МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХЕРСОНСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ НАУК ФІЗИКИ ТА МАТЕМАТИКИ
КАФЕДРА ІНФОРМАТКИ, ПРОГРАМНОЇ ІНЖЕНЕРІЇ ТА ЕКОНОМІЧНОЇ КІБЕРНЕТИКИ

РОЗРОБЛЕННЯ WEB-ДОДАТКУ АВТОМАТИЗОВАНОГО МОНІТОРИНГУ ТЕМПЕРАТУРНИХ НОРМ В ХДУ

Кваліфікаційна робота (проєкт)

на здобуття ступеня вищої освіти "бакалавр"

Виконав: студент 4 курсу
Спеціальності 121 Інженерія
програмного забезпечення
Освітньо-професійної програми
«Інженерія програмного забезпечення»
першого (бакалаврського) рівня освіти
Шкворець Владислав Владленович
Керівник доктор фізико-математичних
наук, професор Песчаненко Володимир
Сергійович, доктор педагогічних наук,
старший викладач Співаковський
Олександр Володимирович
Рецензент кандидат педагогічних наук
Гончаренко Тетяна Леонідівна

Херсон – 2020

ЗМІСТ

ВСТУП

Веб-розробник має можливість вибрати з широкого спектру веб-фреймворків, використовуючи Python як серверну мову програмування. Він може скористатися повнофункціональними веб-фреймворками Python, щоб прискорити розробку великих і складних веб-додатків, скориставшись рядом надійних функцій і інструментів. Крім того, можна також вибрати серед безлічі мікро- і малих веб-фреймворків Python для створення простих веб-додатків. В нашому випадку ці фреймворки допоможуть створити свій веб-додаток для моніторингу температурних норм, який в майбутньому може бути легко вдосконалений або оновлений. Цей проект буде вирішувати доволі складне питання для роботодавця, як дотримування температурних норм у робочих або навчальних приміщеннях.

Метою роботи є розробка веб-додатку по автоматизованому моніторингу температурних норм у приміщеннях Херсонського державного університету.

Об'єктом роботи є веб-додатки для відслідковування температурних норм в приміщеннях університету.

Предметом роботи є веб-додатки для відслідковування температурних норм в приміщеннях університету.

Завдання роботи:

1. Аналіз аналогів та температурних норм в навчальних закладах
2. Визначення фунціоналу додатку
3. Обґрунтування вибору програмного забезпечення та порівняння характеристики програмних компонентів, а саме:
    - швидкість,
    - підтримка роботи з базами даних.
4. Розгорнення проекту

Структура роботи. Робота складається зі вступу, трьох розділів, висновків та списку використаних джерел.

РОЗДІЛ 1. Аналіз аналогів та температурних норм повітря у приміщеннях

1.1 Аналізування аналогів

В нашому великому світі технологій ми можемо знайти рішення для будь-якого нашого питання, проте не завжди воно відповідає усім нашим критеріям. Так само і в нашому випадку, є багато рішень цього завдання, проте усі мають своїй вади. Сьогодні ми розглянемо самі відомі системи для моніторингу температурних норм в приміщеннях.

Testo Besure- світовим лідером з виробництва портативної вимірювальної техніки. Продукція компанії Testo сьогодні ділиться на 4 основні групи:

- Портативні вимірювальні прилади для вимірювання температури, вологості повітря, витрати повітря, тиску, pH, провідності, освітленості, рівня шуму, активності води і прилади для настройки холодильних систем.
- Пірометри і тепловізори - прилади для безконтактного виміру температури, що реєструють інфрачервоне (теплове) випромінювання від об'єкта і перетворюють його в значення температури або в теплове зображення.
- Газоаналізатори - прилади для аналізу різних характеристик димових газів в системах опалення або промислових пальниках, з метою налаштування обладнання відповідно до екологічних вимог.
- Вимірювальні системи стаціонарно розміщених приладів, що виконують безперервні вимірювання вологості і температури на виробництві, в холодильних камерах, на складах, в музеях, лабораторіях, в агрегатах і технічних системах.

Усі ці датчики мають свій веб-додаток для зображення даних. Проте це рішення не підходе для нашого завдання бо система має закритий початковий код, а також встановлення цієї системи потребує набагато

більше матеріальних витрат. Наступний аналог це **Anemone cloud** про цю систему зовсім нічого не відомо, окрім того щоб приєднатися до цієї системи потрібно зоставляти договір. Але в цього проекту є і більш важливий недолік для нас. Проект працює тільки на території Росії та Європи.

Третій та останній аналог це напіввідкрита платформа **Oregon Scientific** він має відкритий код для збіра даних з датчиків, проте самі датчики потрібно купувати саме цієї фірми. Тобто ми можемо розробити веб-додаток для збора та зображення даних, які будуть вимірюватися завдяки цим датчиками.

Рис. 1 Датчик Orgon Scientific

Проте якщо знадобиться в майбутньому вдосконалювати або масштабувати систему то ці датчики можуть вже бути не в продажі, в такому випадку веб-додаток прийдеться перероблювати під нові датчики цієї компаній, або взагалі під інші системи які можуть з’явитися в майбутньому.

Тому в нашому випадку розробка свого веб-додатку потребує вилучення цих мінусів із нашої системи.

Перший мінус який потрібно вилучити з нашого проекту це складність оновлення датчиків, тобто якщо в майбутньому потрібно буде оновити датчики то не буде потрібності шукати такі самі датчики.

Другий мінус це складність вдосконалення системи, якщо в майбутньому буде потрібно не тільки моніторинг температур, а й вологість або швидкість вітру то наш проект повинен легко вдосконалюватися.

1.2 Аналіз температурних норм в навчальних закладах

В даному розділі ми розглянемо офіційні температурні норми повітря в приміщеннях навчального закладу, навіщо вони потрібні, а також приклади зображення цих норм у графічному методі.

Хочете, щоб ваші співробітники завжди працювали ефективно? Погодьтеся, що важко працювати коли людина відчуває дискомфорт. А для цього потрібно дотримуватися хоча б температурних нормативів на робочому місці, вона повинна бути зручною. Роботодавець зобов'язані створювати не тільки безпечні умови на робочому місці в офісі або на робочому місці, але і підтримувати комфортну атмосферу-температуру, рівень вологості і т.д. Також не дотримування цих норм може не тільки погіршити ефективність співробітників або учнів, а також стану здоров’я. Для токо щоб зрозуміти, які показники температур є задовільні, а які ні. Нам потрібно звернутися до офіційних документів **державної санітарних правил і норм влаштування**. Цьому документі прописані норми, яким потрібно дотримуватися.

Приміщення	**Температура повітря (норма) градусів за Цельсієм**
Класи та кабінети	17-20
Майстерні по обробці металу і дерева	16-18
Спортивний зал	15-17

Роздягальні при спортивному залі	19-23
Актовий зал	17-20

Рис. 1.2 Розрахункова температура повітря у приміщеннях

Також нам потрібно не тільки знати ці норми, а також як ці дані можливо зобразити.

Для цього ми будемо застосовувати декілька варіантів:

Перший метод це табличний. Цей метод буде використовуватися, по-перше для систематичного аналізу даних, а також для порівняння декількох датчиків.

Дата		12.02.2020								
Час		15:16:44	17:49:37	22:37:04	12:00:00	08:00:00	11:00:00	14:00:00	17:00:00	18:00:00
Корпус 1	Басейн	18				14	15	16	15	11

Рис. 1.3 Табличний метод зображення даних

Другий метод це графічний. На цей раз зображення буде можливо тільки для одного датчику, проте за довгий період.

Рис. 1.4 Зображення температурних норм

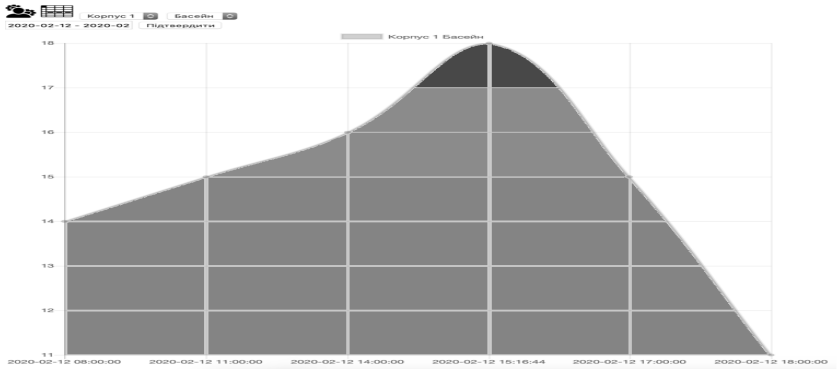


Рис. 1.5 Графічний метод зображення даних

РОЗДІЛ 2
ОБҐРУНТУВАННЯ ВИБОРУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТА ПОРІВНЯННЯ ХАРАКТЕРИСТИКИ ПРОГРАМНИХ ТА АПАРАТНИХ КОМПОНЕНТІВ

2.1 Порівняння фреймворків flask та django

Фреймворк (Framework, каркас, платформа, структура, інфраструктура) – програмна платформа, що визначає структуру програмної системи або інфраструктура програмних рішень, що полегшує розробку складних систем [1]. Тобто, це є багаторівнева структура, яка вказує, як програми можуть або повинні бути побудовані, і як їх компоненти будуть взаємозв'язані. Фреймворки можуть включати в себе програми підтримки, компілятори, бібліотеки коду, набори інструментальних засобів і інтерфейси прикладного програмування (API), які об'єднують всі різні компоненти, що забезпечують розробку проекту або системи.

Проектувальники фреймворків мають на меті полегшити розробку програмного забезпечення, дозволяючи розробникам програм присвятити свій час розробці програмного забезпечення, а не працювати з більш стандартними інструментами низького рівня робочої системи, тим самим скорочуючи загальний час розробки.

Позитивною стороною фреймворку є те, що він забезпечує стандартність структури програми. Так об’єктно-орієнтовані фреймворки складаються з класів (визначених та абстрактних) та інтерфейсів [2]. Тому програмісту достатньо використати вже існуючу реалізацію певної задачі чи додати певний функціонал для вирішення проблеми за допомогою своїх власних класів. Більшість фреймворків розраховані на вирішення конкретних завдань. Зрозуміло, що кожен фреймворк має свої особливості

роботи з ним, які потрібно розуміти перед початком розробки проекту з його використанням.

Фреймворки визначаються певними особливостями:

- розширюваність: користувач може розширити структуру, наприклад, шляхом перевизначення деяких компонентів фреймворку додаванням коду, щоб забезпечити додаткову функціональність;
- кросплатформність (багатоплатформність) – властивість програмного забезпечення працювати на декількох програмних чи апаратних платформах;
- універсальний інтерфейс для програміста (API) та користувача;
- сумісність і повторно використовуваний код;
- стійкість коду: платформа активно забезпечує безпеку додатків, реалізуючи ряд дуже важливих механізмів.

Розглянемо приклад за допомогою Flask. Для того, щоб браузер відобразив рядок «Hello, world», необхідно програмно відкрити порт на комп'ютері і прослуховувати цей порт для спілкування з сервером. Коли запит на цей порт буде отримано, програма дізнається, який шлях (Uniform Resource Locator, URL - стандартизована адреса певного ресурсу) був запитаний і які параметри було передано. Потім програма вибирає дані, які треба відобразити, в даному випадку символьний рядок і повертає у вигляді HTML відгуку на сторінку браузера. При цьому необхідно обробляти такі запити як HTTP-пакети, кодування та декодування запитів у відповідь на кодування в форматі пакетів, постійне з'єднання з сервером, аутентифікація тощо. Але якщо використовувати веб-фреймворк, то програмісту треба створити метод, який містить шлях до ресурсу та саме ресурс (див. рис. 1.1), який треба відобразити [12]. А фреймворк буде піклуватися про відкриття, прослуховування порту, HTTP комунікацію, кодування та декодування пакетів, передавання html-сторінки, спілкування з оголошеннями бази даних, аутентифікації тощо.

Починаючи розробляти програми в Python, ви, швидше за все, розробили так звані «додатки командного рядка». Користувачеві потрібно запустити ці скрипти в оболонці, або командному рядку і передати дані, що вводяться в якості аргументів, або через стандартний ввід. Кілька років тому, природний розвиток від побудови таких додатків до створення GUI додатків - програми, яка дозволяє користувачеві взаємодіяти за допомогою миші і клавіатури, в якій містяться кілька меню та інших інтерактивних елементів. В наші дні помітний великий зсув у бік розробки веб додатків - ваші користувачі можуть працювати з вашою програмою через браузер. Якщо ви зважилися на створення веб додатки, і ви вирішили зробити це в Python, вам дійсно знадобиться так званий веб фреймворк. Побудова бекенд логіки, призначеного для користувача інтерфейсу і зборів всього пов'язаного з Інтернетом і навігацією користувача в вашому додатку через браузер складається з повторюваних і нудних частин. Веб-фреймворк націлений на реалізацію всіх функціональних можливостей, загальних для більшої частини веб-додатків, таких як зіставлення URL-адрес частинам коду Python.

Саме від того, що являє собою той чи інший фреймворк залежить те, що залишається розробнику для створення програми. Ключова різниця між Flask і Django це:

- Flask реалізується з мінімальними надбудовами, які цілком надані аддонам або розробнику;
- Django слід філософії «все включено», і дає вам великий асортимент для роботи.

Ми розглянемо різницю між ними більш детально.

Фреймворк Flask

Flask – веб-фреймворк для побудови простих сайтів зі статичним контентом, наприклад, блогів. Він забезпечує весь необхідний функціонал і дозволяє налаштувати веб-додаток декількома способами. Flask – це

легкий і розширюваний веб-фреймворк на основі мови Python, який є мікрофреймворком, оскільки не має вбудованих спеціальних засобів чи бібліотек: у ньому відсутній рівень абстракції для роботи з базою даних, перевірки форм, панелі адміністрування або інші компоненти, які надають широковживані функції за допомогою сторонніх бібліотек. Однак, існують розширення для встановлення об’єктно-реляційних зв’язків, перевірки форм, контролю процесу завантаження, підтримки різноманітних відкритих технологій аутентифікації та декількох поширених засобів, типових для фреймворків [8].

Flask має на меті зберегти серцевину фреймворку невеликою, але дуже розширюваною. Це робить написання додатків або розширень дуже простим і дає розробникам можливість вибирати конфігурації (додаткові бібліотеки), які вони хочуть для їх застосування, без накладення будь-яких обмежень на вибір бази даних, рушія шаблонів і так далі [9].

Flask містить такі особливості:

- випускається з вбудованим сервером розробки, який допомагає у процесі розробки та тестуванні програм;
- вбудована підтримка тестів (при виконанні коду, якщо будь-яка помилка виникла в дорозі, трасування стека помилок буде показано на веб-сторінці);
- використання об'єкту Session - функція, яка зберігає сеанс користувача. Вона зберігає запити користувача, щоб програма могла запам'ятати різні запити від користувача і швидко відтворити їх;
- Підтримка шаблонів, що обслуговуються гнучким шаблонізатором Jinja2;
- управління запитами RESTful: декоратор маршруту, який допомагає прив'язати функцію до URL-адреси, може приймати методи HTTP як аргументи, які прокладають шлях до створення API в ідеальному порядку;

•легка робота з NoSQL базами даних;

•використання протоколу інтерфейсу шлюзу веб-сервера (WSGI) під час роботи з запитами від клієнтів, і він є 100% сумісним з WSGI 1.0 [3].

Фреймворк має один великий недолік, пов'язаний з безпекою. Flask захищає тільки від однієї з найбільш поширених проблем сучасних веб-додатків: міжсайтових сценаріїв (XSS) [8].

Перше, що нам потрібно зробити в нашому списку роботи з Flask, це встановити Flask. Це робиться легко за допомогою pip. Залежно від того, як ваш pip встановлений, і в якій версії Python ви працюєте, вам може не знадобиться використовувати 3 або прапор -user в наступні команді:

```
pip3 install flask --user
```

Після цього, створіть файл Python, під назвою flaskhello.py, і введіть наступний код:

```
from flask import flask
app = flask(_name_)
@app.route("/")
def hello():
     return"hello,world!"
if _name_ == "_main_":
     app.run()
```

Перший рядок імпортує Flask

Третій рядок инициализирует змінну додатки, використовуючи атрибут __name__

П'ятий рядок містить в собі чудеса Flask. @ App.route - це декоратор Python. Він бере функцію знизу і модифікує її. В даному випадку, ми використовуємо його для маршрутизації трафіку з певного URL в розташованій нижче функції. Використовуючи різні виклики @ app.route, ми можемо «спровокувати» різні частини коду, коли користувач відвідує

різні частини нашого застосування. В даному випадку, у нас тільки один маршрутизатор «/», який є коренем за замовчуванням в нашому додатку.

У шостому рядку, функція під назвою hello не так вже й важлива. Замість виклику цієї функції з тієї чи іншої частини нашого коду, вона буде викликана автоматично. Це хороша практика для того, щоб дати їй релевантне назву.

Сьома стоку повертає рядок нашому користувачеві. Зазвичай ми рендеремо шаблон або обробляємо HTML, щоб користувач зміг побачити акуратно оформлену сторінку. Дев'ята рядок - це звичайний шаблон Python, який використовується для того, щоб переконатися в тому, що ми не запускаємо нічого в автоматичному режимі, якщо наш код був імпортований з іншого скрипта Python.

У десятому рядку викликається метод run () додатки, яке ми ініціалізували в третьому рядку. Це запускає сервер розробки для Flask і дає нам можливість відвідати наше веб додаток з нашої локальної машини шляхом відвідування localhost.\

Ви також можете запустити команду:

```
python3 flaskhello.py
```

У підсумку отримати повідомлення, на зразок цього прикладу:

```
Python
1  >>> Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Рис. 1.2. Старт сервера Flask

В даному повідомленні «5000» це номер порту, який наш додаток запускає (у вас він може відрізнятися, наприклад «5003»), а '127.0.0.1' означає, що додаток запущено на локальному хості, доступ до якого є тільки у нашої машини . Якщо ви відкрийте браузер і перейдете по http://127.0.0.1:5000/ (заміна номера порту необхідна), ви побачите сторінку, яка видає вітання «Hello, World!".

Flask може генерувати URL. Для створення URL, використовуйте функцію url_for (). Вона приймає ім'я функції в якості першого аргументу, а також ряд ключових аргументів, кожен з яких відповідає змінної частини URL правила.

Тут також використовується метод test_request_context (), який пояснений нижче. Він каже Flask, як потрібно обробляти запит, навіть якщо ми взаємодіємо через йшов Python. Чому ми використовуємо побудова URL замість їх жорсткого завдання в шаблонах? На то є три хороші причини:

Реверсування, часто є більш описовим методом і дозволяє змінювати URL на одному диханні.

Автоматичне екранування спеціальних символів, УНІКОД. Вам навіть не доведеться замислюватися над цим.

Якщо ваш додаток знаходиться поза кореневої URL (наприклад, / myapplication замість /), функція url_for () буде це правильно обробляти для вас.

2.3. Фреймворк Django

Django – це високорівневий веб-фреймворк з відкритим вихідним кодом для серверних веб-додатків на основі мови Python. Основними його особливостями є простота, гнучкість, надійність і масштабованість. Він розроблений на базі так названих «включених батарейок» (“Batteries included”). Це означає, що Django поставляється з більшістю бібліотек та інструментів, включених у фреймворк, необхідних для звичайних випадків використання [4].

Django має наступні особливості:

- ORM (Object-relational mapping) – об’єктно-реляційне зображення, програмний інтерфейс доступу до бази даних з підтримкою транзакцій;
- архітектура ядра фреймворку – MVC (Model View Controller);

- вбудований інтерфейс адміністратора, з уже наявними перекладами на більшість мов;
- диспетчер URL на основі регулярних виразів;
- розширювана система шаблонів з тегами та наслідуванням;
- бібліотека юніт-тестів Python;
- система кешування;
- локалізація та інтернаціоналізація;
- авторизація та аутентифікація, підключення зовнішніх модулів аутентифікації: LDAP, OpenID тощо;
- бібліотека для роботи з формами (наслідування, побудова форм за існуючою моделлю БД);
- вбудована автоматична документація по тегам шаблонів та моделям даних, доступна через адміністративний застосунок;
- архітектура застосунків, що підключаються, можна встановлювати на будь-які Django-сайти [5].

Django ORM – це дуже потужний інструмент розробки проекту, який зв'язує проект з широко використовуваними базами даних, такими як MySQL, Oracle, SQLite і PostgreSQL, реалізуючи шаблон проектування ActiveRecord (активний запис). Він змішує в одному класі бізнес-логіку і службову логіку доступу до даних (бібліотека «django.db.models», клас «Model»).

На жаль цією особливістю Django порушує принцип єдиної відповідальності (single responsibility principle) – це принцип комп'ютерного програмування, в якому зазначається, що кожен модуль, клас або функція повинні відповідати за одну частину функціональності, що надається програмним забезпеченням, і ця відповідальність повинна бути повністю інкапсульована класом [6].

Цей фреймворк реалізує принцип «не повторюйтеся» (Don’t repeat yourself, DRY) – це принцип розробки програмного забезпечення, спрямований на зменшення повторення шаблонів програмного забезпечення, замінивши його абстракціями або використовуючи нормалізацію даних, щоб уникнути надмірності [7]. Коли цей принцип успішно застосовується, модифікація будь-якого окремого елемента системи не вимагає зміни інших логічно не пов’язаних елементів. Фреймворк якісно реалізує модульність, тому програмування з його допомогою дозволяє мінімізувати кількість логічних зв’яків.

Як зазначалося раніше, Flask розроблений на основі шаблону Jinja2. Як повноцінний рушій шаблонів для Python, Jinja2 розроблений на базі системи шаблонів Django. Вона дозволяє розробникам прискорити розробку динамічних веб-додатків, використовуючи переваги інтегрованого середовища виконання вбудованого програмного забезпечення та написання шаблонів на виразній мові. Django постачається з вбудованим механізмом шаблонів, який дозволяє розробникам визначати рівень користувача веб-програми. Вона навіть дозволяє розробникам прискорити розробку інтерфейсу користувача за допомогою написання шаблонів у мові шаблонів Django (Django Template Language) [15].

Вбудований інтерфейс адміністратора дозволяє розробникам розпочати створення веб-додатків без зовнішнього введення. Django дозволяє архітекторам системи розділити один проект на декілька програм.

Інструментами Django можливо розділити проект на декілька додатків. Таким чином, стає легше писати окремі програми, а також додавати функціональні можливості веб-додатка шляхом інтеграції додатків у проект. Невеликі програми допомагають розширювати та легше супроводжувати між сайтові компоненти.

Завдяки тому, що основні інструменти вбудовані у фреймворк, Django має високий рівень захисту проекту і допомагає уникнути багатьох

поширених помилок безпеки, таких як ін’єкції SQL, міжсайтові сценарії (cross-site scripting), підробку запитів між сайтами (cross-site request forgery) тощо. Система аутентифікації користувачів забезпечує безпечний спосіб керування обліковими записами користувачів і паролями [4].

Загалом, Django – це повний стековий фреймворк, його модулі попередньо зконфігуровані, що дозволяє швидше розробляти і використовувати як прості, так і складні веб-додатки з динамічним контентом (з урахуванням майбутнього масштабування).

Однією з позитивних сторін фреймворку є те, що Django можна запускати спільно з веб-серверами Apache, Nginx, використовуючи WSGI, Gunicorn або Cherokee, використовуючи flup (модуль Python) [10].

Django також можна встановити за допомогою pip, запустіть наступну команду:

```
Pip3 install Django --user
```

Після установки Django, нам потрібно запустити кілька скриптів Django для створення проекту, щоб створити додаток. Після установки Django, у вас також з'явиться команда django-admin, яку ми також зараз використовуємо. Запустіть наступний код:

```
Django-admin startproject hellodjango
```

Він створює новий проект Django і каталог hellodjango в тій локації, в якій ви запустили команду. Якщо ви погляньте на каталог hellodjango, ви побачите, що він створив файл manage.py і підкаталог, який також називається hellodjango. Усередині підкаталогу містяться три скрипта Python. Нас цікавить тільки urls.py для нашого проекту «Hello World».

Наступний крок, це використання Django для створення додатка, у якого організаційна структура нижче, ніж у проекту Django (один проект може містити кілька додатків). Ми використовуємо файл manage.py, який був створений попередньою командою, щоб створити додаток. Запустіть наступну команду з зовнішнього каталогу hellodjango:

```
python3 manage.py startapp helloworld
```

Вона створює додаток helloworld, і робить його частиною проекту hellodjango. Тепер нам потрібно налаштувати маршрутизацію URL (як ми робили з @ app.route в Flask). Так як проекти Django мають більше число налаштувань структури, ніж додатки Flask, нам потрібно зробити кілька додаткових кроків. Попередня команда створила каталог helloworld в зовнішньому каталозі hellodjango. Відкрийте автоматично створений файл helloworld / views.py і додайте наступний код:

```
from django.http import httpresponse
def index(request):
    return httpresponse("hello,world!")
```

Перший рядок імпортує функцію HttpResponse, яку ми можемо використовувати для відправки рядка над HTTP користувачеві нашого додатку. Як і з Flask, нам не потрібно використовувати це часто, так як нам по можливості краще уникати додаткового головного болю з рендерингом шаблонів HTML. У будь-якому випадку, це все, що нам потрібно для нашого застосування Hello World.

У третьому рядку ми визначаємо функцію індексу. Тут, на відміну від Flask, нам не потрібно використовувати декоратор, який буде говорити цієї функції, що її викликають, коли користувач відвідує наш додаток. Замість цього, ми виконаємо аналогічну настройку за допомогою двох файлів urls.py - один для проекту, який був створений автоматично, і ще один для додатка, яке нам потрібно створити.

Четвертий рядок видає повідомлення «Hello, World!» В загорнутої рядку в HttpResponse, так що воно може бути відображено в нашому веб браузері.

Тепреь нам потрібно створити файл urls.py для нашого застосування. Створіть helloworld / urls.py і додайте наступний код:

```
From Django.conf.urls import url
From . import views
```

```
Urpatterns = [
    url(r"^", views.index, name="index"),]
```

Перший рядок імпортує функцію url, так що ми можемо пов'язати певні посилання з функціями в нашому файлі views.py.

Третій рядок імпортує файл views.py, який ми додали нашим поданням index ().

Рядки 5-7 складають список url патернів - які є еквівалентом декораторам app.route, які ми використовували у Flask. Ми зіставляємо певні посилання, використовуючи регулярні вирази, і пов'язуємо їх з функціями в нашому скрипті views.py. В даному випадку, ми встановлюємо одиночний патерн, який відповідає порожньому URL (як і у випадку з «/» в Flask, іншими словами, сторінка за замовчуванням в нашому додатку) і зв'язується з функцією views.index, яку ми вписали раніше.

Це була настройка URL для нашого застосування (helloworld). Нам також потрібна настройка URL для нашого проекту hellodjango. Відредагуйте файл hellodjango / hellodjango / urls.py, який був створений автоматично (це може трохи заплутати: у нас є два файли urls.py, але в цьому є сенс, тому що один належить всьому проекту, маршрутізіруя URL в різні додатки, а другий належить тільки додатком helloworld) наступним чином:

```
From Django.conf.urls import include, url
Urlpatterns = [
     url(r '^hello/' , include('hellowold.urls')),
]
```

Це аналогічно попередній запис. Однак, замість маршрутизації посилань певного патерну для конкретного відображення, ми робимо це для конкретного додатка. У нашому випадку, це будь-який URL, що містить / hello який був відправлений в наш додаток helloworld, з подальшим переглядом helloworld.urls, щоб визначитися, яке зображення викликати.

Тепер повернемося до зовнішнього каталогу / hellodjango (той, в якому міститься файл manage.py) і запустимо його з наступною командою:

```
python3 manage.py runserver
```

Цей код запустить сервер розробки Django, який дасть нам відвідати наш додаток через localhost також, як ми робили це з Flask. Ви отримаєте видачу, на подобу цієї:

```
Python
1  Performing system checks...
2
3  System check identified no issues (0 silenced).
4
5  You have 13 unapplied migration(s). Your project may not work properly until
6  Run 'python manage.py migrate' to apply them.
7
8  March 03, 2017 - 16:14:20
9  Django version 1.10.5, using settings 'hellodjango.settings'
10 Starting development server at http://127.0.0.1:8000/
11 Quit the server with CONTROL-C.
```

Рис. 1.2. Старт сервера Django

Ви можете не звертати уваги на попередження про переміщення - це відноситься до бази даних веб додатки, яке ми не використовуємо. Десята рядок дуже важлива. У ній, як і в випадку з Flask, йдеться про те, який порт використовує сервер в даний момент. У нашому прикладі це 8000, так що ми можемо перейти по http://127.0.0.1:8000/hello, щоб побачити заповітну напис «Hello,World!» (Не забудьте про / hello в кінці, що вказує Django, яку програму потрібно відкрити).

Загалом, Django – це повний стековий фреймворк, його модулі попередньо зконфігуровані, що дозволяє швидше розробляти і використовувати як прості, так і складні веб-додатки з динамічним контентом (з урахуванням майбутнього масштабування).

Однією з позитивних сторін фреймворку є те, що Django можна запускати спільно з веб-серверами Apache, Nginx, використовуючи WSGI, Gunicorn або Cherokee, використовуючи flup (модуль Python) [10].

2.4 Переваги Python и JS

Python є найбільш швидко зростаючою мовою програмування за останні кілька років. Про це свідчать дослідження 2019 року.

Переваги Python

Граничне значення низького входу. Синтаксис Python більш зрозумілий для новачка. Логічно, лаконічно і зрозуміло. У порівнянні з багатьма іншими мовами, Python має легкий для читання синтаксис, за винятком того, що Visual Basic також легко. Наприклад, є анекдот про Perl, що це "тільки писати" мова, тому що синтаксис важко читати: https://en.wikipedia.org/wiki/Write-only_language. Крос-платформенний: підходить для різних платформ: Linux і Windows. Існує реалізація перекладачів для мобільних пристроїв і непопулярних систем. Широке застосування. Використовується для розробки веб-додатків, ігор, зручних для автоматизації, математичних обчислень, машинного навчання, в області Інтернет речей. Існує реалізація під назвою Micro Python, оптимізована для роботи на мікроконтролерах (можна писати інструкції, логіку взаємодії пристрою, організовувати комунікацію, реалізувати розумний будинок). Сильна спільнота та багато конференцій. Наприклад, нещодавно в Одесі відбувся PyCon. Чотири зарубіжні спікери виступили на конференції, яка торкнулася цікавих тем. Особисто мені було корисно почути співавтор бібліотеки Mindскб, спікер з Нью-Йорка (розмовляв про штучний інтелект в розробці інструментів розпізнавання обличчя на

фото), спікер британської компанії еластичний (виступив з розмови на «інструментах моніторингу виконання вашої програми»), спікер з Мінська (дівчинка говорила про протокол MTT для зв'язку Інтернет-пристроїв речей). Сильна підтримка гігантів індустрії IT. Такі компанії, як Google, Facebook, Dropbox, Spotify, zuora, Netflix, на певних етапах розвитку використовували Python. Високий попит на ринку праці. Є багато бібліотек якості в світі Python, так що вам не доведеться винаходити колесо, якщо вам потрібно терміново вирішити деякі комерційні проблеми. Є багато інтелектуальних книг для навчання, в першу чергу англійською мовою, звичайно, але переклад також містить гідну літературу. Сьогодні є багато підручників на YouTube: відео-блоги, семінари та конференції. Python має суворі вимоги до написання коду (вимагає відступу).

JavaScript (JS)- є прото-продукт мова, яка проорієнтована. Вона відображає мову **ECMAScript**, прототипом якої була спочатку. Перша варіація з'явилася в 1995 і постійно покращилася з тих пір, поки вона не прийшла до свого нинішнього виду.

Мова JavaScript використовується для:

Можливість змінювати сторінки браузера

- Додавання або видалення позначок
- Зміна стилів сторінки
- Інформація про дії користувача на сторінці
- Запит доступу до випадкової частини вихідного коду сторінки
- Внесення змін до цього коду

Переваги JavaScript

- Жоден сучасний браузер не обходиться без підтримки JavaScript
- Навіть не фахівець може обробляти плагіни та скрипти, написані на JavaScript
- Корисні функціональні налаштування
- Постійно вдосконалюючи мову в даний час розробляється бета-варіацією проекту, JavaScript2
- Взаємодія з додатком можна навіть зробити за допомогою текстових редакторів, таких як Microsoft Office і Open Office
- Перспектива використання мови в програмуванні та навчанні информатики

2.5 Авторизація завдяки Django

Система аутентифікації Django відповідає за обидва аспекти: аутентифікацію і авторизацію. Якщо коротко, то аутентифікація перевіряє користувача, а авторизація визначає, що аутентіфіцированний користувач може робити. Далі термін "аутентифікація" буде використовуватися для позначення обох аспектів.

Система аутентифікації складається з:

• Всього учасників

• Прав: Бінарні (так / ні) прапори, що визначають наявність у користувача права виконувати певні дії.

• Груп: Загальний спосіб призначення міток і прав на безліч користувачів.

• налаштовувати системи хеширования паролів

• Інструментів для форм і уявлень для аутентифікації користувачів або для обмеження доступу до контенту

• Системи плагінів

Аутентифікаційних система Django намагається бути дуже простий і не надає деякі фічі, поширені в інших системах веб аутентифікації. Такі фичи реалізовані в сторонніх пакетах:

• Перевірка складності пароля

• Обмеження спроб входу

• Аутентифікація через сторонні сервіси (OAuth, наприклад)

Підтримка аутентіфікації скомпонована у виде модуля в django.contrib.auth. За замовчуванню, необхідні настройки Вже включені в settings.py, створюваній с помощью командіровку django-admin startproject, и являються собою два записи в параметрі конфігурації INSTALLED_APPS:

'Django.contrib.auth' містіть ядро системи аутентіфікації и ее Стандартні моделі.

'Django.contrib.contenttypes' є фреймворком тіпів, який дозволяє створювати вами моделі.

І записи в параметрі конфігурації MIDDLEWARE_CLASSES:

SessionMiddleware управляє сесіямі за часом запиту.

AuthenticationMiddleware асоціює користувачів із запит с помощью сесій.

SessionAuthenticationMiddleware разлогініт користувача з усіх его сесій, если ВІН поміняє пароль.

При наявності ціх налаштування,! Застосування командіровку manage.py migrate створює в базі Даних необхідні для системи аутентіфікації табліці, створює права для будь-якіх моделей всех зареєстрованіх додатків.

2.6 Використання SVG графіки для створення 3D моделі

У розділі поселення можна продивитися план гуртожитків розробленому за допомогою Scalable Vector Graphics (SVG – формат векторної графіки, специфікація мови розмітки, що базується на XML, Extensible Markup Language (розширювана мова розмітки)). Векторна графіка є зображеннями комп'ютерної графіки, які визначаються точками, які з'єднані лініями і кривими, описаними функціями. Тож існує можливість збільшити будь-яку частину зображення SVG без втрати якості. Додатково, до елементів SVG документу можливо застосовувати фільтри — спеціальні модифікатори для створення ефектів, подібних вживаним при обробці растрових зображень (розмиття, витискування, складні системи трансформації тощо). В тексті SVG-коду фільтри описуються тегами, візуалізацію яких забезпечує засіб перегляду, що не впливає на розмір початкового файлу.

SVG надає всі переваги XML:

- Можливість роботи в різних середовищах.
- Інтернаціоналізація (підтримка Unicode).
- Широка доступність для різних застосувань.

SVG має також такі особливості:

- Малий розмір: об'єкти SVG важать менше растрових зображень.
- Зменшення HTTP-запитів: при використанні SVG скорочується кількість звернень до сервера, відповідно збільшується швидкість завантаження сайту.
- Анімація і редагування: за допомогою мов програмування можна анімувати SVG, а також редагувати в текстовому або графічному редакторі.
- За допомогою CSS можна змінювати параметри графіки на сайті, наприклад фон, прозорість або границі [16].

2.7 Налаштування бази даних та створення моделей представлення даних

Для налаштування бази даних використовуватимемо PostgreSQL, для цього ми встановили компонент Psycopg2 та sqlparse, що додається разом із ним. Sqlparse забезпечує підтримку розбору, розбиття і форматування SQL-операторів. Щоб використати цю базу даних необхідно спочатку встановити сервер PostgreSQL на комп'ютер. Після цього відкриваємо консоль (командний рядок): у операційній системі Windows запускаємо командний рядок системи (cmd.exe) і виконуємо команду

psql –U postgres,

після цього вводимо пароль; у операційній системі Linux запускаємо термінал і виконуємо команду

sudo –u postgres psql.

Згодом, створюємо користувача для бази даних сайту:

create user db_user,

вводимо пароль для цього користувача і створюємо базу даних командою

create database ksutemp_db owner db_user

Останнім кроком заходимо у файл settings.py і змінюємо запис у налаштуваннях бази даних. PostgreSQL має багато переваг над стандартною базою даних Sqlite для Django (див. додаток А).

Створимо власне свій проект з назвою app командою

python manage.py startapp app.

Тепер створимо моделі за допомогою ORM фреймворку Django. Кожна модель є класом, який унаслідується від класу django.db.models.Model, в якому реалізовані методи для роботи з базою даних. Модель містить набір полів і поведінку даних, які ми зберігаємо. Також Django дотримується принципу DRY, тобто кожну окрему концепцію або частину даних слід зберігати лише в одному місці.

Створимо модель NewTempPlace (нахождение датчика), яка містить такі відомості:

- корпус
- поверх;
- аудіторія;
- час оновлення

Кожне поле моделі визначається відповідним екземпляром класу Field: PositiveIntegerField для невід'ємних цілих чисел, CharField для символьних полів, TextField для текстів, DateTimeField для дати і часу, ImageField для зображень, шлях до яких треба зберігати. Ці класи вказують Django, які типи зберігатимуться в базі даних. Назви полів використовуватимуться для опрацювання даних в бізнес-логіці та як назви колонок в базі даних. Перший параметр конструктора класу Field – це назва, яка буде відображатися при роботі з моделлю в панелі адміністрації Django. Підклас Meta – це просто контейнер класу з деякими параметрами (метаданими), прикріпленими до моделі. Він визначає такі речі, як доступні права, пов'язане ім'я таблиці бази даних, чи є модель абстрактною чи ні, визначає назву моделі в однині та множині тощо.

Для зручного зображення об'єкту створеної моделі додамо метод __str__(), що при звертанні до цього об'єкту нам відображався заголовок новини.

Клас ImageField успадковує всі атрибути і методи з класу FileField, але також перевіряє, що завантажений об'єкт є дійсним зображенням. Значення поля буде додано до вашого шляху MEDIA_ROOT у файлі settings.py, щоб сформувати розташування на локальній файловій системі, де будуть зберігатися завантажені файли.

Додатково створимо функцію ззовні класу NewsTemp, яка отримує сигнал після видалення запису (новини) з бази даних і видаляє файл зображення, пов'язаного з записом.

```
From django.db.models.signals import post_delete

@receiver(post_delete, sender= NewTempPlace)
def NewTempPlace _delete(sender, instance, **kwargs):
    instance.photo.delete(save=False)
```

Django включає в себе «диспетчер сигналів», який допомагає дозволити логічно роз'єднаним компонентам додатка отримувати повідомлення про те, що дії відбуваються в іншому місці програми фреймворка. В даному випадку функція NewTempPlace_delete() спрацює тоді, коли об'єкт класу (sender= NewTempPlace) сповістить про своє видалення (post_delete).

Для того, щоб у базі даних з'явилася таблиця NewTempPlace необхідно виконати команду

python manage.py makemigrations,

яка створить команди для створення необхідних таблиць. У файлі міграції спостерігаємо наступні записи:

```
from django.db import migrations
class Migration(migrations.Migration):
    dependencies = [
        ('app', '0008_auto_20190221_0845'),
    ]
    operations = [
                migrations.AlterModelOptions(
            name='newspost',
            options={'verbose_name': 'Температура',
'verbose_name_plural': 'Температура'},]
```

Для того, щоб запустити у дію зроблені міграції, виконаємо команду

python manage.py migrate.

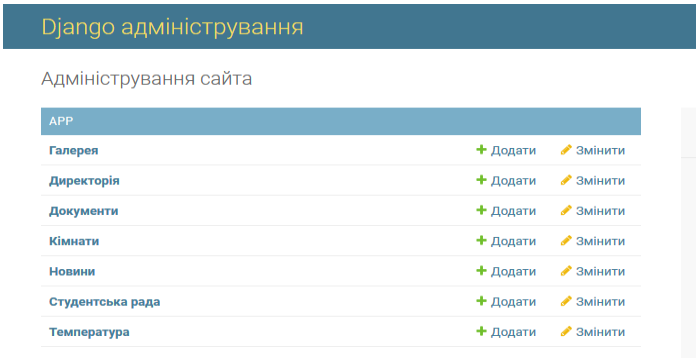


Рис. 2.1. Створені таблиці у базі даних

Після того, як додамо усі необхідні класи (див. додаток В) і зареєструємо їх в панелі адміністрації Django спостерігатимемо перелік доступних таблиць бази даних (див. рис. 2.2).

2.8 Порівнняння апаратних компонентів та обрання

В даному пункті ми розглянемо декілька датчиків та порівняємо їх для обрання найбільш відповідного для вирішення поставленого питання. На ринку ми можемо зустріти декілька схожих рішень, які підходять для цього проекту. Для порівняння ми використаймо таблицю.

Назва датчику	Точність вимірювання температури,°C	Точність вимірювання вологості,%	Загальна вартість, грн
SHT20	±1	±5-7	160
AM2302	±1	±7	80
LM35	±1,5	±8	200
CH-S01	±0,5	±5	700

У таблиці з верху порівнюється декілька різних датчиків завдяки цьому був обраний датчик, який най краще нам підходе. А саме AM2302 цей датчик має дуже просте підключення, та дуже простий протокол для ведення моніторингу та спілкуванням з сервером. В датчику використовується протокол HTTPS, тобто датчик відправляє на сервер звичайний текс, який ми завдяки JS парсингу перетворюємо на дані, які записуються до бази даних. А саме підключення до серверу відбувається завдяки звичайного WI-FI.

РОЗДІЛ 3 Розробка веб-додатку для спостереження температури

3.1 Обрання серверу для розгорнення проекту

С початку розробки було обрано власні можливості підтримки серверної частини. Перша частина створення віртуального серверу була дуже проста, це встановлення будь якого дистрибутиву Linux за допомогою Parallels Desktop або VirtualBox. Parallels Desktop було обрано бо цей продукт має власні бібліотеки віртуалізації та відображення графіки. Також має майже апаратну швидкість. Але це рішення призвело до іншої проблеми яку потрібно було вирішити. А саме оновлення даних та моніторингу стану датчиків коли віртуальний сервер був вимкнутий. Її тому було прийнято рішення о пошуку хостингу, було розглянемо декілька варіантів. Перше, це використати безкоштовний або платний хостингом. Але на томість було вирішено використати потужності університету, а саме один із багатьох серверів. Імя цього серверу не буде вказано.

3.2 Розгорнення проекту

Даному розділі ми розглянемо, які дії були зроблені для розгорнення проекту.

По перше нам знадобилося знайти відповідні датчики для вирішення даної теми. Для цього було вирішено обрати “asair am2302”, які буде використовуватися для замірювання температури. На далі ми займемося підключенням цих датчиків, було використано простіше рішення. Підключення до ліній 220 вольт в університеті, таким чино нам не знадобиться постійно слідкувати за потужністю акумуляторів.

Наступним кроком було розгорнення індивідуальної сеті Wi-Fi, для вирішення цього питання знадобилась допомога працівників університету. Встановлення додатку пройшло дуже добре, проте було декілко помилок.

Так як додаток розроблявся на локальній машині, на якою було встановлена інша операційна система, знадобилося встановити декілька бібліотек та трішки виправити код. Для початку міграції сайту треба створити проект на основі фреймворку. Для цього створимо віртуальне середовище. Python, як і більшість інших сучасних мов програмування, має власний, унікальний спосіб завантаження, зберігання модулів. Головне завдання віртуального середовища – створення ізольованого середовища для проектів Python, що в свою чергу, буде залежати від певного проекту. Цей метод в Python 3 володіє перевагою, яке змушує використовувати специфічну версію інтерпретатора Python 3, який використовуватиметься для створення віртуального середовища. Таким чином, ви уникаєте непорозумінь при з'ясуванні, яка інсталяція Python базується в новій віртуальному середовищі і які компоненти були чи будуть встановлені як додаток до інтерпретатора.

Перебуваючи в каталозі проекту в консолі створюємо віртуальне середовище (virtual environment, venv):

python3 –m venv

Або робимо це за допомогою засобів середовища програмування

Тепер у каталозі з'явилась папка venv, у якій знаходяться:

- bin – файли, які взаємодіють з віртуальним середовищем;
- include – C-заголовки, компілюються пакети Python;
- lib – копія версії Python разом з папкою «site-packages», в якій встановлена кожна залежність.

Але так як ці ресурси ізольовані в середовищі, потрібно їх активувати. Для цього використаємо наступну команду

source env/bin/activate

Тепер можна побачити в консолі наступне

Якщо продивитися структуру інтерпретатора проекту ми знайдемо два пакети: setuptools та pip. Setuptools – це бібліотека процесів розробки

пакетів, призначена для полегшення упаковки проектів Python шляхом розширення стандартної бібліотеки інтерпретатора. Ця бібліотека включає в себе:

- визначення пакета і модуля Python;
- метадані пакета розподілу (назва та версія);
- встановлення проекту;
- деталі, специфічні для платформи;
- тестування встановлених розширень [14].

Pip – це система управління пакетами, яка використовується для встановлення і управління програмними пакетами, запрограмованими мовою Python. За допомогою цього розширення можливо додати нову бібліотеку до віртуального середовища командою

pip install назва пакету [13].

Це і зробимо з Django:

pip install django

Для сайту ще знадобляться такі розширення:

- Markdown – легка мова розмітки з синтаксисом форматування звичайного тексту. Її конструкція дозволяє перетворити її на безліч форматів виводу, з яких нам знадобиться HTML (*pip install markdown*).
- Pillow (Python Imaging Library) – бібліотека мови Python, призначена для роботи з растровою графікою (*pip install pillow*).
- Psycopg2 – найпопулярніший адаптер бази даних PostgreSQL для мови програмування Python (*pip install psycopg2*).
- Pygments – це пакет підсвічування синтаксису, написаний на мові Python (*pip install pygments*).

Тепер продивимося структуру нашого віртуального середовища, де всі компоненти були успішно встановлені

За допомогою пакету pip можемо зробити файл requirements.txt (вимоги), який міститиме інформацію про встановлені розширення: назву та версію. Це необхідно виконати з наступних причин:

•При перенесенні проекту на інший комп'ютер не має потреби копіювати віртуальне середовище.

•При втраті віртуального середовища легко відновити всі компоненти.

Створимо файл requirements.txt консольною командою

```
pip freeze > requirements.txt,
```

відновити всі компоненти можна, виконавши команду

```
pip install –r requirements.txt.
```

Після встановленя додатку на сервер, було проведенно налаштування: створення адміністратора, добавлення роз положення датчиків а також оновлення системи підключення датчиків. Після усіх цих дії та виправлень додаток почав функціонувати у тестовому режимі. В дальшому буде проведені декілька стрес тестів для знаходження помилок у роботі додатку або датчиків.

На разі додаток має майже весь робочий функціонал, окрім зображення 3D модель, E-mail розсилки коли показники не відповідають нормі або не відповідають на запит серверу.

Висновки

В рамках даного проекту було досліджено особливості роботи з фреймворком та розроблено сайту моніторингу температурних норм. Досліджено особливості різних видів фреймворків, які поділені на типи за мовою програмування. Визначено, що вибір фреймворку специфічний для кожного проекту і залежить від поставлених завдань. Ми обрали Django, оскільки він зручний та гнучкий в роботі.

Було описано використання Django в проекті. Сам проект складено з окремих частин (додатків, модулів, пакетів тощо), що дає йому гнучкість в роботі і продуктивність у використанні коду. Суть структури додатка проекту на Django зводиться до того, що вона працює частково за образом MVC, але істотно від відрізняються моделі в першу чергу тим, що контролери в MVC – це URL Django.

Сформована структура сайту відповідно до потреб цільової аудиторії сайту. Вона вміщує сторінки, розділи, підрозділи, навігацію (посилання), розташування цих елементів і їх зв’язок між собою. Продумана і зручна структура допомагає нам значно скоротити витрати на збір інформативної складової сайту та уникнути великої кількості помилок на етапі розробки.

Створений сайт моніторингу температурних норм з використанням Django Framework. У роботі детально описані кроки розробки проекту.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Review of faramework terminology [Електронний ресурс]. – Режим доступу: URL: https://riehle.org/computer-science/research/dissertation/chapter-2.html

2. Фреймворки у веб-розробці. [Електронний ресурс]. – Режим доступу: URL: https://web-creator.ru/articles/about_frameworks.

3. Rakesh Vidya Chandra, Bala Subrahmanyam Varanasi (2015). Python Requests Essentials. ISBN 978-1-78439-541-4

4. Чому використовувати Django [Електронний ресурс]. – Режим доступу: URL: https://djangostars.com/blog/why-we-use-django-framework

5. Офіційна документація Django (веб-фреймворк) [Електронний ресурс]. – Режим доступу: URL: https://docs.djangoproject.com/en/dev/faq/general/

6. Foote, Steven (2014). Learning to Program. Addison-Wesley Professional. p. 336. ISBN 9780133795226.

7. Martin, Robert C. (2003). Agile Software Development, Principles, Patterns, and Practices (ISBN 978-0135974445)

8. Офіційна документація Flask [Електронний ресурс]. – Режим доступу: URL: http://flask.pocoo.org/docs/1.0/foreword/#what-does-micro-mean

9. Shalabh Aggarwal (2014). Flask Framework Cookbook. ISBN 978-1-78398-340-7

10. Налаштування сервера Django на Cherokee [Електронний ресурс]. – Режим доступу: URL: http://cherokee-project.com/doc/cookbook_django.html

11. Налаштування сервера Django, використовуючи WSGI [Електронний ресурс]. – Режим доступу: URL: https://docs.djangoproject.com/en/dev/howto/deployment/wsgi/

12. Miguel Grinderg (2018). Flask Web Development. ISBN 978-1-491-99173-2

13. Документація додатку «pip» [Електронний ресурс]. – Режим доступу: URL: https://pypi.org/project/pip/

14. Документація додатку «setuptools» [Електронний ресурс]. – Режим доступу: URL: https://pypi.org/project/setuptools/

15. Офіційна документація шаблонізатора Jinja2 [Електронний ресурс]. – Режим доступу: URL: http://jinja.pocoo.org/

Офіційна документація SVG формату [Електронний ресурс]. – Режим доступу: URL: https://www.w3.org/TR/SVG/intro.html

ДОДАТКИ

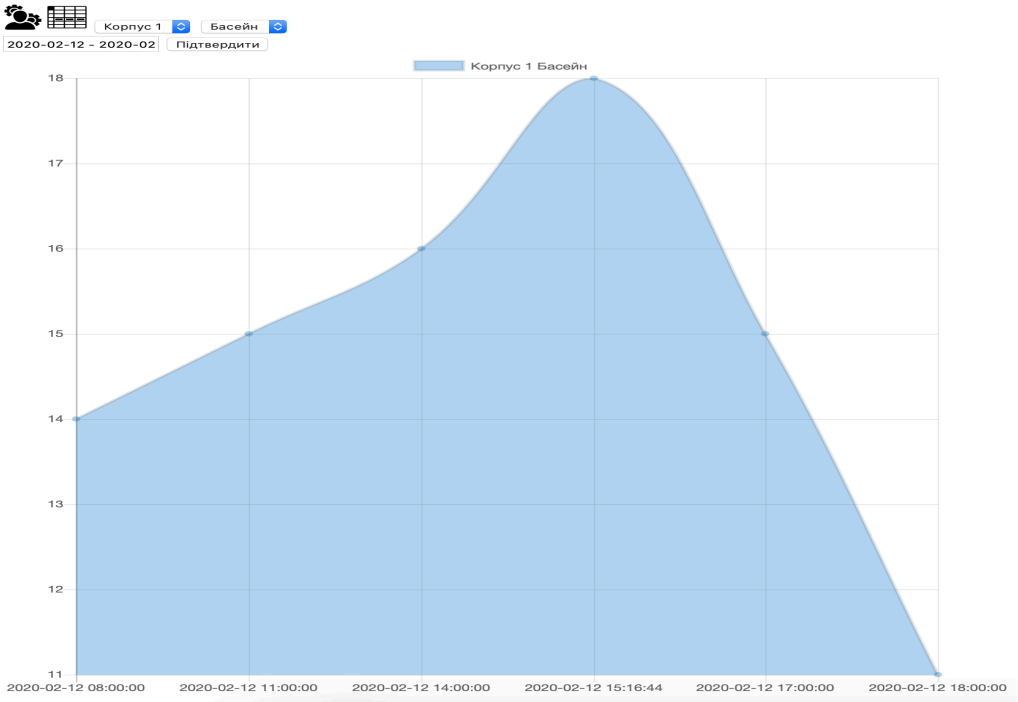
Додаток А.

Табличний метод відображення даних

Дата		12.02.2020								
Час		15:16:44	17:49:37	22:37:04	12:00:00	08:00:00	11:00:00	14:00:00	17:00:00	18:00:00
Корпус 1	Басейн	18				14	15	16	15	11

Додаток Б.

Графічний метод відображення даних



КОДЕКС АКАДЕМІЧНОЇ ДОБРОЧЕСНОСТІ
ЗДОБУВАЧА ВИЩОЇ ОСВІТИ ХЕРСОНСЬКОГО
ДЕРЖАВНОГО УНІВЕРСИТЕТУ

Я, Шкворець Владислав Владленович, учасник(ця) освітнього процесу Херсонського державного університету, **УСВІДОМЛЮЮ**, що академічна доброчесність – це фундаментальна етична цінність усієї академічної спільноти світу.

ЗАЯВЛЯЮ, що у своїй освітній і науковій діяльності **ЗОБОВ'ЯЗУЮСЯ**:

– дотримуватися:

- вимог законодавства України та внутрішніх нормативних документів університету, зокрема Статуту Університету;
- принципів та правил академічної доброчесності;
- нульової толерантності до академічного плагіату;
- моральних норм та правил етичної поведінки;
- толерантного ставлення до інших;
- дотримуватися високого рівня культури спілкування;

– надавати згоду на:

- безпосередню перевірку курсових, кваліфікаційних робіт тощо на ознаки наявності академічного плагіату за допомогою спеціалізованих програмних продуктів;
- оброблення, збереження й розміщення кваліфікаційних робіт у відкритому доступі в інституційному репозитарії;
- використання робіт для перевірки на ознаки наявності академічного плагіату в інших роботах виключно з метою виявлення можливих ознак академічного плагіату;

– самостійно виконувати навчальні завдання, завдання поточного й підсумкового контролю результатів навчання;

– надавати достовірну інформацію щодо результатів власної навчальної (наукової, творчої) діяльності, використаних методик досліджень та джерел інформації;

– не використовувати результати досліджень інших авторів без використання покликань на їхню роботу;

– своєю діяльністю сприяти збереженню та примноженню традицій університету, формуванню його позитивного іміджу;

– не чинити правопорушень і не сприяти їхньому скоєнню іншими особами;

– підтримувати атмосферу довіри, взаємної відповідальності та співпраці в освітньому середовищі;

– поважати честь, гідність та особисту недоторканність особи, незважаючи на її стать, вік, матеріальний стан, соціальне становище, расову належність, релігійні й політичні переконання;

– не дискримінувати людей на підставі академічного статусу, а також за національною, расовою, статевою чи іншою належністю;

– відповідально ставитися до своїх обов'язків, вчасно та сумлінно виконувати необхідні навчальні та науково-дослідницькі завдання;

– запобігати виникненню у своїй діяльності конфлікту інтересів, зокрема не використовувати службових і родинних зв'язків з метою отримання нечесної переваги в навчальній, науковій і трудовій діяльності;

– не брати участі в будь-якій діяльності, пов'язаній із обманом, нечесністю, списуванням, фабрикацією;

– не підроблювати документи;

– не поширювати неправдиву та компрометуючу інформацію про інших здобувачів вищої освіти, викладачів і співробітників;

– не отримувати і не пропонувати винагород за несправедливе отримання будь-яких переваг або здійснення впливу на зміну отриманої академічної оцінки;

– не залякувати й не проявляти агресії та насильства проти інших, сексуальні домагання;

– не завдавати шкоди матеріальним цінностям, матеріально-технічній базі університету та особистій власності інших студентів та/або працівників;

– не використовувати без дозволу ректорату (деканату) символіки університету в заходах, не пов'язаних з діяльністю університету;

– не здійснювати і не заохочувати будь-яких спроб, спрямованих на те, щоб за допомогою нечесних і негідних методів досягати власних корисних цілей;

– не завдавати загрози власному здоров'ю або безпеці іншим студентам та/або працівникам.

УСВІДОМЛЮЮ, що відповідно до чинного законодавства у разі недотримання Кодексу академічної доброчесності буду нести академічну та/або інші види відповідальності й до мене можуть бути застосовані заходи дисциплінарного характеру за порушення принципів академічної доброчесності.

23.04.2020		
(дата)	(підпис)	(ім'я, прізвище)