 1350px;  
 padding: 0 10px;  
 margin: 0 auto;  
}  
  
body {  
 font-family: 'Open Sans', sans-serif;  
 font-weight: 400;  
 font-size: 15px;  
 line-height: 138%;  
 color: #050505;  
 background-color: gray;  
}  
  
.header\_\_title {  
 font-weight: 700;  
 font-size: 32px;  
 line-height: 1.2;  
 margin-bottom: 15px;  
}  
  
.header\_\_content--inner {  
 font-family: Raleway, sans-serif;  
 color: #ffffff;  
 /\* margin-bottom: 120px; \*/  
 background-color: rgba(0, 0, 0, .4);  
 border-radius: 35px;  
 display: flex;  
 flex-direction: column;  
 justify-content: center;  
 padding: 15px;  
 width: 700px;  
}  
  
.header\_\_content {  
 z-index: 5;  
 margin: 0 35px;  
}  
  
.header\_\_title::after {  
 content: "Kyiv - GMT+2, online 8:00-22:00";  
 position: absolute;  
 bottom: 0;  
 opacity: 0;  
 transition: all .5s;  
 color: #dddd2c;  
}  
  
  
.header\_\_inner {  
 height: 100vh;  
 display: flex;  
 flex-direction: column;  
 justify-content: space-between;  
}  
  
.header\_\_top {  
 display: flex;  
 justify-content: space-between;  
 align-items: center;  
 /\* margin-bottom: 200px; \*/  
 padding-top: 70px;   
 flex-wrap: wrap;  
}  
  
.logo\_\_link {  
 position: relative;  
 font-family: 'Birthstone', cursive;  
 font-size: 65px;  
 font-weight: 400;  
 color: #ffffff;  
 line-height: 1.2;  
}  
  
.logo\_\_link {  
 font-family: 'Birthstone', cursive;  
 font-size: 65px;  
 font-weight: 400;  
 color: #ffffff;  
 line-height: 1.2;  
}  
  
.logo\_\_link--img {  
   
 width: 35px;  
 height: 35px;  
 position: absolute;  
 bottom: 0;  
 top: 50%;  
 transform: translateY(-35%);  
 right: -35px;  
}  
  
  
.header\_\_list {  
 display: flex;  
}  
  
.header\_\_link {  
 font-family: Montserrat, sans-serif;  
 font-size: 17px;  
 line-height: 21px;  
 font-weight: 700;  
 color: #fff;  
 margin-left: 50px;  
}  
  
.header\_\_item span{  
 font-family: Montserrat, sans-serif;  
 font-size: 17px;  
 line-height: 21px;  
 font-weight: 700;  
 color: #fff;  
 margin-left: 50px;  
}  
  
.header\_\_item:last-child .header\_\_link{  
 background-color: #fffd44;  
 color: inherit;  
 text-align: center;  
 border-radius: 30px;  
 padding: 17px 21px;  
 position: relative;  
}  
  
  
.header\_\_item:last-child .header\_\_link::after {  
 content: '';  
 position: absolute;  
 border-radius: 30px;  
 left: 0;  
 top: 0;  
 width: 100%;  
 height: 100%;  
 transition: all .5s;  
}  
  
.header\_\_item:last-child .header\_\_link:hover::after {  
 background-color: rgba(0, 0, 0, .25);  
 border: 3px solid #fffd44;  
}  
  
.ring {  
 position: relative;  
 width: 600px;  
 height: 500px;  
 display: flex;  
 justify-content: center;  
 align-items: center;  
}  
  
.ring-items {  
 z-index: 1;  
}  
  
.ring i {  
 position: absolute;  
 inset: 0;  
 border: 2px solid #fff;  
 transition: 0.5s;  
}  
.ring i:nth-child(1) {  
 border-radius: 38% 62% 63% 37% / 41% 44% 56% 59%;  
 animation: animate 6s linear infinite;  
}  
.ring i:nth-child(2) {  
 border-radius: 41% 44% 56% 59%/38% 62% 63% 37%;  
 animation: animate 4s linear infinite;  
}  
.ring i:nth-child(3) {  
 border-radius: 41% 44% 56% 59%/38% 62% 63% 37%;  
 animation: animate2 10s linear infinite;  
}  
.ring:hover i {  
 border: 6px solid var(--clr);  
 filter: drop-shadow(0 0 20px var(--clr));  
}  
@keyframes animate {  
 0% {  
 transform: rotate(0deg);  
 }  
 100% {  
 transform: rotate(360deg);  
 }  
}  
@keyframes animate2 {  
 0% {  
 transform: rotate(360deg);  
 }  
 100% {  
 transform: rotate(0deg);  
 }  
}  
  
  
.content\_\_title {  
 font-family: Montserrat, sans-serif;  
 font-weight: 700;  
 font-size: 30px;  
 line-height: 37px;  
 margin-bottom: 25px;  
}  
  
.petition\_\_list--item {  
 padding-bottom: 25px;  
}  
  
.petition\_\_list--link {  
 font-size: 25px;  
}  
  
.form {  
 max-width: 300px;  
}  
  
.form li{  
 display: none;  
}  
  
  
.form input, textarea {  
 font-family: Montserrat, sans-serif;  
 font-size: 14px;  
 line-height: 17px;  
 font-weight: 700;  
 color: #fffd44;  
 display: block;  
 width: 100%;  
 border: none;  
 background-color: inherit;  
 height: 35px;  
 border-bottom: 3px solid #000;  
 border-left: 3px solid #000;  
 padding: 0 15px;  
}  
  
.form textarea {  
 height: 150px;  
 padding-top: 15px;  
}  
  
.errorlist.nonfield {  
 width: 300px;  
 padding: 0 15px;  
 color: #ff0057;  
}  
  
.form button {  
 margin: 0 auto;  
 border: none;  
 background-color: #fff;  
 font-family: Montserrat, sans-serif;  
 font-weight: 700;  
 font-size: 16px;  
 line-height: 20px;  
 cursor: pointer;  
 text-align: center;  
 border-radius: 5px;  
 padding: 5px 8px;  
  
}  
  
.form\_\_btn-red:hover {  
 background-color: #ff0057;  
}  
  
  
.form\_\_btn-green:hover {  
 background-color: #00ff0a;  
}  
  
  
.form button:hover.form\_\_delete-btn:hover {  
 background-color: #ff0057;  
}  
  
  
.footer {  
 text-align: right;  
 position: relative;  
 font-family: Raleway, sans-serif;  
 font-size: 18px;  
 font-weight: 400;  
 color: #ffffff;  
 line-height: 1.2;  
}