Лаврінков І. О.,
студент 4 курсу
фізико-математичного факультету
Науковий керівник: **Карплюк С. О.,**
кандидат педагогічних наук, доцент
Житомирського державного університету імені Івана Франка

ПЕРЕВАГИ ВИКОРИСТАННЯ ФРЕЙМВОРКУ DJANGO У СТВОРЕННІ ONLINE-РЕСУРСУ ДЛЯ ПРОВЕДЕННЯ КОНФЕРЕНЦІЙ

Сучасний світ сьогодні неможливо уявити без сервісів і послуг, які надаються шляхом активного використання інформаційно-комунікаційних технологій і телекомунікацій. Не виключенням є і науково-дослідницька діяльність. У цьому контексті особливої уваги заслуговує питання створення цікавого online-ресурсу для якісної організації та проведення науково-практичних конференцій. Дана проблема є достатньо актуальною ще й з позиції широкої інформатизації українського суспільства, яке потребує мобільності й доступності до будь-якого роду інформації. Крім того, така постановка проблеми зумовлена ще й необхідністю не відставати від прогресу в сфері інформаційних технологій, оскільки важливо знати і вміти застосовувати нові послуги, знаходити вирішення та здійснювати наукові досягнення. Оптимальне використання інформаційних технологій дозволяє по-новому підійти до вирішення старих завдань, а також сформувати оригінальне розв’язання тих завдань, які раніше були нерозв’язаними. При цьому досягається зниження трудовитрат, економія часу і скорочення капіталовкладень.

Сьогодні Інтернет-простір наповнений значною кількістю як вітчизняних, так і зарубіжних сайтів, які висвітлюють необхідну інформацію щодо проведення науково-практичних конференцій. Виникає достатньо логічне запитання щодо вибору платформ або інших засобів, які сприяють якісному проектуванню такого роду електронних продуктів. З

огляду на це, метою нашого дослідження є спроба охарактеризувати переваги одного із засобів створення сайту для проведення конференцій.

З розвитком мережі Інтернет з'являється такі інструменти, які дозволяють створювати веб-додатки і при цьому надають можливість раціонально використовувати людські й часові ресурси. Одним з таких інструментів є веб-фреймворк *Django*, який і спробуємо охарактеризувати.

Django – це високорівнева веб-інфраструктура *Python*, яка дозволяє швидко створювати безпечні та підтримувані веб-сайти. Побудований досвідченими розробниками, *Django* надає можливість вирішувати значну кількість проблем щодо веб-розробки. Варто зауважити, що це безкоштовний і відкритий фреймворк, який має процвітаючу і активну спільноту, якісну документацію і безліч опцій для безкоштовної та платної підтримки [2].

Django слідує філософії «батарейки в комплекті» і надає майже всі можливості, які очікують від даного ресурсу розробники. *Django* може бути використаний для створення практично будь-якого типу веб-сайту – від систем управління контентом до соціальних мереж. Він може працювати з будь-якої клієнтської платформи і може доставляти контент практично в будь-якому форматі (включаючи *HTML*, *RSS*-канали, *JSON*, *XML* тощо).

Django допомагає розробникам уникнути багатьох поширених помилок безпеки, надаючи інфраструктуру, яка була розроблена для «правильного рішення», щоб автоматично захистити сайт. Яскравим прикладом цього є те, що *Django* забезпечує безпечний спосіб управління обліковими записами користувачів і паролями, уникаючи поширених помилок, таких як включення інформації про сеанс в файли *cookie*, де вона вразлива (замість цього файли *cookie* містять тільки ключ, а фактичні дані зберігаються в базі даних), або зберігання паролів у відкритому вигляді, замість їх хешів.

Хеш пароля – це значення фіксованої довжини, створене шляхом обробки пароля через криптографічну хеш-функцію. *Django* може перевірити правильність введеного пароля, пропустивши його через хеш-функцію і порівнявши висновок зі збереженим значенням хеша. Завдяки «однобічному» характеру функції, навіть якщо збережене хеш-значення скомпрометовано, зловмисникові буде складно витягти вихідний пароль.

Django забезпечує захист від багатьох вразливостей за замовчуванням, включаючи *SQL*-ін'єкцію, міжсайтовий скриптинг, підробку міжсайтових запитів і клікджекінг [4].

Django використовує компонентну «*shared-nothing*» архітектуру (кожна частина архітектури не залежить від інших, і отже, може бути замінена або змінена при необхідності). Чіткий поділ між різними частинами означає, що вона може масштабуватись для збільшення трафіку шляхом додавання обладнання на будь-якому рівні: кешуються сервери, сервери баз даних або сервери додатків. Деякі з найбільш відвідуваних сайтів успішно маштабуются *Django* для задоволення своїх вимог (наприклад, *Instagram*, *Disqus* та інші).

Даний інструмент є достатньо зручним, оскільки код *Django* написаний з використанням принципів і шаблонів дизайну, які заохочують створення підтримуваного і багаторазового коду. Зокрема, він використовує принцип *Do not Repeat Yourself* (*DRY*), тому немає непотрібного дублювання, що зменшує кількість коду. *Django* також сприяє угрупованню пов'язаних функцій в багаторазові «додатки» і на більш низькому рівні групує пов'язаний код модулів відповідно до *Model View Controller* (*MVC*).

Django написаний на мові програмування *Python*, яка є простою у вивченні й водночас достатньо потужною. Ця мова має ефективні структури даних високого рівня та простий, але ефективний підхід до об'єктно-орієнтованого програмування. Елегантний синтаксис *Python* і

динамічний набір, разом із інтерпретованою природою, роблять його ідеальною мовою для сценаріїв та швидкого застосування додатків у багатьох областях на більшості платформ [3].

Веб-додатки написані на *Django* зазвичай групують код, який обробляє кожен з цих кроків, в окремі файли:

– *URLs*: Хоча можна обробляти запити з кожної *URL*-адреси за допомогою однієї функції, набагато зручніше писати окрему функцію для обробки кожного ресурсу. *URL-mapper* використовується для перенаправлення *HTTP*-запитів у відповідне подання на основі *URL*-адреси. *URL-mapper* також може витягувати дані з *URL*-адреси відповідно до заданого шаблону і передавати їх у відповідну функцію у вигляді аргументів;

– *View:* Подання (*view*) – це функція обробника запитів, яка отримує *HTTP*-запити і повертає відповіді. *View* має доступ до даних через моделі (необхідних для задоволення запитів і делегування відповіді в шаблони);

– *Models:* Моделі є об'єктами *Python*, які визначають структуру даних програми та надають механізми для управління (додавання, зміни, видалення) та виконання запитів до бази даних;

– *Templates: Template* (шаблон) – це текстовий файл, який визначає структуру або розмітку сторінки (наприклад *HTML* сторінки), з полями для підстановки, які використовуються для подання актуального вмісту. *View* може динамічно створювати *HTML* сторінки, використовуючи *HTML* шаблони і заповнюючи їх даними з моделі (*model*). Шаблон може бути використаний для визначення структури файлів будь-яких типів, не обов'язково *HTML* [1].

Таким чином, можна зробити висновок, що *Django* є достатньо ефективним інструментом, який дозволяє створювати веб-додатки, від сайтів-візитівок до масштабних високонавантажених проектів, а в нашому

досліджені якісний сайт для організації та проведення науково-практичних конференцій.

Література

1. Django Введение [Електронний ресурс] – Режим доступу: https://developer.mozilla.org/ru/docs/Learn/Server-side/Django/% D0%92%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%B8%D0%B5
2. Django [Електронний ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/Django
3. Python Tutorial [Електронний ресурс] – Режим доступу: https://docs.python.org/3/tutorial/index.html
4. Знакомство с Django [Електронний ресурс] – Режим доступу: https://djbook.ru/rel1.9/intro/overview.html