МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХЕРСОНСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
Факультет комп'ютерних наук, фізики та математики
Кафедра інформатики, програмної інженерії та економічної кібернетики

СТВОРЕННЯ ПРОГРАМНОЇ СИСТЕМИ НАВЧАЛЬНОГО ПРИЗНАЧЕННЯ З ТЕМИ «НАБЛИЖЕНІ МЕТОДИ ІНТЕРПОЛЯЦІЇ, ЕКСТРАПОЛЯЦІЇ ТА АПРОКСИМАЦІЇ»

Кваліфікаційна робота (проєкт)
на здобуття ступеня вищої освіти "магістр"

Виконав: здобувач 2 курсу, 261М групи
Спеціальності 126 Інформаційні системи та технології
Освітньо-професійної програми
«Інформаційні системи та технології»
другого (магістрського) рівня освіти
Шишман Владислав Володимирович
Керівник: доктор фізико-математичних наук, професор
Львов Михайло Сергійович
Рецензент: Senior Software Engineer (EPAM) Кожевніков Дмитро Іванович

Херсон – 2021

ЗМІСТ

ВСТУП

В наш час, техніка та наука, в цілому, рухається дуже швидко, людство продовжує освоювати все нові і нові галузі. Виникло багато нових і досить складних проблем з кібернетики, механіки, електроніки. Для їх вирішення яких потрібні нові методи.

Зустрічаються задачі, точний розв’язок яких отримати неможливо і не потрібно. В такому випадку використовуються наближені методи вирішення. Застосування сучасної обчислювальної техніки неможливе без вмілого застосування чисельних методів.

Обчислювальна техніка сьогодення є потужним засобом для фактичного виконання обчислень. Завдяки цьому в багатьох випадках стало можливим відмовитись від наближеного трактування прикладних задач і перейти до їх вирішення в точній постановці. Застосування сучасної обчислювальної техніки неможливе без вмілого застосування чисельних методів.

Математичне моделювання можна розглядати як засіб вивчення реальної системи шляхом заміни її більш зручною для експериментального дослідження системи (моделлю), яка зберігає суттєві риси оригіналу[1].

Воно включає такі етапи: дослідження об'єкта і створення його математичного опису; побудова алгоритму, який моделює поведінку об'єкта; перевірка адекватності моделі і об'єкта; використання моделі.

Ефективність математичного моделювання в більшості визначається дією використовуваних для розрахунків моделей обчислювальних методів і алгоритмів.

На сьогоднішні день суспільство потребує фахівців високої кваліфікації. Для підготовки інженерів необхідний час і значні ресурси

техніки. Є статистика, згідно з якою більше половини трафіку приходиться на мобільні пристрої.

Разом з тим по країнах ОЕСР, близько 48% студентів повідомили, що вони виконують домашнє завдання за допомогою комп'ютера, 38%-спілкуються зі своїми однолітками про шкільні роботи за допомогою електронної пошти, а 33%-обмінюються матеріалами через комп'ютер (ОЕСР, 2014).

В 2019 році хворобу Ковід 19 було названо пандемією. Це призвело до обмежень на соціальні активності, і навчання в цілому. Було проведено дослідження згідно якого порівнювалося дві категорії студентів. Одна група здобувала знання віч на віч, інша дистаційно.

Незважаючи на те, що обидві групи досягли успіхів, є різниця у порівнянні їх готовності до роботи. Студенти першої групи досягли кращих результатів роботи, ніж їх аналоги. Показники в академії та готовність до роботи не пропорційно пов'язані. Ці свідчать про те, що вища освіта, як правило, не така активна з точки зору ринку праці, в той час як онлайн-навчання фактично зробило освіту набагато пасивнішою.

З огляду на вищесказане напрям потребує дослідження.

Мета: Створити програмну систему для навчання з тем інтерполяції, екстраполяції та апроксимації

Завдання, потрібні для досягнення поставленої мети:

1. Вивчити методи Ньютона, Лагранджа, Найменших квадратів, Бі сплайнів[2].
2. Дослідити технології функціонування веб-сервісів.
3. Розробити програмний продукт.
4. Опублікувати в мережі.

Об'єктом роботи є математичні методи, програмні засоби, що необхідні для розробки систем дистанційного навчання.

Предмет: Методи чисельних методів математичного аналізу, принципи роботи веб-сервісів.

В роботі було проведено аналіз існуючих аналогів, робота з предметною областю. Після чого були вияснені вимоги і здійснювалася розрбока програмного продукту. Опісля його розгортання на хмарному сервері.

Досліджуючи дану проблему було виявлено що доцільно буде використовувати кеш Redis для роботи системи при високому навантаженні. Поєднуючи його з оновленням даних через AJAX і постійним зберіганням структурних частин в базі даних(MySQL/Postgre)[9].

Як результат систему можна використовувати для навчання в аудиторії, так і в дистанційному режимі між студентами класу.

Робота прошла апробацію в журналі Магістерські Студії.

РОЗДІЛ 1
АНАЛІЗ НАЯВНИХ СИСТЕМ ТА ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Пропрієтарні додатки

Встановлюємі системи існують певний час і харатеризуються певними плюсами:

1. Maple може працювати як з аналітичними, так і чисельними методами, апроксимації і лінійної оптимізації (симплекс метод). Візуалізація будується плавно для трьох вимірів. Робота з ним можлива в режимі інтерактивного діалогу, і шляхом складання та налагодження програм на спеціальній мові.

Maple пропонує найпотужніші в світі системи розв'язку диференціальних рівнянь. Наприклад, рівняння Ейлера-Бернуллі-диференціальне рівняння 4-го порядку - для моделювання вигину балки. Помимо вирішення було розписано в формулах, типу LaTeX[11].

Робочий простір, складається з окремих областей, схожих на комірки для запуску коду. На початку роботи можуть бути відображені результати попередньої сессії.

Maple містить інформацію про константи, одиниці вимірювання, систему перерахунку формул. Зручний інтерфейс редагування коду, якість формул[4]. Роботу можна вигрузити в формати: HTML, rtf, та ін.

2. Matcad використовується інженерами для побудови моделей. Має обробку не підготовлених даних, обмеження доступу, API для взаємодії з іншими додатками в системі. Потрібно зазначити, що програма може відображати дані паралельно введенню: додавання даних обчислення поліному Ньютона відображалося графічно без перезапуску.

Основна відмінність Mathcadвід аналогічних програм це наявність графічного, а не текстового режиму введення формул[3]. Обчислення можуть здійснюватися декількома шляхами:

- ліниво, за бажанням користувача або миттєво;
- одночасно з набором;
- за командою.

Формули рахуються зверху вниз та зліва направо, як при читанні тексту. Потрібно зауважити що Mathcad може використовувати в розрахунках розмірності величин. Причому можна вибирати різні системи одиниць. В свою чергу це значно спрощує відстеження помилок в розрахунках, особливо в фізичних та інженерних.

Є можливість використання зовнішніх документів, підготовлених в сторонніх програмах.

Має добре розвинену кастомізацію, різні режими і способи відображення траєкторій (у вигляді ліній і точок різних типів, діаграм, суцільних і каркасних поверхонь і т. д.), осей координат, сіток, підсвічування, заголовків і ін.

3. Matlab дозволяє працювати з машинним навчанням. Також виділяється поміж попередніх тим, що може обмінюватися розрахунками з хмарою, Parallel Server. Таким чином, економляться системні ресурси і можна робити обчислення з великими об’ємами даних.

Має користувацький інтерфейс, майже для кожного інструменту. Тим самим доступний для користувачів різних рівнів підготовки.

Може взаємодіяти з програмами/бібліотеками написаними C і C++, компіляцію завдяки Mathworks. Є певне обмеження на переносимість коду. Версія Matlab Component Runtime(MCR) має співпадати, оновлення кожні 6 місяців. Реалізація математичних алгоритмів пропрієтарна.

4. Mathematica була задумана як система, що максимально автоматизовувала праці математиків-аналітиків та наукових працівників.

Вона працює в режимі постійного діалогу з користувачем[12]. В програмі є вікна з кнопками. Це може бути символ, ряд функцій чи алгебраїчне перетворення. При натисканні останній переносяться в робочий документ на вказане курсором місце.

Як і в Mapple велика увага приділяється графіці, у тому числі динамічній, є можливості інтеграіції мультимедіа - підсилюється динамічна анімація та синтез звуків.

Помимо чисельних методів, розглянуті вище програмні продукти можуть бути використані при вивченні інших дисциплін, таких як геометрія, теорія імовірності, фізика, статистика[10].

1.2 Предметна область

Основна ідея застосування інтерполяції полягає в тому, що для подальшої побудови поліноміальної кривої з групи точок необхідно, щоб вона була на одну одиницю більше ступеня многочлена, що підлягає підрахунку. Наприклад: для прямої другої, квадратична параболи - третьої. Розглянуті далі методи перші два методи базуються на цій ідеї.

1. Метод Ньютона. Застосовуються коли крок[h], відстань між x є константою. В такому випадку формула для інтерполяції бути мати вигляд:

$$f(x) = b_0 + b_1(x - x_0) + b2(x - x_0)(x - x_1) + \cdots + bn(x - x_0)(x - x_1) \cdots (x - x_n) \quad (1.1)$$

b_i, коефіцієнти полінома висловити через відомі величини x_0, крок, n, y_i (i = 0, 1,...n).

Особливістю полінома Ньютона є те, що коефіцієнти b_i можуть бути визначені за допомогою дуже простої математичної процедури.

Оскільки многочлен проходить через кожну точку даних, для точки площини (x_i, y_i), значення функції f (x_i) = y_i, таким чином, ми маємо, що f (x_0) = b_0 = y_0

Значення функції в точці x_1 буде рахуватися за формулою b_0 + $b_1(x_1 - x_0) = y_1$.

b_0 ми знаємо, виразивши звідси b_1 отримаємо дріб в чисельнику якого буде різниця перших двух значень y, в знаменнику x.

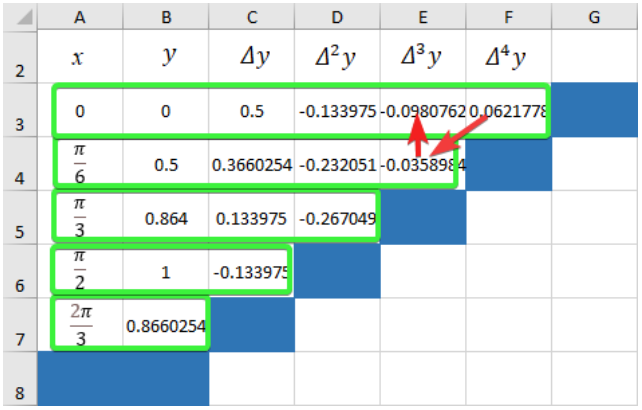
Якщо знову додаємо точку, то маємо рахувати b_2 матиме вигляд в чисельнику розрахунок для коефіцієнту b_1 мінус той що b_0. Внизу дробу буде різниця аргументів другої і першої точки:

$$b_2 = \frac{\frac{y_2 - y_1}{x_2 - x_1} \quad \frac{y_1 - y_0}{x_1 - x_0}}{x_2 - x_0} \tag{1.2}$$

При додаванні координати (x3, y3), для розрахунку b_3 в чисельнику потрібно буде рахувати дроби більшого розміру, відповідно b_2 і b_1. і тд. Отриманий шаблон розрахунку називають кінцевою різницею.

Перевага використання цього методу полягає в тому, що після визначення коефіцієнтів додавання нових точок даних не змінить раніше розрахований коефіцієнт; нам потрібно тільки розрахувати більш високі відмінності таким же чином, на схемі сині прямокутники. Вся процедура знаходження цих коефіцієнтів може бути зведена в розділену таблицю різниць. Нижче показано приклад розрахунку для п’яти точок. Це значення функції sin x, для 20 градусів, на відрізку[0, 120].

Кожен елемент в таблиці може бути розрахований з використанням двох попередніх елементів (зліва), показано червоною стрілкою. Насправді ми можемо обчислити всі елементи і зберегти їх в діагональній матриці, тобто у вигляді матриці коефіцієнтів, у вигляді (Рис. 1.1).



	A	B	C	D	E	F	G
2	x	y	Δy	$\Delta^2 y$	$\Delta^3 y$	$\Delta^4 y$	
3	0	0	0.5	-0.133975	-0.0980762	0.0621778	
4	$\frac{\pi}{6}$	0.5	0.3660254	-0.232051	-0.0358984		
5	$\frac{\pi}{3}$	0.864	0.133975	-0.267049			
6	$\frac{\pi}{2}$	1	-0.133975				
7	$\frac{2\pi}{3}$	0.8660254					
8							

Рисунок 1.1 — Таблиця метод Ньютона

За збереження в цих даних в програмі будуть відповідати масиви, в [таблиці] виділені зеленим прямокутником.

Варто зазначити що є два різні записи формули. Розібраний вище є записом вперед. Було вичислено многочлен для n, значень функції у вузлах даної таблиці, які розташовані правіше аргументу x, для якого проводиться інтерполяція. Якщо ж аргумент x знаходиться ближче до кінця таблиці, наприклад, лежить в інтервалі (x_{n-1}, x_n), то побудувати поліном першого виду неможливо (за винятком лінійного). У подібних випадках слід замість нього використовувати другий інтерполяційний поліном Ньютона (або поліном для інтерполювання назад). Цей інтерполянт схожий, але його коефіцієнти виражаються через значення функції у вузлах, розташованих лівіше заданого аргументу x.

В наведеному прикладі при перевірці по двом формулам точність почала відрізнятися в шостому знакі. Оскільки точка розміщувалася ближче до початку перша формула показала себе краще.

2. Поліном Лагранжа, доцільно побудувати коли відстані між вузлами неоднакові[5]. Він дорівнює сумі добутків фактичних значень у на функцію коефіцієнтів $\prod_{\substack{t=0 \\ t\neq i}} \frac{x-x_t}{x_i-x_t}$. Побудова частину поліному ґрунтується на ідеї що функція буде приймати значення один в точці x_0.. і 0 в усіх інших.

Недоліком такого підходу є те що на відміну від попереднього способу при додаванні точки потрібно проводити переоцінку всіх коефіцієнтів, що несе за собою додаткові затрати.

Потрібно зазначити що хоча формули поліномів різні, але ідея одна то прогнозовані значення функції для інтерполяції мають вийти однаковими.

Варто визначити що многочлен Лагранжа визначаються за межами інтервалу області інтерполяції, вони ростуть/змінюють напрям дуже швидко. На малюнку функція різко змінює напрям після точки (2, 1). Це небажано, оскільки в цілому вона не відповідає поведінці початкових даних. Таким чином, краще не використовувати цей метод для екстраполяції (Рис. 1.2).

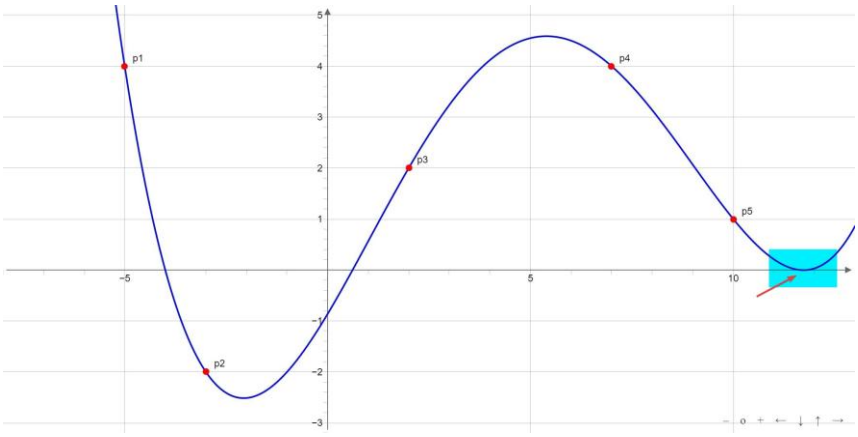


Рисунок 1.2 — Зміна поліном Лагранжа

3. Метод найменших квадратів (МНК). Є одним із способів апроксимації даних [6]. Ідея полягає в пошуку мінімальної суми квадратів відхилень. Потрібні квадрати для того щоб похибки не впливали одна на одну. Він дозволяє використовувати апроксимуючі функції довільного вигляду і відноситься до групи глобальних методів.

Найбільш часто зустрічаються:

- апроксимація прямою лінією (лінійна регресія);
- апроксимація поліномом (поліноміальна регресія);
- апроксимація лінійною комбінацією довільних функцій.

Простішим варіантом методу є перший $\sum_{i=1}^{N}(y_i - a - bx_i)^2 \rightarrow min.$ x_i, y_i – значення експериментальних даних; a, b – коефіцієнти прямої, які ми підбираємо. Для цього частинні похідні прирівнюються до нуля, записуються в систему і фінкція досліджується на екстремум. Отримуємо формули $a = \frac{\sum x_i^2 \sum y_i^2 - \sum x_i \sum y_i}{\Delta}$; $b = \frac{N \sum x_i y_i - \sum x_i \sum y_i}{\Delta}$; $\Delta = N \sum x_i^2 - (\sum x_i)^2$. Також можна оцінити погрішність розрахунку коефіцієнтів a і b.

Існують деякі варіації методу, наприклад МНК з вагами, що будуть корегувати a і b. Чим менше похибка тим вага буде більша.

В багатьох випадках більш складну залежність можна привести до прямої, шляхом лінерізації. Приклад рівняння експоненти ae^{kx} якщо почнемо одразу вирішувати, то замість системи лінійних рівнянь одержимо трансцидентні.

Тому цей вираз логарифмується і додаються заміни z = lny, a = lnA. Тоді в нових позначках, завдання зводиться до пошуку коефіцієнтів лінійної функції z = a + kx. Заміною змінних звели задачу до попередньої. Для неї є формули і завдання легко вирішується.

Апроксимувати залежність можна многочленом. Тоді, важливо мати на увазі, що не потрібно брати дуже високу степінь полінома. Якщо взяти на 1 меньше чим кількість точок, то задача зводиться до глобальної інтерполяції і функція піде через вузли. Щоб такого не сталося зазвичай використовується функція степеню у декілька разів менше чим кількість вузлів, 2-3тя степінь.

В загальному випадку: як і в випадку з прямою розв’язується задача мінімізації. В процесі ми маємо систему лінійних алгебраїчних

рівнянь. В складових якої велика кількість сум вхідних даних різного порядку. Так як значення точок нам відомі, ми рахуємо і виводимо коефіцієнти a_0-a_n.

$$a_0 N + a_1 \sum x_i + a_2 \sum x_i^2 + \cdots + a_n \sum x_i^n = \sum y_i \quad (1.3)$$

$$a_0 \sum x_i + a_1 \sum x_i^2 + a_2 \sum x_i^3 + \cdots \ldots + a_n \sum x_i^{n+1} = \sum y_i x_i \quad (1.4)$$

$$a_0 \sum x_i^n + a_1 \sum x_i^{n+1} + a_2 \sum x_i^{n+2} + \cdots \ldots + a_n \sum x_i^{2n} = \sum y_i x_i^n \quad (1.5)$$

4. Метод найменших квадратів (МНК). Сума квадратів відстаней у до прямої що прогнозуємо має бути мінімальною. Якщо вона буде більше або дорівнювати, то функцію потрібно буде дослідити на екстремум, відшукати стаціонарні точки, в яких він можливий (Рис. 1.3).

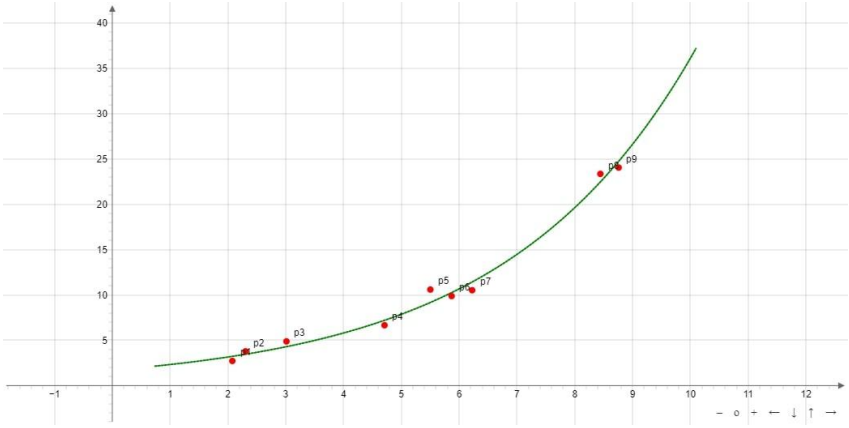


Рисунок 1.3 — МНК

5. Метод сплайнів. Сплайн – це функція що розбиває простір на певні поліном, кусочно-кубічні. Функцію можна застосовувати для різних цілей Вони бувають апроксимуючими та інтерполяційними. Досліджуючи літературу, було вияснено що розрахунки побудови B-сплайну для екстраполяції, не відрізняють від інтерполяційних[13]. Існує певна залежність між кількістю коефіцієнтів та точок. Якщо в рівнянні других більше, то це буде інтерполяційний, а інакше згладжуючий.

Особливо часто використовується кубічний-поліном 3-го ступеня, так як він є безперервною функцією і має безперервні першу і другу похідну на інтервалі інтерполяції (x_0 , x_n). Графік функції може «гнутися» в обидві сторони.

Для побудови слід орієнтуватися на принцип МНК, а в точці перетину/спіпадіння значень у частин розуміти наступне

a. Формули многочленів можна прирівняти. Функція буде перетинати у в точках зліва і справа:

$$L_i(x_i) = y_i,\ i = \text{змінюється від 1 до } n - 1 \quad (1.6)$$

$$L_i(x_i+1) = y_i+1,\ i = 1 \ldots n - 1 \quad (1.7)$$

b. Перші і другі похідні, також рівні. Таким чином кожен поліном як найбільш плавно буде з'єднувалася зі своїми сусідами, ми обмежуємо сплайни безперервними першої і другої похідними в точках даних i = 2, ..., n − 1:

$$Li'(x_i+1) = L_i'+1(xi+1),\ i = 1 \ldots n - 2 \quad (1.8)$$

$$Li''(x_i+1) = L_i''+1(xi+1),\ i = 1 \ldots n - 2 \quad (1.9)$$

c. Для обчислення коефіцієнтів $L_i(x)$ потрібні ще два рівняння. Вони беруться із деякої фрівольності, останні два обмеження є довільними; вони можуть бути обрані відповідно до обставин виконуваної інтерполяції, яка поведінка на краях нам потрібна. Набір кінцевих обмежень полягає в припущенні, що другі похідні дорівнюють нулю в кінцевих точках, тоді, функція приймає вигляд прямої лінії:

$$L_1''(x_1) = 0 \quad (1.10)$$

$$L_n''-1(x_n) = 0 \quad (1.11)$$

Потрібно зауважити з приводу осциляцій. Є функції, які погано піддаються побудові наведених кривих. Це функція модулю, і колокоподібна, наприклад функція Гауса. Справа в тому, що ці фукнкції погано піддаються інтерполяції многочленом.

Разом з тим існує проблема Рунге (Рис. 1.4). Якщо інтерполювати функцію поліномом високої степені, то з’являються осциляції. Тому потрібно в кожному інтервалі зберігати степінь в межах кубічного і ми не отримаємо таку велику волатильність значень.

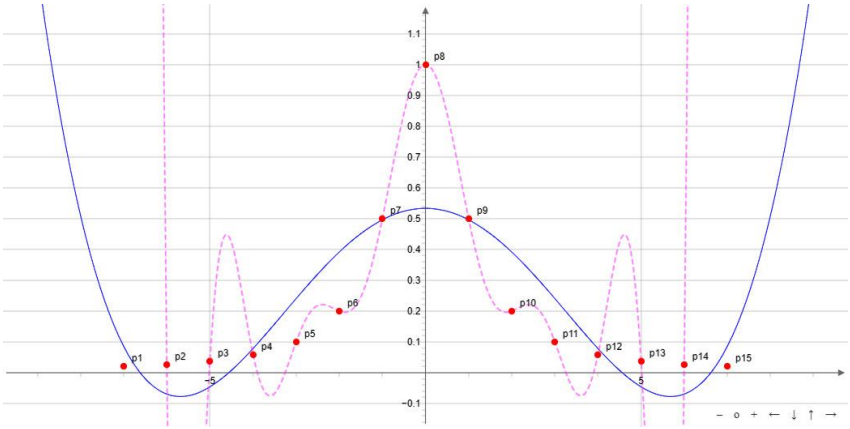


Рисунок 1.4 — Функція Рунге

РОЗДІЛ 2
ПРОЕКТУВАННЯ

2.1 Архітектура бази даних

Проектування має велике значення в функціонуванні майбутнього додатку[7]. Завдяки цьому систему можна буде підтримувати і зберігати її цілісність.

Етапи були наступні:

1. Концептуальний

- Побудова семантичної моделі предметної області тобто інформаційні моделі найбільш високого рівня абстракції.
- Без орієнтації на будь-яку конкретну СУБД і модель даних.
- Включає в себе опис інформаційних об'єктів або понять предметної області і зв'язків між ними. Також опис обмежень цілісності тобто вимога до допустимих значень даних і зв'язків між ними[31].

Основною дійовою особою в системі є учасник. Університет може їх мати від нуля до декількох. Заняття проводиться в кімнатах[Room], у користувача системи може бути їх декілька, тому відношення багато до багатьох. В кімнатах є можливість залишити коментар. Кожен відноситься до певної кімнати, тому відношення один до багатьох. Про подію може бути відправлено сповіщення в декілька місць за бажанням. Одне з яких - телеграм, тому зв'язок один до одного з виключенням. Для побудови графіку необхідність точки, тому відношення між нею і кімнатою один до багатьох. Графік будується одним із методів, їх 4, точку необхідно поєднувати з цими даними тому відношення один до багатьох.

2. Логічний

- Створення схеми бази даних на основі конкретної моделі даних, наприклад реляційної.

- Для попередньої, це модель має на увазі набір схем-відносин із зазначенням первинних ключів, також зв'язків між ними, що представляють собою зовнішні ключі.

- На етапі логіологічного проектування враховується специфіка конкретної моделі даних, однак може не враховуватися специфіка конкретної СУБД.

В кожній із наведених раніше сутностей були створені синтетичні первинні ключі в таблицях. Зроблено зовнішні ключі. В таблиці Учасник, на таблицю Університет та Телеграм, таблиці Точка було посилання на Метод. між Учасник та Кімнатою було є проміжна де також зберігається вказівка на Точку і Коментар (Рис. 2.1).

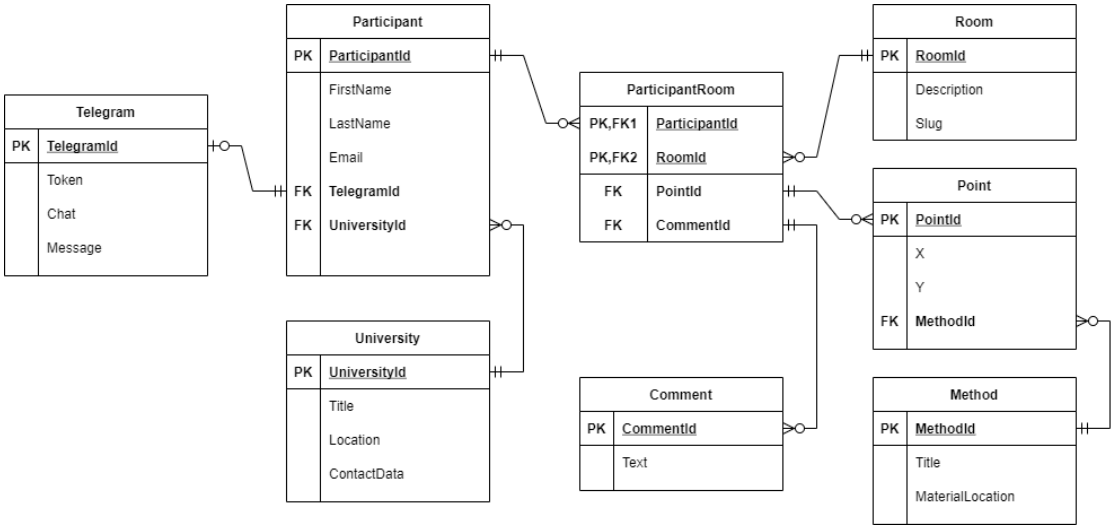


Рисунок 2.1 — ER-діаграма

3. Фізичний

Уточнення схеми, побудованої на попередньому рівні, деталізація типів даних, роботи з СУБД.

Специфіка конкретної СУБД може включати в себе:

- обмеження найменування об'єктів бази даних;

- обмеження на підтримувані типи даних;

- вибір методів управління дисковою пам'яттю;

- поділом БД по файлам і пристроєм;

- розподілу методів доступу до даних;

- особливості реалізації конкретної системи управління.

В якості бази даних була обрана PotgreSQL. Була розглянута можливість використання типу JSON для зберігання даних точки[14]. Однак, це є антипатерном, порушує першу нормальну форму. В такому випадку ускладнюються запити і зміна значень конкретних точок.

Сервіс працює на фреймворці django, Який має власні таблиці і Ролі були пов'язані із моделлю користувачів.

З функціонування системи:

- Основні обчислення проводяться на Сервері.
- Виходячи з предметної області вигідно зберігати результати обчислення методом Ньютона.
- Система може мати досить велике навантаження.

З огляду на вищесказане, було прийнято рішення ввести кеш, NoSQL базу даних Redis. Дані зберігають в масивах по ключах такого типу «id_користувача:метод:координата».

Слід згадати про принципи проектування, спроектована база даних знаходиться в перших трьох формах.

Перша нормальна форма:

- Немає повторюваних рядків.
- Всі атрибути прості типи даних.
- Всі значення визначені.

Друга нормальна форма:

- Таблиці знаходиться в першій нормальній формі.
- Має бути первинний ключ.

- Якщо первинний ключ складається із декількох полів, неключові атрибути мають залежати від нього повністю.

Третя нормальна форма:

- Таблиці повинні бути приведені до другої нормальної форми.
- Не ключові атрибути не мають залежати від не ключових, вони залежали тільки від первинного ключа.

Для дотримання нормальної форми Бойса-Кодда, необхідно щоб:

- Таблиці відповідали третій нормальної формі.
- Ключові атрибути не залежить від неключових, тобто по не ключовому атрибуту не повинно бути можливо однозначно визначити ключовий.

З огляду на вищенаведене наша база даних нормалізована.

2.2 Реляційна і NoSQL бази даних

Що таке SQL? Це декларативна мова програмування для створення та обробки даних у реляційній базі даних. У той же час NoSQL скоріше визначає набір підходів до зберігання даних, відмінних від того, як це робить SQL.

Будучи створеними в 1960-х роках, бази даних NoSQL змогли завоювати реальну популярність тільки останнім часом. Цьому сприяло створення таких баз даних NoSQL, як Couchdb, MongoDB, Apache Cassandra, Redis[15].

Модель. У SQL схема містить типи полів і таблиці бази даних. Це означає, що для запуску вашого проекту на SQL вам необхідно розробити базу даних до будь-якої бізнес-логіки. Після того, як ви створили свою схему, буде досить складно внести будь-які зміни.

Для NoSQL таких обмежень немає. Користувачі можуть вносити зміни в свою базу даних в будь-який час. База даних NoSQL без схем не

вимагає заздалегідь встановлювати будь-якої жорсткий дизайн бази даних.

Транзакції: Найвища цілісність і точність даних має першорядне значення для систем SQL. Ще одним доказом цього є їх механізми транзакцій. Ці бази даних дозволяють розміщувати два або більше оновлень під час будь-якої транзакції. Це означає, що при будь-якій транзакції всі оновлення можуть бути або прийняті, або відхилені. У будь-якому випадку точність ваших даних буде збережена[32].

Цей механізм не працює для баз даних NoSQL, де кожне оновлення приймається або відхиляється індивідуально. Це може призвести до сумних наслідків: в кінцевому підсумку ваші дані можуть виявитися неактуальними.

Масштабованість. Для баз даних SQL масштабованість може бути досить проблематичною. Якщо буде вирішено виділити пов'язані дані, ви можете розглянути можливість кластеризації декількох серверів навколо одного центрального сховища.

Для користувачів NoSQL це набагато простіше виконати. Системи NoSQL пропонують функції масштабування, які ви можете запускати з самого першого дня вашої роботи в системі.

2.3 Застосування Redis

Redis-це сховище даних в форматі NoSQL, має виразність і може зберігати складні типи даних[16].

Із корисних речей:

- можна використовувати як брокер повідомлень;
- закриті списки, і інший функцій набору, комбінації і перекриваються;
- він може виконувати математичних операції зі списками з хорошою продуктивністю[33].

Доцільно використовувати якщо:

- ми читаємо з нього набагато більше, ніж при написанні/оновленні;
- даним дозволяється бути втраченими за деяких обставин.

В проекті він виористовується для кешування даних в двух місцях:

1. Сесій користувачів.
2. Даних обчислень.

Коли ви відкриваєте сайт, браузер відправляє HTTP-запит на сервер. HTTP без стану, значить кожен запит повністю не знає про дії, які виконували попередні запити[34]. Але якщо він не має стану, як ми будемо відстежувати "стан" між сайтом і конкретним браузером, наприклад, вибрана мова або яку тему використовувати. Для вирішення цієї проблеми були створені файли cookie і сеанси.

Коли браузер робить запит вперше, Django створює новий сеанс і зберігає його в базі даних. Дані, які він зберігає, складаються з ключа сеансу, даних сеансу та дати закінчення терміну дії. Ключ - це просто випадковий рядок, яку Django надсилає користувачеві в заголовках відповідей[35].

За замовчуванням дані сеансу зберігаються у базі даних. Яка має свої обмеження. Було дослідження в якому Redis порівнювався з реляційною базою PostgreSQL. По результатам, Redis, виявився швидше в два рази для 100 тис.і 1 млн рядків[17]. З наведеного. Якщо робити комбінацію, зберігати основні дані в реляційній базі, а сесії зберігати в Redis ми отримаємо вийграш в швидкості.

Для цього треба зробити:

Спочатку було написано код менеджера. В файлі redis_engine.py підключили класс SessionBase, cached_property, redis.

Далі створили клас RedisSession наслідуємо від SessionBase в якому написані методи create, load, save, delete, exist.. Окремо слід виділити 2 методи це _connection і save. Перший відповідає за з'єднання з екземпляром бази він помічений декоратором @cached_property щоб виконувати коннект тільки коли обєкт використовується[36]. Другий має атрибут must_create і може створити сессію за необхідністю, разом з тим прописує cookie теперішню дату і час завершення дії.

Після чого було внесено зміни в settings.py: замінено engine сесії на перевизначений нами, визначено час дії cookie на 86400 секунд.

Кешування даних обчислень застосовується в views.py для даних точок. Якщо request.method POST, ми беремо дані по ключам координат. Далі комбінуємо ключ в виді f"{request.user.id}: {reques.GET["method"]}: {reques.GET["xs"]}". Опісля перевіряємо чи є ключ redis_connect .get(kwargs['key']). Якщо він відсутній то створюємо значення массив в який добавляємо дані. redis_connect.rpush(key, *value).

В цілому redis здатний зберігати велику кількість даних. В дослідженні команди Flipgrid, використання памяті для близько мільйона користувачів було 250MB[18]. Однак, про всяк випадок в конфігураційному файлі було прописано обмеження на пам'ять.

Такий підхід дозволяє зменшити кількість звернень до бази даних і підняти швидкість обробки.

РОЗДІЛ 3
РОЗРОБКА

3.1 Фронтенд веб-сервісу

Побудований на фреймворці bootstrap. Центральну область складають графіки які відображаються завдяки Chart.js. Праворуч від нього є пункти вибору полів форми. Є три способи доставити дані на веб-сервер ввести їх вручну, завантажити csv файл, генерувати. Для цього вводиться список іксів і обирається одна із наявних функцій(sin, exp,..) для y.

В правому верхньому куту є зміна мови англійська і українська, вибір теми оформлення сайту і посилання на профіль.

Зліва знаходиться sidebar в якому є пункти домашня сторінка, книга і чати.

Дані на графіку обновляються за допомогою технології AJAX(Рис. 2.2).

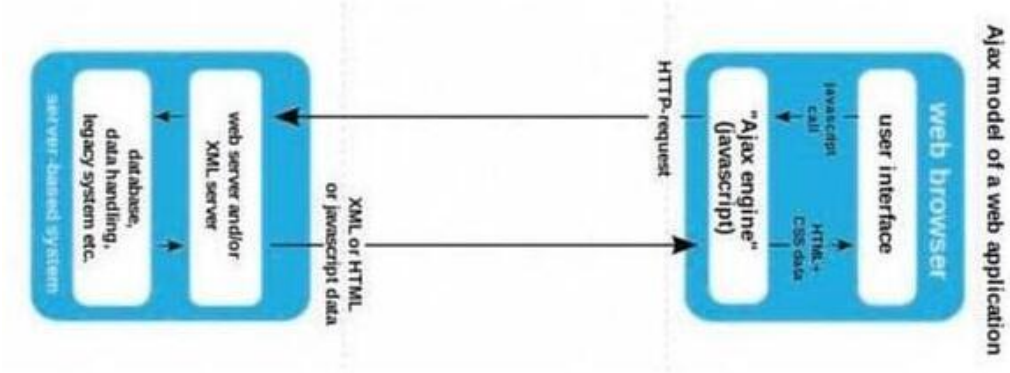


Рисунок 3.1 — AJAX взаємодія з веб-сервером

1. Браузер створює виклик JavaScript, який потім активує XMLHttpRequest[37].

2. У фоновому режимі веб-браузер створює HTTP-запит до сервера.

За це відповідає фукнція ajax_chart. Вона приймає на вхід змінну графіка і endpoint Django. В середині використовує функцію jquery $.getJSON який заодно оновлює підписи і дані графіків.

3. Сервер отримує, витягує і відправляє дані назад у веб-браузер (Рис. 2.3).

В файлі dashboard/views.py функції chart перевіряється request.is_ajax, якщо так, то повертається відповідь JsonResponse відповідні дані[38].

{ "": ["-3", "3"], "data": {"x": [-3, -2, -1, 0, 1, 2, 3] "y": [0.1, 0.2, 0.5, 1, 0.5, 0.2, 0.1], "method": "Spline","latex":"$$x=\frac{1}{1+x^2}$$" }}

4. Веб-браузер отримує запитані дані, які будуть безпосередньо відображатися на сторінці. Перезавантаження не потрібно.

Данні оновлюються функцією по відповідним плейсхолдерам в розмітці. За відображення формули відповідає MathJax-script[39]. Завдяки йому async рендериться LaTeX.

Рисунок 3.2 — Панель управління

3.2 Обробка форми вводу точок

Функціонал обробки форми даних викликається при відправці POST запиту, на url патерн функції post_detail_form;

Слід відмітити, числові методи можуть оперувати над великими масивами, ручний ввід яких є дуже вичерпною процедурою. Тому постає необхідність в автоматизованому зчитуванні даних чи їх генерації для подальшої обробки.

Перше можливо реалізувати, використовуючи Excel (.xls) файли – для зберігання масивів та бібліотеки pandas – для роботи з цими файлами. Зчитування відбувається в кілька кроків: спершу весь файл, використовуючи метод read_excel з параметром, що відповідає шляху до нього на диску системи, далі обробляються окремі стовпці даних, методом iloc[40]. Після отримаємо масиви даних, що можуть бути використані в методах.

Для генерування даних використовується метод бібліотеки numpy, linspace. Значення ys рахуються від xs генератором, наприклад [1/(1+x^2) for x in xs].

3.3 Обчислення методу Лагранжа

Метод Лагранжа[8]. В функцію create_lagrange_polynom подається списки xs, ys. В локальній області видимості створюється список базових поліномів make_polynoms. Після чого запускається цикл for, по довжині масиву xs. В тілі циклу в упомянутий раніше список додаємо результати виклику функції make_basic_polynoms(x, i). і це індекс теперішнього елементу.

make_basic_polynoms(xs, i) діє за принципом обгортки. Вона повертає ссилку на внутрішню функцію make_polynom. Попередня проходиться по списку xs, будує поліном після чого повертає дріб. Ссилка на make_polynom повертається в функції загального поліному create_lagrange_polynom і додається в масив.

Після проходження всих елементів ссилка на акумулюючу функцію lagrange_polynomial повертається із функції. Вона відповідальна сумму добутків у і рахованих вище поліномів.

Весь цей код працює завдяки замиканням.

Імпортується в views.py модуля mathematics і там встраюється в шаблон для рендера (Рис. 2.4).

```
def make_basic_polynoms(xs, i):
    def make_polynom(x):
        divider = 1 #знаменник
        result = 1 #числельник
        for j in range(len(xs)):
            if j != i:
                result *= (x-xs[j])
                divider *= (xs[i]-xs[j])
        return result/divider
    return make_polynom

def create_lagrange_polynom(xs, ys):
    make_polynoms = []
    for i in range(len(xs)): # вичисляємо базові поліноми
        make_polynoms.append(make_basic_polynoms(xs, i))

    def lagrange_polynomial(x): # інтерполююча
        result = 0
        for i in range(len(ys)):
            result += ys[i]*make_polynoms[i](x)
        return result
    return lagrange_polynomial
```

Рисунок 3.3 — Функція розрахунку Лагранж

3.4 Обчислення методу Ньютона

За метод Ньютона відповідає функція count_interpolate_newton. Вона приймає два списка даних, ікси та ігрики. Спочатку зберігається довжина масиву x. Потім дані групуються в комбінації точок і сортуються по збільшенню, так ми можемо не перейматися за порядок.Після цього розпаковуються назад. Визначаються три змінні: два списки(коефіцієнтів, ітераціонних даних) і змінна глубини.

Далі, пока глибина не буде дорівнювати кількості ми будемо рахувати колонку/масив в таблиці. В вкладеному циклі проходимося по коміркам, рахуємо зміну y(різницю наступного і теперішнього). потім зміну x. Частку від різниць зберігаємо в змінну колонки. Після в залежності від метода, в випадку з прямим методом це верхній ряд тому

робимо перевірку якщо ми тільки почали вложений цикл то зберігаємо коефіцієнт.

За межами внутрішнього цикла змінюємо дані для ітерації і збільшуємо глибину.

Завдяки логіці вище отримали коефіцієнти. Тепер щоб порахувати значення по координаті, в середині оглянутої функції є функція count. В ній змінна акамулятор потім цикл по коефіцієнтам. Далі, в ньому ще один, де ми робимо доданки поліному, а в оточуючому формуємо суму. Повертаємо значення, а із count_interpolate_newton повертаємо посилання на внутрішню.

3.5 Обчислення методу наименьших квадратів

Тепер продемонструємо реалізацію апроксимації лінійною функцією методом найменших квадратів.

Вхідними даними є перелік значень змінної x та перелік відповідних ним значень y, які є нашими експериментальними даними, а лінійна функція повинна бути максимально наближена до них[19]. Пошук цієї лінійної функції зводиться до знаходження коефіцієнтів a та b з рівняння y=a*x+b.

Отже, у нас є функція aproxima_square, вхідними параметрами якої є 2 списка:

```
def aproxima_square(x_list, y_list)
```

Потім у тілі функції ми ініціалізуємо початкові значення, які нам необхідно буде обчислити – це суму усіх значень x, суму усіх значень y, суму усіх значення добутку x*y та суму усіх значень квадратів x:

```
sum_x = 0
sum_y = 0
```

```
sum_xy = 0
sum_xx = 0
```

Після цього ми знаходимо кількість елементів у списку зі значеннями x:

```
n = len(x_list)
```

Тепер ми проходимо по усьому списку із значеннями x та списку із значеннями y і знаходимо відповідні суми за допомогою циклу for:

```
for i in range(n):
    sum_x += x_list[i]
    sum_y += y_list[i]
    sum_xy += x_list[i]*y_list[i]
    sum_xx += x_list[i]**2
```

Після того, як ми знайшли усі необхідні суми, необхідно знайти середнє значення відповідних величин. Для цього потрібно поділити знайдені суми на кількість елементів, з яких вони складались:

```
avg_x = sum_x/n
avg_y = sum_y/n
avg_xy = sum_xy/n
avg_xx = sum_xx/n
```

Тепер нам необхідно знайти різницю між середнім значенням квадратів x та квадрату середнього значення x:

```
d_x = avg_xx - avg_x**2
```

І наприкінці ми знаходимо шукані коєфіцієнти a і b та повертаємо їх у вігляді такої структури даних, як кортеж:

```
a = (avg_xy - avg_x*avg_y)/d_x
b = avg_y - a * avg_x
return (a, b)
```

Апроксимація по методу найменьших квадратів може бути застосовано до лінійних залежностей. В випадку коли дані мають вид якоїсь функції, наприклад експоненти вводиться така процедура як лінеарізація. Суть полягає в заміні значень наших даних на інші за формулами щоб привести до виглядку формулу лінії. Так наприклад в логарифмічній формулі a+b*ln(x) замінити q на y, a l на ln(y). Варто пам'ятати що введення таких замін додає обмеження. Таким чином y мусить бути більше нуля.

Та функція що підходить для опису даних краще, має мінімальне середнє квадратичне відхилення.

В коді є класс Linearization, в ньому є конструктор. де ініціалізуються поля title[назва метода]. variable_replace[формула заміни], back_normal[вирази для зворотньої заміни], func_approximate[функція для якої ми робимо апроксимацію] також коефіцієнти a і b та змінні для обмежень.

Класс має методи make, str. Перша буде вичисляти значення y по формулі ax + b. __str__ використовується для строкового повернення обєкту.

Далі існують пять функцій в яких прописані формули для перетворень, із зведеної таблиці. Наприклад def exp_func(x): return math.exp(x).

Після прописані різні апроксимуючі криві. Створюється екземпляр класу і прописуються ліміти через set(lambda:..)

Наприклад функція степеня. При ініціалізації пишемо заміну для y і для x логарифм, третім параметром передаються ссилка на ф-ію експонети і таку ж, останнім функція кривої. За схожим принципом побувадовано для кожної кривої.

Не всі функції можна примінити до наших даних. Вони можуть не співпадати по обмеженнях. Для цього ми визначаємо метод find_suitable_functions. Аргументами він приймає список xs, ys, список для перевірки. В тілі, ми в циклі проходимося по списку функцій. В вкладеному проходимося по данним і перевіряємо умови. Якщо x_lim(x) повертає false ми переходимо в зовнішній цикл і міняємо функцію.

Далі потрібно виконати розрахунок змінених x і y. Це написано в функції make_func_approximate_coefficients. В циклі вона проходиться по перевіреним раніше умовам. За формулами заміни змінює значення x і y. Після цього викликається описаний попереднє МНК для лінійних змінних і повертає кортеж коефіцієнтів. Вони за зворотнюю формулою і зберігаються як поля для теперішньої апроксимуючої функції.

Опісля, потрібно визначити фунцію яка найкраще описує отримані дані, для цього потрібно визначити standart deviation.

В тілі, ми в циклі for ідемо по даним і рахуємо різницю у фактичного від теоретичного. Додаємо значення до акумулючої змінної і по завершенню цикла повертаємо.

Описані вище дії, викликаються в функції find_better_function. Вона порахує коефіцієнти, sd, та визначить кращу.

3.6 Аутентифікація і ауторизація.

Існуючі в системі користувачі можуть входити, виходити з аккаунта і відновлювати пароль. Тепер ми додамо обробник, щоб дозволити їм реєструватися в системі і давати додаткові відомості в профілі.

Ми створили модельну форму для користувача, включивши в неї тільки поля username, first_name і email. Вони будуть валідуватися відповідно до типу полів моделі. Наприклад, якщо користувач введе логін, який вже використовується, йому повернеться повідомлення із зазначенням на помилку, через те що в моделі поле username визначено як unique=True.

Також ми додали два поля: password і password2 - для завдання і підтвердження пароля. У методі clean_password 2() ми перевіряємо, чи збігаються обидва паролі. Якщо вони відрізняються, то буде повернута помилка.

Ми можемо додавати методи з назвою виду clean_<fieldname> () для будь-якого поля форми, щоб Django перевіряв відповідне поле і в разі некоректних даних прив'язував помилку до нього[20]. Крім того, у форм реалізований метод clean (), який перевіряє коректність всієї форми. Він застосовується, коли необхідно виконати перевірку взаємопов'язаних полів.

Обробник реєстрації нового користувача гранично простий. Замість збереження пароля користувача» як є", ми використовуємо метод set_password() моделі User. Він збереже пароль в зашифрованому вигляді.

3.7 Розширення моделі користувача.

У моделі користувача Django описаний мінімальний набір полів для зберігання відомостей про користувачів. У більшості випадків при роботі з акаунтами цих полів досить. Але в деяких проектах можуть знадобитися додаткові дані користувачів. Щоб зберігати ці відомості, потрібно зробити свою модель профілю, яка буде містити всі додаткові поля і посилання «один до одного» на модель Django.

Поле один до одного user дозволить нам пов'язати додаткові дані з конкретним користувачем. Ми передаємо CASCADE як параметр on_delete, тому пов'язані з користувачем дані будуть видалені при видаленні основного об'єкта User. Поле photo має тип ImageField. Для роботи з ним нам необхідно встановити бібліотеку зображень Pillow за допомогою команди: pip install Pillow==5.1.0 для того щоб Django знав, де зберігати медіафайли, завантажені користувачами, додали наступні рядки в settings.py:

```
MEDIA_URL = '/media/'
MEDIA_ROOT = os.path.join(BASE_DIR, 'media/')
```

В файлі налаштувань проекта є фрагменгт коду:

```
if settings.DEBUG:
    urlpatterns +=
static(settings.MEDIA_URL,document_root=settings.MEDIA_ROOT)
```

Ці налаштування переписуються під сервер при подальшому розгортанні.

На сайті в блоці кімнати / навчальному місці користувачі можуть залишати коментарі.

Для того щоб все працювало вірно потрібні наступні кроки:

1. Модель для збереження коментарів.
2. Форма для відправки і валідації коментарів.
3. Обробник, який буде перевіряти форму і зберігати коментар.
4. Шаблон окремої сторінки, список коментарів.
5. Налаштування панелі адміністратора.

Модель Comment містить Foreign Key для прив'язки до певної кімнати. Це відношення визначено як "один до багатьох", тому що одна

може мати безліч коментарів, але кожен коментар може бути залишений тільки для однієї статті.

Атрибут related_name дозволяє отримати доступ до коментарів конкретної кімнати. Тепер ми зможемо звертатися до кімнати з коментаря, використовуючи запис comment.room, і назад за допомогою room.comments.all().

Якби ми не визначили related_name, Django використовував би ім'я пов'язаної моделі з постфіксом _set (наприклад, comment_set).Ми додали булеве поле active, для того щоб була можливість приховати деякі коментарі(через лексику..), і визначили поле created для сортування коментарів в хронологічному порядку.

Форма коментаря у файлі forms.py.

Обробка і збереження коментарів будуть відбуватися в контроллері, тому тут немає методу збереження. Ця форма використовується тільки для введення коментарів і відправки їх на сервер.

Django надає два базових класи для форм: Form і ModelForm. Тут застосовуємо ModelForm, ми зможемо динамічно генерувати форму по моделі Comment.

Поле класса parent_comment дуже важливе, оскільки воно буде містити ідентифікатор батьківського коментаря, який буде автоматично замінений при відповіді на один з коментарів під статтею. Це поле буде приховане. Крім того, заповнення необов'язково, оскільки коментарі можуть мати пряме відношення до статті.

У views перед методом коментаря стоїть декоратор @login_required.

Над методом room_detail для відображення сторінки кімнати. В коді room_detail щоб отримати всі активні коментарі є змінна:comments = room.comments.filter(active=True). Це QuerySet, через об'єкт кімнати room, викликається менеджер пов'язаних об'єктів comments, визначений в моделі Comment в аргументі related_name.

Щоб зберегти коментарі, використовуйте той же менеджер. Спочатку змінна new_comment має значення None. Пізніше вона буде використовуватися, коли новий коментар буде успішно створений. Коли приходе GET запит форма ініціалізується comment = Comment Form (). Якщо ж отримуємо room запит, то заповнюємо форму даними із запиту і валідуємо її функцією is_valid().

Якщо форма заповнена некоректно, повертаємо в template данні і бачимо інформацію з повідомленнями про помилки. Якщо всі поля успішно пройшли валідацію, виконуємо наступні кроки.

В шаблоні, блок коментаря це рядки в системі сітки Bootstrap. Вид дерева реалізується шляхом переміщення стовпців. Дуже важливим тут є наявність у кожного рядка id={{ comment.id }} - те значення, яке буде підставлятися в приховане поле форми, якщо користувач коментує коментар, а не статтю в цілому.

У користувача завдяки цьому ж id і JavaScript буде переміщатися форма коментаря по сторінці. За переміщення відповідає функція show_comments_form(). Вона викликається в обробник посилання Відповісти у кожного коментаря і в кнопці Прокоментувати. Ця функція використовує бібліотеку jQuery. Тому вона підключається в базовому шаблоні. В відображенні використовується Bootstrap тому підключається та версія.

Js регулює відображення так, якщо id == write_comment , то означає, що коментар має перший рівень і Приховане поле буде порожнім, а форма переміщається під напис Прокоментувати. В іншому випадку заповнюється приховане поле і розміщується під коментарем, під яким ми пишемо відповідь.

Поведінка того що відображати, в залежності від умов доступу прописуються в шаблоні сторінки. Якщо користувач не авторизований, то показується рендер повідомлення про необхідність реєстрації/звязок з вчителем.

Для агрегації отриманим данними і налаштування вони доступні в адмін панелі. Попередній звязок додано в admin.py. Там класс CommentAdmin що наслідується від ModelAdmin щоб могти міняти поля і вони прописані в формі атрибутів з кортежами[21]. Окремої уваги заслуговує list_filter. Тут є агрегуючі поля по даті і вкладеності. Щоб вони працювали як стовпець помічені read_only.

3.8 Система сповіщень

Надсилання електронної пошти є однією з важливих частин веб-сайту. Фактично, майже всі веб-сайти потребують цієї частини, для таких операцій, як активація облікових записів за допомогою електронних листів користувачів, скидання паролів та інші дії.

Для надсилання електронних листів з веб-сайту нам був потрібен сервер хостингу електронної пошти, і в цьому випадку для тесту ми будемо використовувати SMTP-сервер Google[22]. Для того, щоб використати сервер треба включити IMAP в налаштуваннях і ввімкнути доступ для додатків менш безпечних для облікового запису.

Налаштування сервера прописуються внизу settings.py

EMAIL_BACKEND вказує серверну частину django, яка буде працювати з хост-сервером електронної пошти.

EMAIL_HOST це посилання на smtp-сервер google, вказаний Google.

EMAIL_USE_TLS повідомляє django, який безпечний протокол слід використовувати для підключення до сервера.

EMAIL_PORT дорівнює 587.

Наступні два поля відповідають за логін і пароль для авторизації на сервері[23]. Вони, як і інша конфіденційна інформація зберігаються в змінних середовища.

Для відправки електронних листів було інпортовано деякі компоненти в views.py файл додатка notify.

send_mail-це функція, яка надсилатиме електронні листи нашим користувачам, вона використовує раніше додані налаштування. Вона приймає кілька параметрів, але для простоти використовуються необхідні, для таких цілей, як надсилання посилання для підтвердження облікового запису чи пароля. У файлі перегляду функції відправки буде мати наступні поля:

subject: тема, залежить від endpoint

message: в випадку з підтвердженням аккаунту буде містити посилання, з тимчасовим токеном

email_from: пошта для відправки

recipient_list: електронна пошта користувача

Нотифікації телеграм це другий метод сповіщення який доступний на сайті.

Для його роботи необхідні id користувача[вчителя], чи чата в який потрібно надсилати повідомлення.

Дані зберігаються в моделі TeleSettings, вона має відповідні три поля tg_token, tg_chat, tg_message.

Телеграм дозволяє керувати ботом запитами по https. Для цього написана функція sendTelegram[24]. На вхід вона приймає два атрибути tg_name, tg_phone. Спочатку перевіряється чи вдалося отримати id користувача із бази даних. Потім із моделі беруться описані раніше атрибути і формується запит api + token + '/sendMessage'.

Текст для повідомлення редагується в адмін панелі, щоб вставити туди специфічні дані користувача надіславшого коментар, чи інше звернення, розміщютьсяються в фігурних дужках. Тут ці плейсхолдери виявляються, текстовими функціями r.find і заміняються коректними даними.

Опісля в блокі try через requests виконується POST запит. В блокі finally записуються логи.

Ця функція визивається визивається в декількох місцях. Одне із них це thanks_page в views.py. Сторінка відображається як привітання після реєстрації. Ім'я і телефон користувача відправляються в цю функцію після чого передаються в шаблон.

3.9 Розгортання

Для публікаціє є декілька можливих місць це: хостинг, віртуальний сервер, хмарні сервера.

Хостинг, наприклад Heroku. Переваги такого рішення, відносно бистрий запуск сервісу, системи розподілені по різним додаткам, внесені частина налаштувань, а саме налаштування nginx, etc. Мінусом для

безкоштовних аккаунтів є труднощі в налаштуванні глобальних dns і конфігурації параметрів серверу[25].

Віртуальний сервер(VPS, VDS). Переваги, це повноцінний сервер. Мінуси - це обмеження по потужності в 5-10%, 100% зазвичай дається на невеликий період протягом місяця.

Хмарні сервери. Переваги – це виділений сервер, можна побудувати масштабовану архітектуру, додаткові можливості для інтеграції з Amazon Web Services, кількість послуг, надійність(зазвичай їх представляють великі компанії), плата тільки за ті ресурси які використовуються[26]. Мінусом є ціна.

З огляду на перекислене вище було обрано хмарні сервіси, розглянуто налаштування веб-серверу на системі linux.

Буде використовуватися:

Ubuntu 20.04 LTS.

Gunicorn - це HTTP-сервер Python WSGI для UNIX, який буде взаємодіяти з NGINX- веб-сервером. NGINX приймає вхідний HTTP-запит і відправляє його на сервер gunicorn[27]. Він переводе запити отримані від NGINX в формат який може обробляти Django веб-додаток.

NGINX - це програмне забезпечення з можливостями кешування, потокової передачі, балансування навантаження. Зараз ми будемо використовувати його тільки в якості веб-сервера[28].

Supervisor-це програма, яка дозволяє своїм користувачам відстежувати і контролювати кілька процесів в Linux системах. Ми будемо використовувати supervisor для управління процесами (gunicorn), автоматичного запуску програм, в випадку падіння[29].

Cerbot – це сервіс що дає можливість отримати HTTPS сертифікат для веб-сайту. Вони мають пакет для розгортання в Linux системах. Це дасть можливість індексуватися в пошукових системах, використувати мультиплексування. Можна передавати дані від браузера до серверу не турбуючись про їх безпеку[30].

Процес:

1. Запустити сервер.
2. Відкрити 22 порт для з’єднання по SSH.
3. Завантажити ключ від провайдера.
4. Зберегти в директорію користувача.
5. Виконати команду chmod 400. Щоб надати необхідні дозволи і зробити ключ доступним тільки даному користувачу.
6. Під єднатися до серверу SSH -i "ключ.pem" [користувач_логін]@[сервер_адрес].
7. Змінити стандартний пароль, команда passwd.
8. Налаштувати DNS на сайті реєстратора домена.
9. Обновити пакети на сервері, sudo apt-get update і upgrade.
10. Перевіряємо наявність python. В нашому випадку присутня версія потребувала оновлення.
11. Встановлюємо python і службові пакети htop, redis-server, nginx, supervisor, git, wget, curl.
12. sudo bash. Відкрити консоль з правами адміністратора.
13. Помилки нема, команда exit.
14. Створюємо папку проекту на сервері.
15. Активуємо віртуальне оточення.
16. Копіюємо в папку проекту завдяки gitclone.
17. Встановлюємо -r requirements.txt.
18. Встановлюємо gunicorn в python.

19. Створюємо конфігураційний файл. Тут прописана команда для запуска із віртуального оточення, ip і порт на якому він запускається. Також є користувач, ліміти, шлях до налаштувань django. Workers по принципу ядро серверу помножене на два плюс один.

20. запустити gunicorn в папці bit, .start_gunicorn.sh.

21. Створюємо bash скрипт для запуску.

22. Даємо права на виконання.

23. Налаштовуємо дефолтні налаштування NGINX. Тут написано що працюємо на 80 порту, html сторінки NGINX, ім’я серверу наш домен, локація залишаємо як є. Тут прописані налаштування, які проксирують запит, який прийшов в gunicorn запущений на 8003 порту, атрибути зєднання.

24. Встановлюємо PostgreSQL, для цього пакети postgresql postgresql-contrib libpq-dev.

a. Створити юзера бази, create user wikistudy with password '…';

b. Створити базу даних Postgre, CREATE DATABASE..

c. Дати привілегії на привілегії на таблицю, GRANT ALL PRIVILEGES..

d. Налаштовуємо локалі: client_encoding, default_transaction_isolation, timezone.

25. Пишемо конфігурації для supervisor. Тут шлях до скрипту запуску gunicorn, користувач бд, назва програми, автостарти.

26. Перезапускаємо сервер.

27. Підключення SSL сертифікату.

a. Встановити spand.

b. Встановити –classic certbot.

c. Прописати символьне посилання на конфігурацію.

d. Вводимо команду автовстановлення.

i. Дозволи умови, приватність.

ii. Вказуємо цифрою потрібний домен.

28. Налаштовано. В конфігураційному файлі NGINX, Cerbot прописав шлях до сертифікатів, зробив автоматичне перенаправлення трафіку на 443 порт.

ВИСНОВОК

Результатом кваліфікаційної роботи є дослідження методів інтерполяції екстраполяції та апроксимації. Було розглянуто та інтегровано: метод Ньютона; метод Лагранжа; метод найменших квадратів; метод сплайнів.

Вияснено їх особливості роботи. Так Метод Ньютона, має дві окремі формули для підрахунку. Метод Лагранжа потребує перерахунку при додаванні точки. Хоча обидва методи мають однакові значення прогнозу.

Досліджено технології функціонування веб-сервісів. А саме: принципи проектування; окреме використання реляційної PostgreSQL і NoSQL кешу Redis; робота з MVP фреймворку Django, взаємодія з сервером при AJAX запитах.

Розроблено програмний продукт та проведено розгортання на linux сервері. З внесенням необхідних конфігурацій прошарків NGINX, gunicorn, та supervisor.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Mathematical model [Електронний ресурс] – Режим доступу до ресурсу: https://www.oxfordreference.com.

2. Newton's Method - Mathematical Python [Електронний ресурс] – Режим доступу до ресурсу: https://personal.math.ubc.ca.

3. Mathcad - описание продукта и расширений - ПТС [Електронний ресурс] – Режим доступу до ресурсу: https://pts-russia.com › mathcad.

4. Мапле [Електронний ресурс] – Режим доступу до ресурсу: http://fmi.npu.edu.ua › Maple_metodychka_1.

5. Lagrange Polynomial Interpolation [Електронний ресурс] – Режим доступу до ресурсу: https://pythonnumericalmethods.berkeley.edu.

6. Линник Ю. В. Метод наименьших квадратов и основы математико-статистической теории обработки наблюдений / Юрий Витльевич Линник. – Москва, 1958. – 336 с. – (Физматгиз). – (158; кн. 1147).

7. Табунщик Г. В. ПРОЕКТУВАННЯ ТА МОДЕЛЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СУЧАСНИХ ІНФОРМАЦІЙНИХ СИСТЕМ / Г. В. Табунщик, Т. І. Каплієнко, О. А. Петрова. – Запоріжжя, 2016. – 250 с. – (Дике Поле). – (158; кн. 300).

8. Інтерполяція за Лагранжем [Електронний ресурс] – Режим доступу до ресурсу: https://web.posibnyky.vntu.edu.ua/fksa/2kvetnyj_komp'yuterne_modelyuvannya_system_procesiv/t1/612..htm..

9. Ресурс Udacity для вивчення програмування [Електронний ресурс] – Режим доступу: https://www.udacity.com/

10. Хрипко Т. Є. ПРОГРАМНІ ЗАСОБИ КОМП’ЮТЕРНОЇ МАТЕМАТИКИ [Електронний ресурс] / Тетяна Єлисеївна Хрипко – Режим доступу до ресурсу: https://conferences.vntu.edu.ua/index.php/pmovc/pmovc/paper/view/5665/4811.

11. Overview [Електронний ресурс] – Режим доступу до ресурсу: https://www.maplesoft.com/support/help/maple/view.aspx?path=latex.

12. Speak [Електронний ресурс] – Режим доступу до ресурсу: https://reference.wolfram.com/language/ref/Speak.html.

13. Кривые Безье и B-Сплайны [Електронний ресурс] – Режим доступу до ресурсу: http://polybook.ru/comma/1.3.2.pdf.

14. JSON Functions and Operators [Електронний ресурс] – Режим доступу до ресурсу: https://www.postgresql.org/docs/9.3/functions-json.html.

15. NoSQL [Електронний ресурс] – Режим доступу до ресурсу: https://en.wikipedia.org/wiki/NoSQL.

16. Introduction to the in-memory datastore Redis [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/featurepreneur/introduction-to-the-in-memory-datastore-redis-e1ba0a3d3f20.

17. POSTGRESQL VS REDIS VS MEMCACHED PERFORMANCE [Електронний ресурс] – Режим доступу до ресурсу: https://www.cybertec-postgresql.com/en/postgresql-vs-redis-vs-memcached-performance/.

18. Estimating Memory Use in Redis and Mass Insert [Електронний ресурс] – Режим доступу до ресурсу: https://andyatkinson.com/blog/2019/08/06/redis-memory-use-mass-insert.

19. Метод найменших квадратів [Електронний ресурс] – Режим доступу до ресурсу: https://znaimo.com.ua/%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_%D0%BD%D0%B0%D0%B9%D0%BC%D0%B5%D0%BD%D1%88%D0%B8%D1%85_%D0%BA%D0%B2%D0%B0%D0%B4%D1%80%D0%B0%D1%82%D1%96%D0%B2.

20. Django - User Profile [Електронний ресурс] – Режим доступу до ресурсу: https://dev.to/earthcomfy/django-user-profile-3hik.

21. Django Tutorial Part 4: Django admin site [Електронний ресурс] – Режим доступу до ресурсу: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/Admin_site.

22. Check Gmail through other email platforms [Електронний ресурс] – Режим доступу до ресурсу: https://support.google.com/mail /answer/7126229?hl=en.

23. Sending email [Електронний ресурс] – Режим доступу до ресурсу: https://docs.djangoproject.com/en/3.2/topics/email/.

24. Telegram Bot API [Електронний ресурс] – Режим доступу до ресурсу: https://core.telegram.org/bots/api.

25. Deploying containerized NginX to Heroku - how hard can it be? [Електронний ресурс] – Режим доступу до ресурсу: https://dev.to/levelupkoodarit/deploying-containerized-nginx-to-heroku-how-hard-can-it-be-3g14.

26. What Is Amazon Web Services and Why Is It So Successful? [Електронний ресурс] – Режим доступу до ресурсу: https://www.investopedia.com/articles/investing/011316/what-amazon-web-services-and-why-it-so-successful.asp.

27. Dockerizing Django with Postgres, Gunicorn, and Nginx [Електронний ресурс] – Режим доступу до ресурсу: https://testdriven.io/blog/dockerizing-django-with-postgres-gunicorn-and-nginx/.

28. Introducing a Technology Preview of NGINX Support for QUIC and HTTP/3 [Електронний ресурс] – Режим доступу до ресурсу: https://www.nginx.com/blog/introducing-technology-preview-nginx-support-for-quic-http-3/.

29. Introduction [Електронний ресурс] – Режим доступу до ресурсу: http://supervisord.org/introduction.html.

30. https://github.com/certbot/certbot/

31. Проєктування бази даних [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%94%D0%BA%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85.

32. Модель даних SQL [Електронний ресурс] – Режим доступу до ресурсу: https://elearning.sumdu.edu.ua/free_content/lectured:a8104441b8e00905159c1ff04257b014dd456247/20151208100106/162253/index.html.

33. How fast is Redis? [Електронний ресурс] – Режим доступу до ресурсу: https://redis.io/topics/benchmarks.

34. An overview of HTTP [Електронний ресурс] – Режим доступу до ресурсу: https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview.

35. How to use sessions [Електронний ресурс] – Режим доступу до ресурсу: https://docs.djangoproject.com/en/3.2/topics/http/sessions/.

36. Python Functools – cached_property() [Електронний ресурс] – Режим доступу до ресурсу: https://www.geeksforgeeks.org/python-functools-cached_property/.

37. Ajax Getting Started [Електронний ресурс] – Режим доступу до ресурсу: https://developer.mozilla.org/en-US/docs/Web/Guide/AJAX.

38. Django JsonResponse [Електронний ресурс] – Режим доступу до ресурсу: https://zetcode.com/django/jsonresponse/.

39. Writing Mathematics for MathJax [Електронний ресурс] – Режим доступу до ресурсу: http://docs.mathjax.org/en/latest/basic/mathematics.html.

40. pandas.read_excel [Електронний ресурс] – Режим доступу до ресурсу: https://pandas.pydata.org/docs/reference/api/pandas.read_excel.html.

ДОДАТКИ

Додаток А

КОДЕКС АКАДЕМІЧНОЇ ДОБРОЧЕСНОСТІ
ЗДОБУВАЧА ВИЩОЇ ОСВІТИ ХЕРСОНСЬКОГО
ДЕРЖАВНОГО УНІВЕРСИТЕТУ

Я, Шиплан Владислав Володимирович,
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.

07.09.2020 (дата) | (підпис) | Шиплан Владислав (ім'я, прізвище)