МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

КАФЕДРА КОМП'ЮТЕРНИХ НАУК

ВИПУСКНА РОБОТА

на тему:

«Модуль реєстрації користувачів Web-додатку мовою програмування Python»

Завідувач
випускаючої кафедри **Довбиш А.С.**

Керівник роботи **Шелехов І.В.**

Студента групи ІН – 63 **Цись М.В.**

СУМИ 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

КАФЕДРА КОМП'ЮТЕРНИХ НАУК

Затверджую_____

Зав. кафедри Довбиш А.С.

"_____" _____2020 р.

ЗАВДАННЯ

до випускної роботи

Студента четвертого курсу, групи ІН-63 спеціальності " Комп'ютерні науки " денної форми навчання Цись Максима Вікторовича.

Тема: **"**Модуль реєстрації користувачів Web-додатку мовою програмування Python**"**

Затверджена наказом по СумДУ

№_____ от _____2020 р.

Зміст пояснювальної записки: 1) Формування MVC моделі модуля; 2) Вибір фреймворку для реалізації моделі; 3) Вибір програмного забезпечення для реалізації окремих компонентів сформованої моделі; 4) Розробка бази даних; 5) Розробка графічного дизайну модуля; 6) Реалізація MVC моделі модуля; 7) Тестування модуля.

Дата видачі завдання "_____" _____2020 р.

Керівник випускної роботи _____ Шелехов І.В.

Завдання прийняв до виконання _____Цись М.В.

РЕФЕРАТ

Записка:. 46 стор., 8 рис., 3 табл., 1 додаток, 10 джерел.

Об’єкт дослідження — процес проектування веб-орієнтованих інформаційних систем і їх компонентів.

Мета роботи — розробка модуля реєстрації користувачів мовою програмування Python на фреймворку Django.

Методи дослідження — методи проектування інформаційних систем, методи створення і оптимізації структури баз даних**.**

Результати — створено Web-додаток, на якому користувач може зареєструватися та увійти у інформаційну веб-орієнтовану систему. Додаток складається з серверної та клієнтської частини, оперує базою даних з чотирбох таблиць. Додаток був реалізований на мові Python на популярному фреймворку Django.

МОДУЛЬ РЕЄСТРАЦІЇ КОРИСТУВАЧІВБ, WEB-ДОДАТОК, PYTHON

ЗМІСТ

ВСТУП

Django – це розроблений на Python, легкий і відкритий фреймворк для розробки веб-додатків. Фреймворк - це набір модулів, що спрощують розробку. Коли, вони згруповані разом, то допомагають розробляти додатки та веб-сайти не з пустого місця, а з вихідних кодів, які вже є.

Найпростіші веб-сайти, із-за цього, можуть мати в собі широкі можливості, наприклад, підтримка автентифікації, коментарі, контактні форми, панелі управління та адміністрування, завантаження файлів, і таке інше. По-іншому, коли ви будуєте сайт з початку, то потрібно робити ці компоненти самому. Коли робите це з фреймворком, то компоненти є вже вбудованими, і вам тільки треба буде налаштувати компоненти так, щоб були адаптовані для вашого сайту.

Офіційний сайт ідеї демонструє Django, що це «Високорівневий Python веб-фреймворк, що робе розробку швидкою, чистий і простий дизайн. Зроблений професійними розробниками, він полегшує складнощі, які пов’язані з веб-розробкою, і тому у вас є можливість фокусуватися на розробці свого додатку і без складнощів винаходу велосипеду. Він нічого не коштує і з доступним вихідним кодом.»

У Django є великий набір модулів, які ми можемо брати у свої проекти. По-перше, фреймворки створені для збереження часу, що витрачається розробниками і збереження їх від супутніх складнощів, з якими вони стикаються в процесі роботи. Django має ту ж філософію.

Важливо, коли Django проектувався, то з думками про front-end розробників. “Проект Django розроблений, щоб стати доступною, легкою і комфортною в навчанні тих, кому подобається працювати з front-end розробників та дизайнерів HTML. Мова шаблонів Django також є гнучким і розширеним, дозволяючи розробникам розширяти мову стандартів відповідно з їхніми потребами”. Коли ви захочете порацювати з Python, якщо це для веб-додатків та веб-дизайну, вам просто прийдеться пам’ятати за Django фреймворк. І він вам, неодмінно допоможе.

1 АНАЛІТИЧНИЙ ОГЛЯД

1.1 Аналіз сучасних Back-End фреймворків

Фреймворк — це платформа, що дає користувачам основу, за для розробки ПЗ. Він має в собі заздалегідь реалізовані і визначені функції або класи. Також, для точних задач можна додати власноруч написаний код до того, що вже є у фреймворку.

Плюси:

- Рентабельність
- Велика кількість фреймворків – мають доступний вихідний код та безкоштовні для використання. Це дуже полегшує створення додатку і ціна самого веб-додатку зменшується, відповідно.
- Продуктивність
- Фреймворки покращують ефективність, завдяки тому, що всі процеси максимально оптимізовані. Набагато простіше працювати з фреймворк з структурованими шаблонами і оптимізацією ніж писати коди з сотнями рядків.
- Безпека

Перевага JavaScript-фреймворка це система безпеки та підтримка спільноти на GitHub.

Yii – це загальний фреймворк, що може застосовуватися до всіх типів веб додатків. Тому, що ця структура компонентна та її підтримка кешування відмінна, особливо фреймворк підходить для створення таких ідей, на кшталт портали, магазини, форуми, CMS або RESTful-додатки.

Перспективи Yii:

- Малий поріг входження
- Шаблон Model View Controller
- Інтерфейси ActiveRecord та DAO для ведення із базами даних (PDO)
- Сприяння інтернаціоналізації
- Кешування сторінок та деяких фрагментів

- Виявлення та виправлення помилок
- Валідація та введення форм
- Авторизація та аутентифікація
- Робота з AJAX та інтеграція з jQuery
- Генерація основного PHP-коду для CRUD-операцій
- Сприяння тем, та оформлення задля їх швидкої зміни
- Змога застосування різних бібліотек
- Міграції БД
- Машинальне тестування
- Допомога REST

Мінуси:

- AR не підтримує AR запити
- Не ефективний взаємозв'язок в БД плагін CAdvancedArBehavior коригує недоліки

Symfony2 – PHP фреймворк, що має велику кількість бібліотек класів, зроблений на PHP 5. В архітектурі є важливі компоненти та засоби, які потрібні, щоб створювати складні веб-додатки. Symfony — це вільна основа, зроблена на PHP5, яка застосовує патерн Model-View-Controller (MVC). Symfony рекомендує стрімку розробку та керування веб-додатками, вона допомагає швидко вирішити буденні задачі веб-програміста. Symfony вільний та досяжний під ліцензією MIT.

Плюси:

•Піддержує багато баз даних (PostgreSQL, MySQL, SQLite, та інші PDO-одночасна СУБД)

- Вбудовані класи, щоб працювати з email
- Змінна система зразків в подані
- Вбудований кодогенератор
- Допомога спонсора Sensio
- Дуже гнучкий

- Висока результативність

Мінуси:

- Важкий в навчанні
- Рекомендовано тільки для великих проектів
- Версія 5.3 запрошує PHP
- Відсутня вбудована ORM

Zend framework – це PHP– фреймворк, що зроблений та солідаризується підприємством Zend, робітники якого самі є винахідниками мови PHP. І цьому він є наслідувачем традиції та цінностей PHP – спирається на прості, об'єктно-орієнтовані принципи, дружні ліцензії та вживанням Agile методів.

Переваги:

- Усі складові загалом орієнтовані на PHP 5 та E_STRICT - загальні
- Вбудовани1 генератор коду
- Структура для використання тільки того, що потрібно, з мінімумом залежності компонента
- Використовує шаблон проектування MVC, що легко розширюється, сприяє підтримці макетів і PHP-скриптів з подачею за замовчуванням
- Піддержує багато різноманітних баз даних, включаючи MariaDB, Oracle, MySQL, Microsoft SQL Server, PostgreSQL, IBM DB2, Informix Dynamic Server and SQLite
- Особливі системи для роботи, відправлення, приймання email за підтримки mbox, IMAP4, Maildir та POP3
- Порядок кешування гнучкий із допомогою багатьох різних сховищ

Мінуси:

- Великий за розміром
- Нешвидкий без додаткового кешування
- Структура дуже складна, складний в навчанні, якщо не розуміти шаблони проектування
- Розвивається занадто повільно

- Остання версія потребує PHP 5.2
- ORM відсутнє

CakePHP являється фреймворком до PHP, він пропонує дуже широку архітектуру для проектування, розробки та обслуговування web-додатків. Він пропонує використовувати відому систему проектування MVC, як і в об'єктно-реляційних відомих фреймворків. Ідея, що є основною, CakePHP, є в нарощуванні результативності проекту, і, як наслідок, полегшення для спеціаліста у зменшенні обсягу роботи над кодом. На початку він проектувався, озираючись на популярний Ruby on Rails, і тому деякі ідеї схожі.

Перспективи:

- Може бути сумісним з PHP5 та PHP4 (включно до 1.3версії)
- Диспетчер URL, що застосовує регулярні обчислення
- При дотриманні правил назви стовпців, генерація коду за схемою БД
- Тестування форм
- Обмежований доступ (ACL), контроль над сесіями, cookies, складники для авторизації, додавання деревоподібних даних (Nested Sets)
- Хелпери (складники) для наповнення та сукупності різних форм, керування кешем, розподіл на сторінки (paginate), JavaScript(а також і AJAX)
- Засоби інтернаціоналізації
- Сукупність SQL-питань, також таблиць з зв’язками ORM
- Scaffolding та сукупність CRUD сторінок для сутностей, Router:mapResources з Put Delete Get Post.
- Є автогенератор коду Bake
- Допоміжна програма Simple Test
- Консольна інтеграція, рівень Shell і задача Task
- Як окремі програми, плагіни, складові
- Міграції

Мінуси:

- Мала результативність
- Мала підтримка документації
- Слабкий до атак CSRF
- ACL складно зрозуміти
- З іменування суворі угоди
- Розвивається занадто повільно

Ruby on Rails - об'єктно-орієнтована основа програми (фреймворк) для того, щоб розробляти веб-додатки, які написані мовою програмування Ruby. Ruby on Rails пропонує основну модель, Model-View-Controller для веб-додатків, а також відповідає за об’єднання сервера бази даних з веб-сервером.

Плюси:

- Простий і лаконічний синтаксис, помітний вплив Python, Eiffel і Ада
- Працює з винятками в стилі Python і Java
- Перевизначаєоператори
- Послідовний синтаксис
- Системні прямі виклики
- Під час роботи швидка поява змін
- Відсутність періоду компіляції
- Мова програмування об'єктно-орієнтована повністю. Всі дані в фреймворку являються Smalltalk об’єктами. Керівні конструкції – це єдиний виняток, і вони на відміну в Ruby не є об’єктами
- Є прибиральник сміття. Який працює і для інших бібліотек, не тільки для об’єктів Ruby
- Для Ruby розробка розширень на C не складна завдяки простому і зручному API та збору сміття
- Підтримує цикли з прив'язкою до змінних
- Підтримує блок коду (do … end або код взятий в { … }). В них є можливість перетворюватись в цикли або використовуватись в методах
- Дані динамічно типізовані

- В мові програмування Ruby розроблено достатньо готових шаблонів.
- Якщо операційна система дозволяє, може швидко завантажувати розширення
- Розроблена для багатьох платформ. Розроблялась на GNU/Linux, на багатьох версіях працює Microsoft Windows (частково, Win32), BeOS, DOS, Unix, OS/2, Mac OS тощо.

Мінуси:

- Працює надто повільно
- Досить громіздкі проекти
- debug код надто складний
- Концепція розвитку залежить від припущень

Apache Struts — фреймворк з доступним кодом для створення Java EE веб-застосунків. Розширює і використовує Java Servlet API, створюючи структуру MVC (Model-View-Controller).

Плюси:

- Легкий POJO-based
- Легкість в тестуванні коду
- Передача даних захищена
- Підтримується AJAX
- Технології валідації AJAX
- Підтримка шаблонів
- Підтримка різних видів результатів
- За допомогою плагінів простота розширювання
- Підтримується фреймворк Spring
- JFreechart плагін
- jQuery плагін (підтримується AJAX, динамічні таблиці, UI віджети, діаграми)
- Rome плагін

Мінуси:

- Складне в опануванні
- Важка оптимізація додатків
- Для інших розробників труднощі в розумінні коду

ASP.Net MVC - фреймворк який використовується для розробки веб-додатків, він реалізує модель Model-view-controller. Цей фреймворк створений компанією Microsoft.

Переваги фреймворку ASP.NET MVC:

- Завдання програми розділяються (логіка інтерфейсу, бізнес-логіка і логіка введення), великий вибір варіантів тестування та розробки на їх основі. Основні контракти MVC розроблені на інтерфейсі та тестуються за сприянням макетів об’єкта, вони створюють реальну поведінку об’єктів програми. У додатку можна проводити модульне тестування без допомоги контролерів у процесі ASP.NET, який полегшує тестування. Використовується будь – яка платформа яка працює сумісно з NET Framework.
- Компоненти платформи ASP.NET MVC легко налаштовуються або замінюються. Її можна доповнювати і розширяти. Фахівець має можливість підключати уявлення маршрутизацію URL-адрес, серіалізацію дій методів та інших компонентів. ASP.NET MVC використовує також інверсії інгредієнта керування (IOC) та структуру контейнера запровадження залежності (DI). Впроваджує об’єкти в клас, дозволяє не очікувати розробку об’єкта класом. Структура інверсії елемента керування діє так, коли один об’єкт потребує інший, то перші об’єкти повинні отримати інший об’єкт із іншого зовнішнього джерела (по типу файл конфігурації), і це прискорює тестування.
- Розширена піддержка маршрутизації ASP.NET. За допомогою цього компонента зіставлення, URL-адрес дає можливість розроблювати додатки з більш зрозумілими адресами, які потім можна буде використати у пошуку. Ці адреси потрібні для піддержки шаблонів іменування, вони не повинні мати розширення імен різних файлів.

Вони відповідають за адресацію адаптовану для SEO (пошукових систем) і для REST (передача репрезентативного стану).

- Використання розмітки в різних файлах сторінок ASP.NET (ASPX), головних сторінок (MASTER) та елементів керування (ASCX) як система стандартів. Використовувати функції ASP.NET можливо з платформою ASP.NET MVC, на кшталт, вбудовані вирази, прив'язка даних, декларативні серверні частини керування, шаблони, локалізація, головні сторінки тощо.

Мінуси:

- Висока межа входження
- Функція зберігання стану відсутня
- Формування бібліотек компонентів ускладнене

Коли стає потреба робити вибір фреймворку, треба ураховувати прект який планується. Це повинен бути основний критерій вибору тому, що ні один з існуючих фреймворків не буде універсальним для запланованого проекту. Нижче таблиця з порівняння основних можливостей розглянутих фреймворків.

Таблиця 1.1 – Характеристика популярних сучасних Back-end фреймворків

Критерій	Фреймворк						
	Zend	Cake PHP	Symfony2	Yii	ASP.NET	ROR	Struts
Мова	PHP	PHP	PHP	PHP	ASP.NET	Ruby	Java
Шаблон проектування MVC	+	+	+	+	+	+	+
Застосування ORM	+	+	+	+	ORM–незалежна	+	+
Інтеграція з БД	+	+	+	+	***	+	***

Продовження табл. 1.1

Критерій	Фреймворк						
	Zend	Cake PHP	Symfony2	Yii	ASP.NET	ROR	Struts
Механізми інтернаціоналізації і локалізації	+	+	+	+	***	+	+
Використання шаблонів при створенні інтерфейсів користувача	+	+	***	+	+	+	+
Створення і перевірка форм	+	+	+	+	+	+	+
Управління доступом на основі ролей	+	+	***	+	+	+	ні
Використання AJAX	+	+	***	+	+	+	+
ЧПУ (Friendly URL)	+	+	+	+	+	+	ні
Модульне тестування (unit–testing)	+	+	***	+	+	+	+
Система кешування	+	+	+	+	+	+	ні

При виборі фреймворку для проекту треба розглянути які завдання він може виконати, визначитись з його призначенням та функціями. На справді не кожен фреймворк має якісну інтеграцію, не завжди має можливість підтримувати ORM для роботи з БД і не завжди має хорошу документацію. Якщо ви плануєте створити сайт треба розглянути простий та зручний фреймворк та ретельно поставитись до його вибору, зважити всі переваги та недоліки.

Проаналізуємо переваги фреймворку:

- Гнучкість планування та розробки проекту;
- Результативне використання можливостей сервера;
- Доступний код;

- Фреймворк складається з перевірених базових налагоджених операцій та функцій, що забезпечує надійність та легкість web-розробок. Він створений на основі об'єктно-націленого програмування;
- Фреймворк постійно вдосконалюється і розвивається;
- Має багато супровідних документів, зразків розробок на різних мовах;
- Має всесвітню популярність, та багато розробників;
- Проект з застосуванням фреймворку легко супроводжується надалі, так як розробка має певні угоди;
- З фреймворком можливо сфокусуватися не на базових завданнях, а на вирішення архітектурних, як при роботі без його вживання;
- З фреймворком можливо вирішувати вузько поставлену задачу.

Одна з важливих переваг фреймворку – це дуже зручна і адаптована розробка не типових проектів. Багато не типових великих проектів не зроблені за допомогою готової CMS (наприклад, сайт знайомств, фотобанк з купівлі фотографій онлайн, twitter.com тощо) на фреймворках розробляють усі великі проекти. Коли веб-проект розробляється на основі фреймворку, він розвивається швидко та динамічно, коли змінюються вимоги змінюється і весь сайт, нам потрібно буде тільки створити другий розділ, або обновити дизайн, або змінити окремий модуль (блок).

Мінуси Back-end фреймворку:

Недоліків фреймворку дуже мало і вони незначні і досить умовні, якщо рівняти з перевагами:

- 1 файл = 1 клас;
- В проекті не використовується багато коду який займає місце в проекті;
- Він складний в освоєнні;
- Відсутність готових компонентів та модулів в мережі інтернет, ані, ані безкоштовних. Клієнт не може сам встановити, всі компоненти потрібно замовляти у розробників;
- Якщо розробляти функціонал для сайтів на CMS, це займе багато часу та обійдеться значно дорожче.

Шляхи використання фреймворку:

Потрібен великий досвід і багаж знань в роботі з додатками, щоб користуватися всіма перевагами і можливостями фреймворку. PHP–фреймворки допомагають знайти і позбутися частої помилки при роботі додатків, таку, як повторення коду. Також фреймворки систематизують процес роботи. Фреймворк є незамінним інструментом і помічником для PHP, мови програмування яка стрімко розвивається, і він допомагає організовувати ваш код.

Для кожної людини є свої переваги у використанні того чи іншого інструменту. Один розробник буде використовувати фреймворки, адже це буде допомога йому і прискорення процесу програмування. Для іншого спеціаліста це буде не потрібно і для нього це буде марною тратою часу. Взагалом використання фреймворків залежить від ступені професіоналізму. Але фреймворки створенні для заощадження часу і абстрагування від буденних, рутинних завдань.

Фреймворки, в основному застосовуються для планування складніших проектів ніж 2-3 сторінковий сайт з текстом.

Раціональне використання Back-end фреймворку.

Залежно від набору основних характеристик фреймворків, вони використовуються в різних сферах. Проаналізуємо базові критерії раціонального використання фреймворку. Таблиця 1.2.

Таблиця 1.2 – Критерії використання Back-end фреймворку

Назва	Характеристика
Підтримка баз даних	Питання підтримки баз даних в PHP фреймворк дуже важливе. Наприклад, CodeIgniter підтримує MySQL, Oracle і SQLite, а фреймворк Kohana не підтримує Oracle і SQLite. Частина фреймворків мають вбудований ORM – шар, частина – ні. Залежно від використовуваної бази даних для розробки проекту доводиться вибирати той чи інший PHP фреймворк.
Підтримка спільноти	Для комфортного вирішення проблем повинен мати хороше співтовариство, не тільки з точки зору розміру, але і в якості і в готовності допомогти. Навіть якщо це маленька спільнота, але є зворотний зв'язок від спільноти, це можна вважати плюсом. Так само плюсом є наявність російськомовного співтовариства.

Продовження табл. 1.2

Назва	Характеристика
Документація	Частина фреймворк мають слаборозвинуту, застарілу документацію. Частина не мають російської документації. Тому перед вибором фреймворка необхідно переконатися в тому що документація актуальна, вчасно оновлюється і доповнюється, і що інструкція із застосування проста в розумінні.
Продуктивність	Ключовим фактором при виборі так же може бути продуктивність фреймворка, наприклад, частина фреймворків підтримує кешування на достатньому рівні, частина – ні.
Безпека	Не всі фреймворки стійкі до різного роду атак, тому перед вибором фреймворка необхідно ретельно проаналізувати активність розвитку, розмір спільноти, а так само наявність вбудованих засобів для захисту від атак.
Поріг освоєння	Не всі фреймворки прості в освоєнні, це дуже важливо враховувати при виборі, так як на освоєння одного фреймворка може не вистачити й року, а на освоєння іншого – вистачить всього тижня.
Швидкість розробки	Так само слід врахувати той факт, що на одному фреймворці проект розробляється швидше, на іншому – ні. Приміром, розробки із застосуванням фреймворка zend триває більше ніж із застосуванням yii.
Архітектура	Фреймворк також повинен використовувати MVC архітектуру. Більшість хороших PHP фреймворків мають бібліотеки, плагіни, модулі та розширення. Це дуже добре для того щоб реалізувати велике коло функціоналу та вдосконалити і прискорити процес розробки.
Швидкість розвитку	Цей пункт так само дуже важливий, так як деякі фреймворки оновлюються раз на пару років (codeigniter), а деякі раз на пару місяців. Це дозволяє уникнути використання старого, недопрацьованого коду при розробці.
Зворотна сумісність	Не всі фреймворки зворотньо сумісні, тобто, при оновленні фреймворка в проекті може виникнути необхідність у повній переробці проекту. Частина фреймворків умовно зворотньо сумісна, наприклад, при оновленні молодшої частини версії (minor) все сумісно, а при старшій – ні. Так само великим плюсом є керівництво по переходу на нову версію фреймворка.
Використання шаблонів при створенні інтерфейсів користувача	Наявність даної функції значно спрощує створення інтерфейсу сайту , а іноді коли потрібний простий сайт то може звести написання дизайну до мінімума.
Створення і перевірка форм	Наявність даного функціоналу в фреймворку полегшує створення полів для вводу тексту, логіну чи паролю до мінімума. А також додає потрібну майже в усіх проектах функцію валідації форм.

1.2 Аналіз сучасних Front-End фреймворків

Для проектування адаптивного дизайну розробники використовують CSS фреймворки. На сьогодні існує багато таких фреймворків кожен з них має недоліки і свої переваги: якісь з них добре функціонують, але мають велику вагу. Розглянемо деякі найпопулярніші.

Front – End фреймворки:

React.js

У 2013 році Instagram і Facebook випустив відмінний фреймворк Java Script з ним ви зможете просто і легко планувати та створювати масштабовані та складні, динамічні додатки. Цей фреймворк використовується для побудови інтерфейсів призначених для користувача. Цей фреймворк має понад 89 тис. зірок на GitHub.

Плюси:

- Девіз React: «Вивчи один раз - пиши всюди»
- Безкоштовне та відкрите джерело
- Сумісний з вже написаним кодом
- Співпрацює з віртуальною функціональністю DOM

Мінуси:

- Алгоритм Virtual DOM повільний і неточний
- При роботі з сервером потрібно важке асинхронне програмування

Angular

AngularJS - це кістяк створений для розробки додатків від Google. Добре поєднується з динамічними веб-додатками і для статичних веб-сторінок використовує HTML. Цей фреймворк є незамінним також для дизайнерів, не тільки для користувачів ПЗ. AngularJS, Angular 4, Angular 2 і добре зарекомендували себе серед популярних фреймворків.

Плюси:

- Вихідний код відкритий
- Деякі фрагменти коду зберігаються для того, щоб використовувати потім

- Прив'язка даних формується на базі елементів Angular тому розробникам зустрічається менше помилок
- Підтримуються MVC елементи
- Добре працює в Agile середовищі
- Для тестів багато інструментів

Мінуси:

- Для новачків складний в розумінні
- В плані архітектури VUE простіше
- Потрібно вникнути в багату концепцію API Angular

Vue.js

Для створення інтерфейсів користувача використовується Vue.js. Для більшої застосовності він розроблений з нуля. Складається з базової бібліотеки, що робить акцент виключно на екосистеми та рівні уявлення, це допомагає опрацьовувати комплекси односторінкових додатків. Цей проект має понад 84 тис. зірок на GitHub.

Плюси:

- За замовчування не потребує ніяких компіляторів
- В процесі використанні, міграція з бібліотеки до фреймворку
- Керування односторінковими додатками
- Легкість процесів реінжинірингу коду читаністю та написання коду

Мінуси:

- Помилки в шаблонах Runtime
- Більш гнучкий підхід в React

Ember.js

MVC JavaScript один з найпопулярніших фреймворків. У 2011 році він з'явився під відкритим вихідним кодом. Він допомагає створювати односторінкові великі веб-додатки досить легко, він забезпечує двосторонню прив'язку даних. Якісно виконує роботу на стороні сервера, DOM-рендеринга. Підтримка сайтів Groupon, Discourse, Vine, LinkedIn.

Плюси:

- Легко налаштувати
- Пропонує великі інтерфейси
- Прив'язка даних двостороння

Мінуси:

- Структура проектів жорстка
- Відсутність стандартного набору різних UI-елементів

jQuery

Це бібліотека, а не повноцінний фреймворк, хоч і найпопулярніша, і найстаріша. JavaScript і jQuery міцно пов'язані вже давно. А з ліцензованою версією MIT ця бібліотека дає можливість розробникам додатків писати коротший код, тим самим скорочуючи робоче навантаження. Підтримує DOM-маніпуляції і разом з CSS та може згодитися для будь-якої задачі.

Плюси:

- За допомогою швидкій розробці широко використовується
- У всіх браузерах працює однаково
- Підходить новачкам для простих додатків

Мінуси:

- Багато функцій, що облегшують роботу з DOM, вже реалізовані.

Node.js

Він дозволяє розробляти швидкі і легкі програми. Сучасний, швидкий і простий. Paypal і GoDaddy - лише мала частина відомих компаній, які користуються Node.js. Ідеально підходить для розробки додатків, пов'язаних з I / O і додатків для ППД.

Плюси:

- Швидкий і простий
- Такі ПЗ можуть працювати на декількох хостах
- Швидкі сервери

Мінуси:

- Без тестування робити нічого в Node.js

Meteor.js

В його арсеналі є безліч функцій, які необхідні будь-якому розробнику для розробки бекенда, рендеринга фронт-енду та керування базами даних.

Плюси:

- платформа Full-Stack
- Розробка повнофункціональних додатків
- Реактивне програмування
- Швидка робота з даними
- Низький поріг входження

Мінуси:

- Для новачків складний інтерфейс
- Недостатня реалізація стандартних функцій

Skeleton

Це фреймворк real JavaScript.

Використання цього фремворку добре підходить для розробки веб-додатків і простих веб-сайтів. Поставляється з точками зупинки,сіткою CSS, нормалізації стилів і має функції API.

Плюси:

- Розроблений на JavaScript і CSS
- Адаптивне зображення для різних моніторів
- Можна примінити до будь-якого дизайну
- Кроссбраузерність

Мінуси:

- Недостатньо реалізовані деякі функції, особливо стандартні функції

Appcelerator Titanium

Це платформа для розроблення десктопних і мобільних додатків мовою JavaScript (HTML + CSS). Це також і хмарна платформа для поширення та

збірки програмного забезпечення і все, що буде потрібно для розробки насичених додатків.

Плюси:

- Простота у навчанні
- Високопродуктивна структура
- Для кроссплатформенних ПЗ

Мінуси:

- Titanium SDK оновлюється не раніше ніж SDK операційних систем
- Відсутність InterfaceBuilder
- Недостатня реалізація деяких функцій, у тому числі і стандартних

Aurelia.js

Цю платформу зазвичай називають фреймворком наступного покоління через те, що вона розроблює мобільні і веб-додатки.

Aurelia містить в собі багато невеликих, незалежних бібліотек, він модульований і це найкращий винахід в Aurelia, тому за допомогою цього можна розробити власний фреймворк та додати його в платформу.

Плюси:

- Великий вибір бібліотек
- Розробка власного фреймворку

Мінуси:

- Набір бібліотек потребує доопрацювання та доповнення

Twitter Bootstrap - це сукупність безкоштовних засобів з доступним вихідним кодом, який застосовується для розробки веб-сайтів, має шаблони HTML та CSS для кнопок, типографіки, навігації, форм та інших різних складників інтерфейсу. Має додаткові функції JavaScript. Bootstrap полегшує проектування динамічних веб-застосунків та веб-сайтів.

Twitter Bootstrap є одним з найпопулярніших CSS-фреймворків: багато пректів були створені на ньому і будуть створюватися не менше. Може бути, що причиною загальної визнаності є ранній старт або, можливо, важливий впливом стало створення проекту відомою компанією Twitter. У будь-якому

разі Bootstrap не зупиняється і на сьогодні є одним з найвідоміших фреймворків.

Вважаю, що головним проривом цього фреймворку є сітка. Bootstrap піддержує 12-ти колонкову адаптивну сітку. А з третьої версії даного фреймворку розроблюються проекти для мобільних пристроїв, і діє підхід, що спочатку проектуються версії сайту для них.

UI це набір деталей Bootstrap є одним з найпоширеніших. В ньому присутні елементи керування для усіх випадків. Якщо є невистачаючі елементи, то їх можна без проблем відшукати в мережі інтернет. Ситуації, коли ви не знайшли чогось для bootstrap, буде мінімум. Свою справу робить популярність.

Bootstrap однаково оновлює розробки для Sass і Less.

Розробники Bootstrap розуміють, що цей проект розрісся і його компоненти можуть не бути затребувані в рамках одного додатку. Щоб не виставляти багато файлів які не знадобляться, на офіційному сайті можна зібрати свій власний дистрибутив Bootstrap, який буде включати в себе тільки всі потрібні елементи проаналізуємо плюси та мінуси цього фреймворку:

Плюси:

- Хороша документаційна підтримка;
- Адаптивна концепція будови сайту;
- Кроссбраузерність;
- Швидка розробка сайту.

Мінуси:

- Важко створити свій унікальний дизайн через велику популярність, багато сайтів схожих на вигляд;
- Багато класів;
- Дуже великий розмір фреймворку уповільнює завантаження сторінки.

Zurb Foundation – це один з найвідоміших фреймворків, для розробки адаптивного дизайну.

Своєю популярністю він не поступається фреймворку Bootstrap. Довіру розробників до фреймворку Zurb Foundation ще раз підтверджує кількість зірок на GitHub. Foundation був розроблений в компанії Zurb (походження назви), яка кваліфікується на розробках дизайну різних веб-додатків. Цей проект створювався для власних, в першу чергу, потреб, тому направлений вирішувати складні задачі.

Foundation вміщує багату колекцію готових елементів (віджетів), сітку 12-ти колонкову та багато інших звичних речей. Бібліотека UI виглядає масштабно, її вистачає для розробки реальних прототипів. Суспільство біля Foundation давно сформувалося, і з цим зникають проблеми з пошуком додаткової інформації/плагінів.

Починаючи з 4-ї версії, Foundation діє по принципу спочатку мобільна версія дизайну – потім десктопна. З Foundation це означає повну адаптивність і застосування бібліотеки zepto.js, і відмову від jQuery.

З явних особливостей відзначають схожість з Bootstrap, тож коли є досвід, то простіше буде перейти на інший фреймворк. Проаналізуємо плюси та мінуси цього фреймворку:

Плюси:

- Валідація форм;
- RTL підтримка;
- Простота в розширенні функціоналу за допомогою великої кількості доповнень;
- Висока підтримка різними мобільними пристроями.

Мінуси:

- Висока межа входження.

Semantic UI – цей фреймворк був створений для побудови інтерфейсу веб-додатку з багатьма функціями щодо налаштування і можливості самостійно змінювати дизайн. На сьогодні він включає більш ніж 50 елементів UI і більш ніж 3000 змінних CSS, цей набір стрімко поповнюється. Semantic UI повністю адаптивний. Основний наголос фреймворку це симантичність

класів. Складові запропоновані Semantic UI містять в собі анімовані кнопки, значки, мітки, сегменти, спливаючі вікна, роздільники, зображення, Text Loader’и, Radio, кастомізовані Checkbox’и та багато іншого. Неможливо не згадати цікаві прийоми. Наприклад є можливість поставити мітки для того, щоб відобразилась “стрічка на елементі”.

Додатковою можливістю фреймворку є додавання картинкам і піктограмами 'Disabled' стан, в них можуть бути змінені розміри і різні кольори. Завдяки Fontello сервісу, існує багато популярних іконок.

Плюси:

- Простий при застосуванні;
- Різноманітні компоненти;
- Семантичний;
- Теми;
- Швидкий розвиток;
- Більше переваг ніж Bootstrap;
- Компоненти для інтерфейсу.

Мінуси:

- Препроцесори SASS, LESS не підтримуються;

Кожен з фреймворків які були описані має переваги та свої недоліки. Коли стає потреба робити вибір фреймворку, треба ураховувати проект, який планується. Це повинен бути основний критерій вибору тому, що ні один з існуючих фреймворків не буде універсальним для запланованого проекту.

Нижче таблиця з порівняння основних можливостей розглянутих фреймворків.

Таблиця 1.3 – Характеристика Front-End фреймворків

Можливості	Bootstrap	Zurb Foundation	Semantic UI
Препроцесори	LESS,SASS	SCSS	-
JS-бібліотеки	Jquery	Jquery	Jquery

Продовження табл. 1.3

Можливості	Bootstrap	Zurb Foundation	Semantic UI
Підтримка плагінів	-	Modernizr, FastClick	-
Адаптивна сітка	+	+	+
Слайдери	+	+	-
Випадаючі списки	+	+	+
Навбар	+	+	+
Canvas Меню	-	+	-

Недоліки і переваги при застосуванні Front-end фреймворків

З описаного раніше можемо відокремити деякі недоліки і переваги у застосуванні CSS-фреймворків:

Плюси:

- Кроссбраузерність;
- Блочна верстка;
- Правильно підібрана стилістика покращує доступ до тексту;
- Полегшення роботи над розробками;
- Одноманітність коду знижує вірогідність розбіжностей при роботі в команді;
- Допомагає новачкам правильно розробляти HTML-макет;
- З ним можливо застосовувати генератори коду.

Мінуси:

- Він залежний від додаткових бібліотек;
- Ускладнюється розуміння коду при використанні префіксів деякими фреймворками;
- Великий розмір.

Як ми побачили CSS, фреймворки, порівняно з невеликої кількості недоліків, мають більше переваг.

1.3 Постановка задачі

В результаті проведеного аналітичного огляду сучасних фреймворків було встановлено, що задача з розробки модуля реєстрації користувачів є актуальною при проектуванні і реалізації будь-якого веб-додатку. Для розв'язання такої задачі необхідно виконати такі завдання:

1) Формування MVC моделі модуля
2) Вибір фреймворку для реалізації моделі
3) Вибір програмного забезпечення для реалізації окремих компонентів сформованої моделі
4) Розробка бази даних
5) Розробка графічного дизайну модуля
6) Реалізація MVC моделі модуля
7) Тестування модуля.

2 ВИБІР МЕТОДУ РОЗВ'ЯЗАННЯ ЗАДАЧІ

2.1 Модель MVC

Модель–вигляд–контролер — архітектурний трафарет, який може бути використаний у проектуванні або розробці ПЗ.

Цей шаблон пророкує розділ системи на 3 повязані між собою частини – це вигляд інтерфейсу користувача, модель данних та модуль керування. Вживається для відокремлення моделі даних від вигляду інтерфейсу так, щоб змінювання інтерфейсу користувача впливали якомога менше на роботу з даними, а зміни в даних могли робити без допомоги інтерфейсу користувача.

Мета шаблону — змінний дизайн ПЗ, який повинен спростити розширення або доступні зміни програм. Та також з ним можливе повторне використання деяких частин програми. Коли використовуєш цю модель у великих проектах, вона впорядковує структуру і полегшує складність та зрозуміння їх.

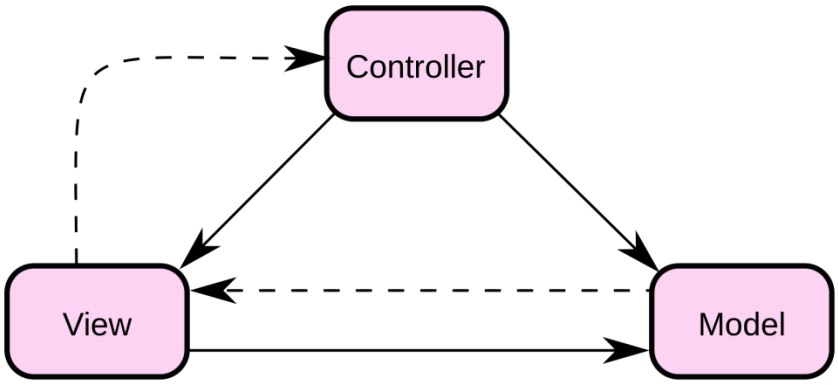

Рисунок 2.1 – Діаграма взаємодії між компонентами шаблону

V - частина, котра відповідає за вибір як відображати матеріал Django, за це відповідають відображення (views) і (templates).

C - частина, яка вибирає відображення операючись на вказівки користувача за це відповідає сам фреймворк, за допомогою налаштувань URLconf і підставляючи потрібну функцію Python для потрібного URL-а.

На основі того, що за «C» відповідає фреймворк і багато процесів

шаблонах, відображеннях, структурах, Django сміливо можна назвати «MVC фреймворком».

В ідеї MTV:

M «Model» - модель, визначає рівень доступу до основних даних. Також і все, що з данними повязано – як здобути доступ до них як їх протестувати і яке в них відношення між собою.

T «Template» - шаблони. Він відповідає за відображення і все, що з ним повязано: як конкретно буде розміщуватись відображення на веб-сторінці та інших документах.

V «View» - уявлення. Це міст який зєднує моделі і шаблони. Він пов'язаний з логікою яка обєднує необхідні шаблони і модель.

2.2 Django як фреймворк для створення web–додатків

Django - це програмний кістяк з різними можливостями, які підходять для створення веб-додатків і складних сайтів, розроблений на мові програмування Python.

Django – це фреймворк на мові Python, створений для веб-додатків. Основна ідея цього фреймворку - DRY (do not repeat yourself). Веб-системи на Django рекомендуються створювати відчужуваними, вони формуються з одного та більше додатків. Це і є най помітнішою відмінністю Django від інших фреймворків (Ruby on Rails наприклад). Ще відмінність в Django від інших фреймворків, це те, що обробники URL конфігуруються за допомогою систематичних виразів, а не задаються автоматично зі структури контролерів.

Django був розроблений для роботи з Apache (з mod_python модулем), а в якості БД використовувався PostgreSQL. На сьогодні джанго, крім PostgreSQL має можливість співпрацювати з іншими СУБД: SQLite, MySQL (MariaDB), DB2, Microsoft SQL Server, SQL Anywhere, Firebird, і Oracle. Джанго використовує свій особистий ORM, для роботи з базою даних, в ньому структура даних відтворюється класами Python, і по ній збирається схема БД.

Структура джанго дуже схожа на MVC. Контролер моделі MVC займає такий рівень, що в джанго має назву View, а логіка уявлення в Django

відповідає рівню шаблонів (Templates). Завдяки цьому структуру джанго деякі користувачі називають (MTV).

Фреймворк джанго спочатку проектувався для створення зручної роботи з новинними ресурсами, що залишило відбиток на архітектурі: фреймворк пропонує засоби, що роблять розробку інформаційних веб-сайтів швидкою. В джанго є вбудована база для управління вмістом і її можна додати в будь-який сайт створений на джанго яка може керувати декількома сайтами одного сервера одразу. І користувачу не потрібно створювати сторінки та контролери для адмін-частини сайту. Адмін-додаток дозволяє змінювати, створювати, і видаляти різні об'єкти вмісту сайту, фіксуючи всі попередні дії, та пропонує інтерфейс для керування групами і користувачами (з призначенням прав об'єктам).

Фреймворк джанго застосовується в таких відомих і масштабних сайтах, як Disqus, Instagram, The Washington Times, Mozilla, lamoda, Pinterest.

Можливості Django:

- API, ORM доступу до бази даних з підтримкою транзакцій
- Вбудований інтерфейс адміна, з багатьма перекладами на різні мови
- URL диспетчер на основі систематичних виразів
- Розширена система шаблонів зі спадкуванням і тегами спадкуванням
- Інтернаціоналізація
- Система кешування
- Використовується архітектура додатків які можливо додавати на будь-які сайти джанго
- «generic views» - шаблони дій контролерів
- Аутентифікація та авторизація, підключення сторонніх модулів аутентифікації: OpenID, LDAP та інші.
- «middleware» система фільтрів, призначених для створення додаткових розробників запитів по типу включених фільтрів для кєшування нормалізації URL, стиснення та піддержки анонімних сесій

- Бібліотека для опрацювання з формами (успадкування, та будова форм по діючої моделі бази даних)
- По тегам моделей даних і шаблонів додана автоматична документація, яка доступна за допомогою адмін додатку

Деякі компоненти джанго слабо пов'язані між собою, тому їх дуже просто можна замінювати на ідентичні та аналогічні. Не з усіма це так просто зробити (наприклад, з ORM). В основу фреймворку вбудовані компоненти які додають переваг джанго збільшують його можливості.

На базі Django створено дуже багато готових ідей які доступні під відкритою ліцензією, серед них системи для керування інтернет-магазинами та різні універсальні та цілеспрямовані проекти.

Django був створений, задля того, щоб люди переходили від зразок до готових послуг як можна швидше. Цей фреймворк вам допоможе створити додаток CRUD до кінця. З Django вам не треба буде винаходити колесо. Django може працювати з коробки що б ви могли зосередитися на продуктах для звичайних ідей та на бізнес-логіці.

Переваги Джанго:

- Django багатофункціональний щодо вирішення завдань пов’язаних з веб-розробкою. Ось декілька можливостей Django, які вам доведеться знайти окремо, якщо ви будете використовувати мікро-фреймворк:
  - Форми
  - ORM
  - Панель адміністратора
  - Міграції БД
  - Аутентифікація користувача
- Стандартизована структура.

Django задає стандарт проекту, як фреймворк. Це допомагає користувачам зрозуміти, де і як додавати ще одну функціональність.

Однакова структура, для проектів допомагає ще простіше знаходити готові рішення та отримувати допомогу від користувачів. Велика кількість

захоплених користувачів допомагає вирішати будь-яке завдання набагато швидше.

- Додатки Django.

Додатки в Django дає можливість розробникам поділити свій проект на частини. Додатки можна встановити через додавання в settings.INSTALLED_APPS. Такий підхід легко інтегрує вже готові рішення.

- Безпечний за замовчуванням.

Django безпечний за замовчуванням та має прилад запобігання розповсюджених атак наприклад SQL-ін'єкцій (XSS) і імітацію міжсайтових запитів (CSRF).

- REST Framework для розробки API.

Django REST Framework, часто скорочують як «DRF», це бібліотека яка допомагає побудувати API. Фреймворк має модульну та налаштовує архітектуру, вона гарно розробляє прості та складні API.В DRF політики автентифікації та рішень доступних з коробки. Він синхронізується с базовими класами за для того, щоб операції CRUD та вбудована утиліта тестувала API.

- GraphQL фреймворк щоб створювати API

REST API майже завжди потребують велику кількість запитів для того, щоб отримати всі необхідні данні. GraphQL – це мова вимог, які дозволяють розмінюватися пов'язаними відомостями набагато простіше.

Graphene-Django дає дозвіл легко додати відповідним чином більше функцій у наш проект. Форми, аутентифікація, моделі, політики згоди та інші функції можливостей Django можна користуватися для розробки GraphQL API. Бібліотека у такий спосіб поставляється для дослідження результату.

Мінуси Джанго:

- Django ORM сьогодні поступається SQLAlchemy останньої версії.

Django ORM основується на такому шаблоні як Active Record, який на справді гірше, ніж Unit of Work, що використовується у SQLAlchemy. На ділі це може виражатися в тому, що у Django моделі є можливість зберігатися за бажанням, а транзакції за замовчуванням відключені.

- Django повільно розвивається.

Django є немалим фреймворком. Це дозволяє розробникам створювати сотні різних додатків та модулів, але це знижує хуткість розробки Django. Крім того, фреймворк має підтримувати свої попередні версії, завдяки цьому він розвивається неквапливо.

Висновок щодо Джанго.

Хоча Django ORM не такий різносторонній, як SQLAlchemy, та багаторазові використані додатки і модулі уповільнюють ріст інфраструктури, відразу стає зрозуміло, що Django має бути головним претендентом на ролі фреймворку для пітоніста.

Легкі фреймворки по типу Flask, хоч і можуть бути вільні на відмінну від Django в конфігурації і екосистемі, можуть забрати зайвий час на пошук або якщо ви захочете створити ще бібліотек та подальшу можливість у довгостроковій перспективі.

Стабільність Django і користувачі цього фреймворку виросли з моменту першої версії. Навчальна документація та офіційні посібники з Django є одні з найкращих у своєму роді. А кожна нова версія Django супроводжується додатковими можливостями.

3 РОЗРОБКА МОДУЛЮ РЕЄСТРАЦІЇ КОРИСТУВАЧІВ

3.1 Вибір програмного забезпечення

Для оптимізації розробки проекту та зручності були використані наступні програми:

1. Пакет “ Django ”(серверна платформа)
2. Графічний інтерфейс “ db.sqlite3 ” для управління СУБД MySQL
3. Редактор програмного коду PyCharm

Кожен з цих додатків має низку унікальних, незамінних властивостей, які полегшують розробку і тестування web-додатку.

3.2 Розробка дизайну

Дизайн модуля реєстрації складається з двох частин: сторінки для реєстрації нового користувача (рис. 3.1) та сторінки аутентифікації (рис. 3.2).

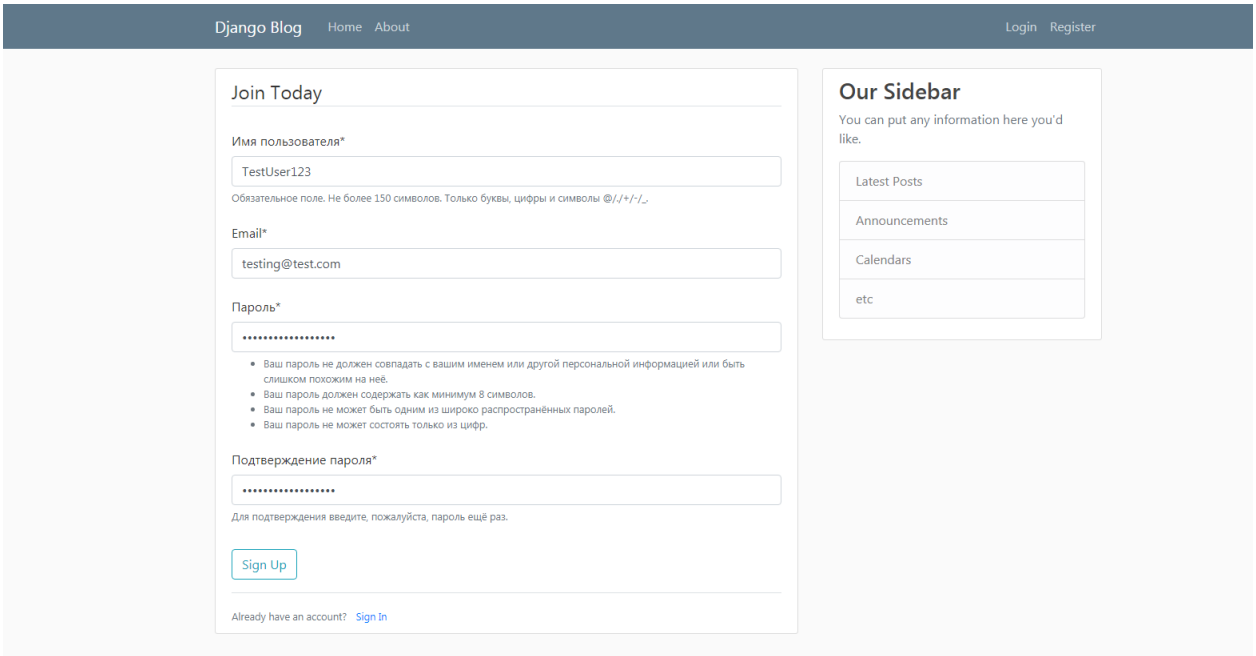

Рисунок 3.1 – Сторінка реєстрації нового користувача

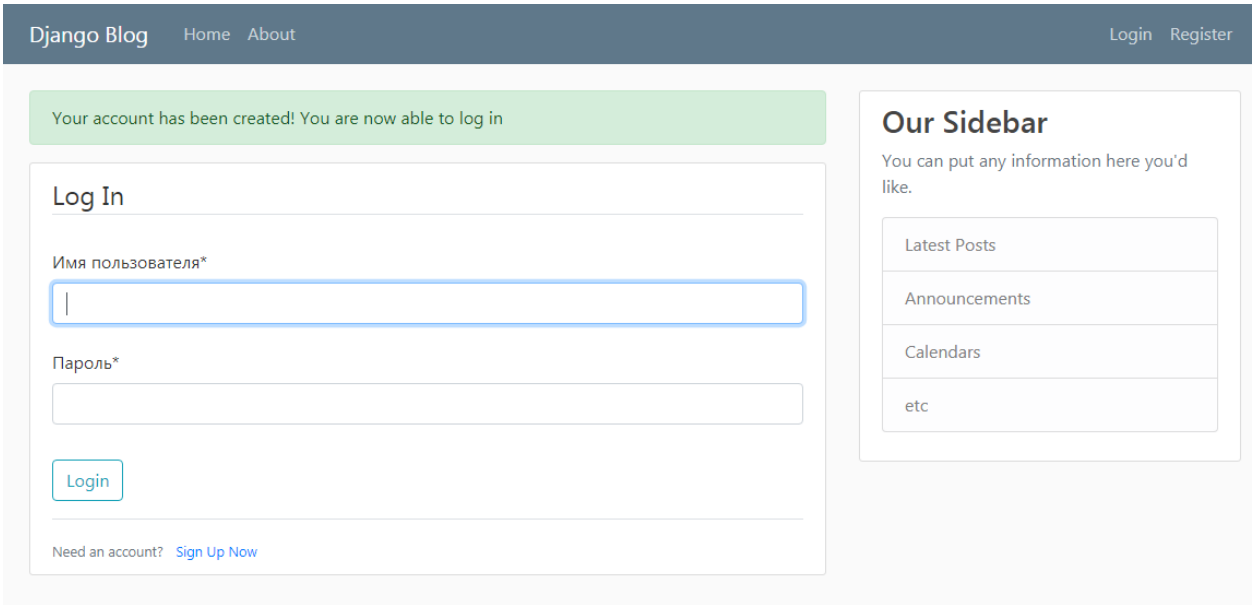

Рисунок 3.2 – Повідомлення про успішну реєстрацію та пропозиція увійти в систему

Графічні елементи та кольорова схема може бути змінена з урахуванням вимог користувача модуля.

3.3 Розробка бази даних

Структуру БД подамо у вигляді ER-діаграми (рис. 3.3).

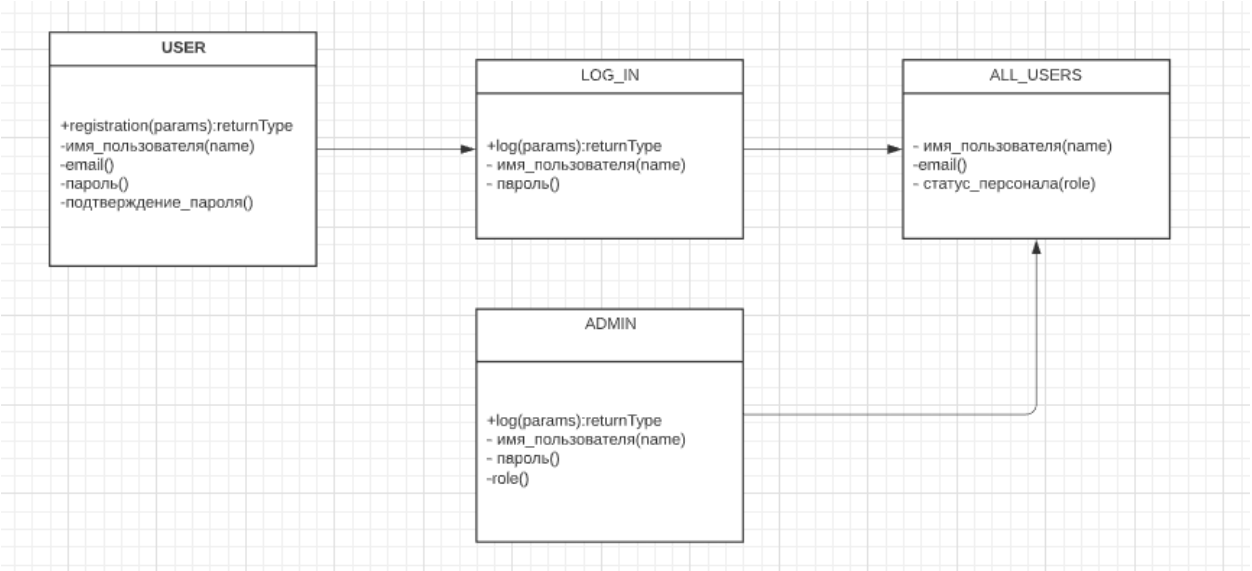

Рисунок 3.3 – ER-діаграма.

В даній ER-діаграмі ми бачимо 4 таблиці. Розглянемо їх детальніше:

USER - містить данні реєстрації нового користувача.

LOG_IN – містить данні входу до системи.

ADMIN – містить данні входу адміністратора сайту.

ALL_USERS – відображення всіх користувачів системи.

3.4 Реалізація моделі MVC

Django була розроблена за допомогою особливої підтримки «слабких зв'язків» та розмежуванням головних частин програми. Якщо ви будете дотримуватись цієї концепції – тоді вам буде легше редагувати додаток, без проблем для інших додатків.

Чи є різниця між проектом і додатком? Web-додаток, який пропонує деякий функціонал, ось деякі з них: Web-блог, сховище певних записів або просто додаток щоб голосувати. Проект - сукупність конфігурації сайту і програм. Проект може мати декілька додатків. Додаток може бути використаний кількома проектами.

Розробляючи додаток, переконайся, що ти перебуваєшь у такому ж каталозі, де й файл manage.py, та напишіть команду:

```
python manage.py startapp users
```

Ця команда створить каталог users:

- __init__.py
- admin.py
- apps.py
- migrations
- __init__.py
- models.py
- tests.py
- views.py

Ці файли містяться у цьому додатку.

Наш проект Django має файл settings.py, у якому будуть усі наші додатки: INSTALLED_APPS. Сюди ми маємо додавати усі додатки, які ми будемо використовувати у нашому проекті

Також у нашому проекті має знаходиться такий файл: urls.py – це головне сховище URL-ів в проекті. Він буде містить усі списки цього файлу urlpatterns:

```
from django.contrib import admin
from django.contrib.auth import views as auth_views
from django.urls import path, include
from users import views as user_views
urlpatterns = [
    path('admin/', admin.site.urls),
    path('register/', user_views.register, name='register'),
    path('login/',
auth_views.LoginView.as_view(template_name='users/login.html'),
name='login'),
    path('logout/',
auth_views.LogoutView.as_view(template_name='users/logout.html'),
name='logout'),
    path('', include('blog.urls')),
]
```

Нас цікавить path з цього списку. Функція path з’єднує введений вираз до адресної строки браузера із першими параметрами функції. Якщо буде введено, функція буде виконана, ім’я якої було передано другою функцією path. Третій параметр дає ім’я маршруту.

Створили функцію, яка обробляє відповідний маршрут:

```
from django.shortcuts import render, redirect
from django.contrib import messages
from .forms import UserRegisterForm

# Create your views here.
def register(request):
    if request.method == 'POST':
        form = UserRegisterForm(request.POST)
        if form.is_valid():
            form.save()
            username = form.cleaned_data.get('username')
            messages.success(request, f'Your account has been created! You
are now able to log in')
            return redirect('login')
    else:
        form = UserRegisterForm()
    return render(request, 'users/register.html', {'form':form})
```

Функція повертає сторінку реєстрації нового користувача на екран, а в разі пост-запиту та заповнення форми повертає перенаправлення на екран сторінку авторизації.

Текст файлу **register.html**:

```
{% extends "blog/base.html" %}
{% load crispy_forms_tags %}
{% block content %}
    <div class="content-section">
        <form method="post">
            {% csrf_token %}
            <fieldset class="form-group">
                <legend class="border-bottom mb-4">Join Today</legend>
                {{ form|crispy }}
            </fieldset>
            <div class="form-group">
               <button class="btn btn-outline-info" type="submit">Sign
Up</button>
            </div>
        </form>
        <div class="border-top pt-3">
            <small class="text-muted">
                Already have an account? <a class="ml-2" href="{% url 'login'
%}">Sign In</a>
            </small>
        </div>
    </div>
{% endblock content %}
```

Django дозволяє створювати моделі сторінок і наслідує їх. У цьому випадку файл register.html наслідує стартовий шаблон base.html:

```
{% load static %}
<!DOCTYPE html>
<html>
<head>

    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no">

    <!-- Bootstrap CSS -->
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
crossorigin="anonymous">

    <link rel="stylesheet" type="text/css" href="{% static 'blog/main.css'
%}">

    {% if title %}
        <title>Django Blog - {{ title }}</title>
    {% else %}
        <title>Django Blog</title>
    {% endif %}
</head>
<body>
    <header class="site-header">
      <nav class="navbar navbar-expand-md navbar-dark bg-steel fixed-top">
```

```
        <div class="container">
          <a class="navbar-brand mr-4" href="{% url 'blog-home' %}">
        Blog</a>
          <button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navbarToggle" aria-controls="navbarToggle" aria-
expanded="false" aria-label="Toggle navigation">
          <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse" id="navbarToggle">
            <div class="navbar-nav mr-auto">
              <a class="nav-item nav-link" href="{% url 'blog-home'
%}">Home</a>
              <a class="nav-item nav-link" href="{% url 'blog-about'
%}">About</a>
            </div>
            <!-- Navbar Right Side -->
            <div class="navbar-nav">
              <a class="nav-item nav-link" href="{% url 'login' %}">Login</a>
              <a class="nav-item nav-link" href="{% url 'register'
%}">Register</a>
            </div>
          </div>
        </div>
      </nav>
    </header>
    <main role="main" class="container">
      <div class="row">
        <div class="col-md-8">
            {% if messages %}
                {% for message in messages %}
                    <div class="alert alert-{{ message.tags }}">
                        {{ message }}
                    </div>
                {% endfor %}
            {% endif %}
            {% block content %}{% endblock %}
        </div>
        <div class="col-md-4">
          <div class="content-section">
            <h3>Our Sidebar</h3>
            <p class='text-muted'>

              <ul class="list-group">
                <li class="list-group-item list-group-item-light">Latest
Posts</li>
                <li class="list-group-item list-group-item-
light">Announcements</li>
                <li class="list-group-item list-group-item-
light">Calendars</li>
                <li class="list-group-item list-group-item-light">etc</li>
              </ul>
            </p>
          </div>
        </div>
      </div>
    </main>


    <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
integrity="sha384-
KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
crossorigin="anonymous"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.j
```

```
s" integrity="sha384-
ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
crossorigin="anonymous"></script>
    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
integrity="sha384-
JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
crossorigin="anonymous"></script>
</body>
</html>
```

3.5 Тестування модуля

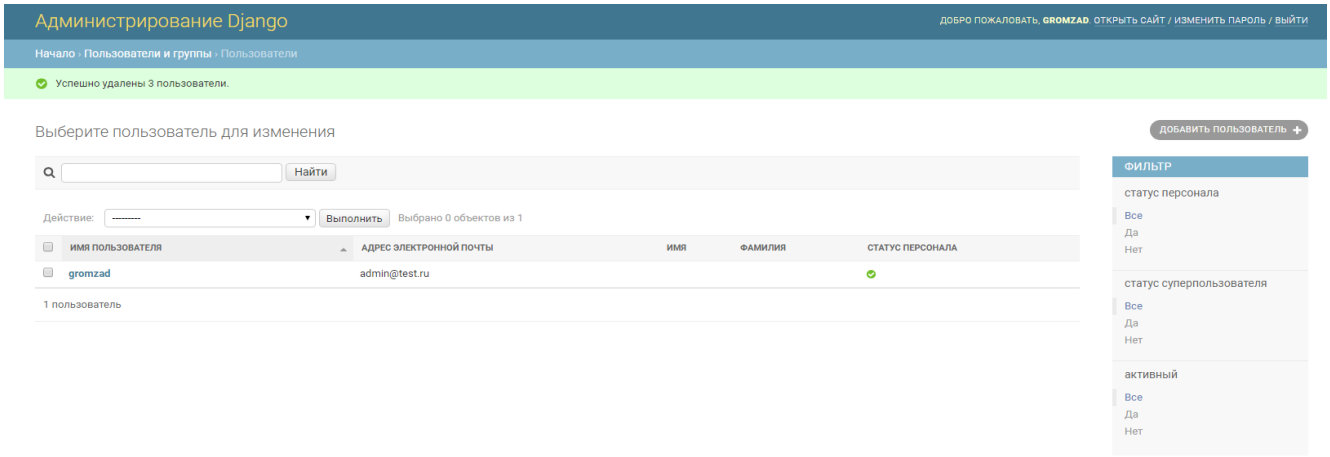

Рисунок 3.4 – Адмін-панель Django

Адмін-панель Django відображає всіх користувачів, що знаходяться у базі даних

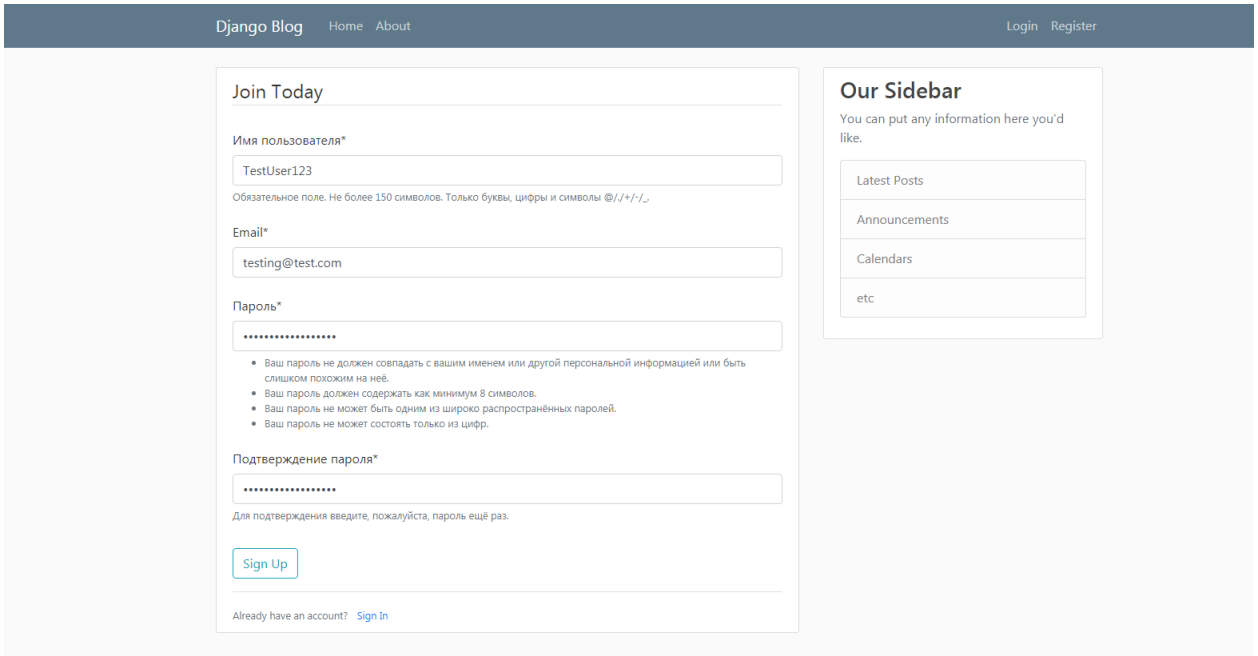

Рисунок 3.5 – Сторінка реєстрації нового користувача

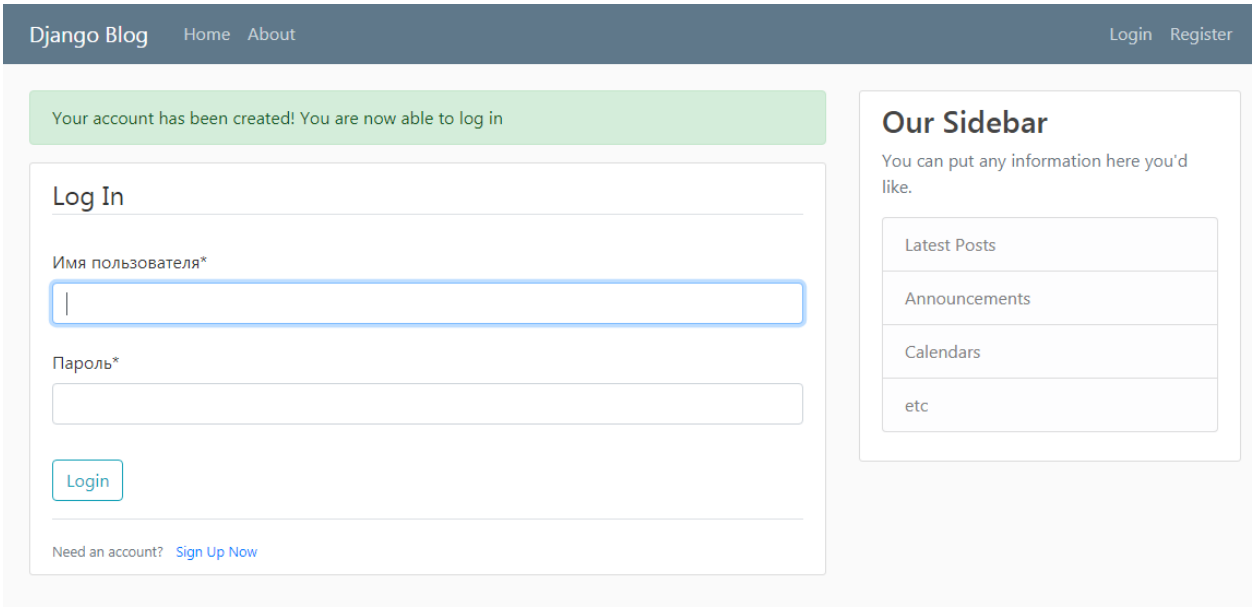

Рисунок 3.6 – Повідомлення про успішну реєстрацію та пропозиція увійти в систему

Рисунок 3.7 – Відображення всіх користувачів системи

Аналіз рис. 3.4-3.7 показує, що створення нових юзерів, використовуючи Front-End працює успішно.

ВИСНОВОК

В результаті виконання поставленої задачі розроблено додаток реєстрації нового користувача системи зі збереження у базу даних.

При цьому було виконано такі завдання:

1) Сформовано MVC модель модуля

2) Для реалізації моделі обрано фреймворк Django

3) Для реалізації окремих компонентів сформованої моделі використовувався графічний інтерфейс “ db.sqlite3 ” для управління СУБД MySQL та редактор програмного коду PyCharm

4) Сформовано базу даних з 4 таблиць, що містить дані реєстрації нового користувача, дані входу до системи, дані входу адміністратора сайту, відображення всіх користувачів системи.

5) Розроблено графічнтй дизайн модуля

6) MVC модель модуля реалізовано мовою програмування Python з використанням фреймворк Django

7) Проведено тестування модуля.

СПИСОК ЛІТЕРАТУРИ

1. Curtin Beau. Django Cookbook: Web Development with Django – Step by Step Guide 2nd edition — CreateSpace Independent Publishing Platform, 2016. — 182 p.
2. Aratyn Tom. Building Django 2.0 Web Applications . - Packt Publishing, 2018. 406 p.
3. Baumgartner Peter, Malet Yann. High Performance Django. - CreateSpace Independent Publishing Platform, 2015. — 184 p.
4. Bendoraitis Aidas, Kronika Jake. Django 3 Web Development Cookbook: Actionable solutions to common problems in Python web development : 4th Edition. — Packt Publishing, 2020. — 936 p.
5. Dauzon S., Bendoraitis A., Ravindran A. Django: Web Development with Python Packt Publishing, 2016. — 730 p
6. Fabrizio Romano, Gaston C. Hillar, Arun Ravindran. Learn Web Development with Python Packt Publishing, 2018. — 796 p.
7. Hillar G.C. Django RESTful Web Services Packt Publishing, 2018. — 320 p.
8. Jaiswal S., Kumar K. Learning Django Web Development Packt Publishing, 2015. — 405 p.
9. Rubio Daniel. Beginning Django Apress, 2017. — 593 p.
10. Vincent William S. Django for Beginners: Build websites with Python and Django Independently published, 2019. — 339 p.

ДОДАТОК

Текст файлу register.html:

```
{% extends "blog/base.html" %}
{% load crispy_forms_tags %}
{% block content %}
    <div class="content-section">
        <form method="post">
            {% csrf_token %}
            <fieldset class="form-group">
                <legend class="border-bottom mb-4">Join Today</legend>
                {{ form|crispy }}
            </fieldset>
            <div class="form-group">
               <button class="btn btn-outline-info" type="submit">Sign
Up</button>
            </div>
        </form>
        <div class="border-top pt-3">
            <small class="text-muted">
                Already have an account? <a class="ml-2" href="{% url 'login'
%}">Sign In</a>
            </small>
        </div>
    </div>
{% endblock content %}
```

файл register.html наслідує базовий шаблон base.html:

```
{% load static %}
<!DOCTYPE html>
<html>
<head>

    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no">

    <!-- Bootstrap CSS -->
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
integrity="sha384-
Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
```

```
crossorigin="anonymous">

    <link rel="stylesheet" type="text/css" href="{% static 'blog/main.css'
%}">

    {% if title %}
        <title>Django Blog - {{ title }}</title>
    {% else %}
        <title>Django Blog</title>
    {% endif %}
</head>
<body>
    <header class="site-header">
      <nav class="navbar navbar-expand-md navbar-dark bg-steel fixed-top">
        <div class="container">
          <a class="navbar-brand mr-4" href="{% url 'blog-home' %}">


            Blog</a>
          <button class="navbar-toggler" type="button" data-toggle="collapse"
data-target="#navbarToggle" aria-controls="navbarToggle" aria-
expanded="false" aria-label="Toggle navigation">
          <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse" id="navbarToggle">
            <div class="navbar-nav mr-auto">
              <a class="nav-item nav-link" href="{% url 'blog-home'
%}">Home</a>
              <a class="nav-item nav-link" href="{% url 'blog-about'
%}">About</a>
            </div>
            <!-- Navbar Right Side -->
            <div class="navbar-nav">
              <a class="nav-item nav-link" href="{% url 'login' %}">Login</a>
              <a class="nav-item nav-link" href="{% url 'register'
%}">Register</a>
            </div>
          </div>
        </div>
      </nav>
    </header>
    <main role="main" class="container">
      <div class="row">
        <div class="col-md-8">
            {% if messages %}
                {% for message in messages %}
```

```
                    <div class="alert alert-{{ message.tags }}">
                        {{ message }}
                    </div>
                {% endfor %}
            {% endif %}
            {% block content %}{% endblock %}
        </div>
        <div class="col-md-4">
          <div class="content-section">
            <h3>Our Sidebar</h3>
            <p class='text-muted'>

              <ul class="list-group">
                <li class="list-group-item list-group-item-light">Latest
Posts</li>
                <li class="list-group-item list-group-item-
light">Announcements</li>
                <li class="list-group-item list-group-item-
light">Calendars</li>
                <li class="list-group-item list-group-item-light">etc</li>
              </ul>
            </p>
          </div>
        </div>
      </div>
    </main>


    <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"
integrity="sha384-
KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
crossorigin="anonymous"></script>
    <script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.j
s" integrity="sha384-
ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
crossorigin="anonymous"></script>
    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
integrity="sha384-
JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
crossorigin="anonymous"></script>
</body>
</html>
```