Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра інформатики факультету інформатики

Розробка системи “Запис на курсові роботи”

Текстова частина до курсової роботи

Галузь знань: 12 «Інформаційні технології»

Спеціальність: 122 «Комп’ютерні науки та інформаційні технології »

Керівник курсової роботи

к-т фіз. мат. наук, ст. викладач

Шабінська Марина Олегівна

(підпис)

“____” _____ 2020 р.

Виконав студент КНІТ-4

Нагнибіда А. А.

“____” _____ 2020 р.

Київ 2020

ЗМІСТ

Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра інформатики факультету інформатики

ЗАТВЕРДЖУЮ

Зав. кафедри інформатики,
кандидат фізико-математичних наук, доцент _____ С. С. Гороховський

(підпис)
„____”_____2020 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ
на курсову роботу

студенту Нагнибіді Андрію Андрійовичу Факультету Інформатики 4 курсу
ТЕМА: Розробка системи запису на курсові роботи

Вихідні дані:
Зміст ТЧ до курсової роботи:
Індивідуальне завдання
Календарний план
Вступ
РОЗДІЛ 1: Технічне завдання проекту
РОЗДІЛ 2: Технічні засоби
РОЗДІЛ 3: Практична частина
Висновки
Список використаної літератури

Дата видачі „___” _____ 2020 р. Керівник _____
(підпис)

Завдання отримав _____

(підпис)

Тема: Розробка системи запису на курсові роботи
Календарний план виконання роботи:

№ п/п	Назва етапу дипломного проекту (роботи)	Термін виконання етапу	Примітка
1.	Отримання теми курсової роботи.	07.11.2019	
2.	Ознайомлення з тематичними матеріалами та побудова структури практичної та теоретичної частин курсової роботи	10.12.2019	
3.	Написання вступу та змісту роботи	08.1.2020	
4.	Створення системи	21.01.2019	
5.	Тестування та внесення змін до створеної системи	27.02.2019	
6.	Оформлення роботи відповідно до вимог написання курсової роботи.	22.03.2019	
7.	Узгодження попередньої версії роботи з керівником	28.03.2020	
8.	Внесення змін до курсової роботи відповідно до зауважень наукового керівника	14.04.2020	
9.	Захист курсової роботи	2020	

Студент Нагнибіда А.А.

Шабінська М.О.

“_____”_____

Вступ

Інформаційні технології — вимога сьогодення, що дозволяє створити суспільство, засноване на знаннях. Новітні інформаційні технології стрімко увірвались до всіх сфер нашого життя, стали такою ж реальністю, як телефонний зв'язок чи подорожування літаком. І цей стрімкий “ривок”, не обійшов стороною і наш університет. Сучасні технології проклали шлях для багатофункціональних пристроїв, таких як смартфони та розумні годинники. Комп'ютери стають все швидшими, більш портативними та більш потужними, ніж будь-коли раніше. Завдяки цим технологічним революціям наше життя стало простішим, швидшим, кращим та веселішим.

На сьогодні в НАУКМА немає автоматизованої системи запису на курсові роботи, тому метою моєї курсової є *оцифрування* Національного Університету Києво-Могилянською Академії, розробивши вище сказану систему.

Головні пріоритети при розробці системи:

- Повністю автоматизована - не потрібно втручання програміста для контролю та роботи програми
- Зручність у споживанні для всіх Користувачів
- Обробка великих обсягів даних

Курсова робота включає в себе 3 частини:

- Частина 1 – Технічне завдання проекту.
- Частина 2 – Технічні засоби.
- Частина 3 – Безпосередньо практична частина.

1. Технічне завдання проекту

1.1. Аналіз існуючої системи

НАУКМА нараховує шість факультетів, кожен з яких містить більше трьох кафедр. Кожен викладач працює на одній з кафедр. Студент навчається на факультеті та має доступ до всіх тем курсових робіт, закріпленими за кафедрами, які належать до його факультету.

Кожен викладач надає певну кількість тем курсових робіт до методиста кафедри, який, у свою чергу, збирає усі запропоновані теми, та формує відповідний документ.

Після цього, студент має відвідати кафедри свого факультету, щоб вибрати тему роботи. У відповідному документі студент має відмітити обрану тему курсової роботи, записавши своє ім’я навпроти. Викладач, що запропонував тему закріплюється за студентом як науковий керівник.

Такий запис має безліч недоліків:

- Обов’язкова фізична присутність студента на кафедрі для запису
- Складність статистичного нагляду над процесом запису
- Відслідковування вільним тем
- Складнощі зі зміною будь-яких даних по темі роботи

Головною метою є прибрати ці недоліки та зробити зручну систему для всіх користувачів.

1.2. Загальний опис поставленої задачі

Головна задача: створити повністю автономну систему запису на Курсові Роботи.

Користувачі цієї системи розділені на три Ролі:

- Адміністратор
- Викладач

- Студент

Елементи, які беруть участь у Проекті:

- Факультет
- Кафедра
- Тема Курсової Роботи
- Заявка на Курсову Роботу

Властивості **Елементів** наступні:

Кафедра -> Прив’язка до Факультету.

Тема Курсової Роботи -> Прив’язка до Кафедри. Прив’язка до Викладача, який її створив.

Заявка на Курсову Роботу -> Прив’язка до Теми Курсової Роботи. Прив’язка до Викладача, який її створив.

Властивості **Користувачів** наступні:

Адміністратор -> Створення та редагування Користувачів. Створення та редагування Факультетів. Створення та редагування Кафедр.

Викладач -> Створення Тем Курсових Робіт. Редагування тільки створених Тем Курсових Робіт. Прийняття чи відхилення Заявок на Курсову Роботу. Редагування власного Профілю. Прив’язується до Кафедри.

Студент -> Подання Заявок на Курсову Роботу, тільки до тих Кафедр, які відносяться до Факультету, до якого записаний студент. Редагування власного Профілю. Прив’язується до Факультету.

2. Технічні засоби

2.1. Загальний Опис використаних Технічних засобів

Головною метою практичної частини була розробка Автоматизованої Системи Запису на Курсові Роботи. Для цього вирішено було зробити зручний WEB-застосунок, адже тоді можна отримати доступ до цієї системи прямо з телефону з будь-якої частини світу.

WEB-застосунок - це комп'ютерна програма клієнт-сервер, з якою користувач (включаючи інтерфейс користувача та логіку на стороні клієнта) працює у веб-браузері. Поширені веб-програми включають веб-пошту, роздрібні продажі в Інтернеті, Інтернет-банкінг та інформаційні сайти.

Програма зазвичай розбивається на логічні шматки, які називають "ярусами", де кожному ярусу відводиться роль. Традиційні програми складаються лише з 1 ярусу, який знаходиться на клієнтській машині, але веб-додатки за своєю природою піддаються n-ярусному підходу. Хоча можливо використати безліч варіацій, найпоширенішою структурою є трирівнева програма. У найпоширенішій формі три рівні називають презентацією, логікою та сховищем у такому порядку. Веб-браузер - це перший рівень (презентація), двигун, що використовує деяку динамічну технологію веб-контенту (наприклад JavaScript). Серверна частина - це середній рівень (логіка), а база даних - третій рівень (сховище). Веб-браузер надсилає запити до середнього рівня, який обслуговує їх, здійснюючи запити та оновлення відносно бази даних та генеруючи інтерфейс користувача.

Отже, треба було реалізувати наступні етапи:

- Інтерфейс Програми
- Серверну частину
- Базу Даних

2.2. Інтерфейс Програми

Для розробки Інтерфейсу програми були використані такі мови програмування:

- HTML
- CSS
- JavaScript

Hypertext Markup Language (HTML) - це стандартна мова розмітки для документів, призначених для відображення у веб-браузері. Цьому можуть допомогти такі технології, як каскадні таблиці стилів (CSS) та мови сценаріїв, такі як JavaScript.

Веб-браузери отримують HTML-документи з веб-сервера або з локального сховища та переводять документи на мультимедійні веб-сторінки. HTML описує структуру веб-сторінки семантично та містить підказки для зовнішнього вигляду документа.

Cascading Style Sheets (CSS) - це таблиці стилів, які використовується для опису подання документа, написаного мовою розмітки, наприклад HTML. CSS - це одна з основних технологій всесвітньої павутини, поряд із HTML та JavaScript.

CSS був розроблений, щоб забезпечити розділення презентації та контенту, включаючи кольори та шрифти. Цей поділ покращує доступність контенту, забезпечує більшу гнучкість та контроль у конкретизації характеристик презентації, дає змогу декількам веб-сторінкам обмінюватися форматуванням, вказавши відповідний CSS в окремому файлі .css, а також зменшити складність та повторність структурного вмісту.

Поділ форматування та контенту також робить можливим подання однієї і тієї ж розмітки сторінки в різних стилях для різних методів візуалізації. CSS також має правила альтернативного форматування, якщо доступ до вмісту виконується на мобільному пристрої.

Поряд із HTML та CSS, JavaScript є однією з основних технологій всесвітньої павутини. JavaScript включає в себе можливість для створення інтерактивних веб-сторінок і є невід'ємною частиною веб-додатків. Переважна більшість веб-сайтів використовують його для проведення операцій на стороні

клієнта, а всі основні веб-браузери мають спеціальний механізм JavaScript для його виконання.

Всередині цього проекту було використано два потужних фреймворки для роботи з клієнтською стороною, а саме:

- Jquery (для JavaScript)
- Bootstrap (для CSS та JavaScript)
- Material Design (для CSS та JavaScript)

Jquery - jQuery - це швидка та багатофункціональна бібліотека JavaScript. Завдяки простому у користуванні API, який працює майже у всіх браузерах, такі речі, як обробка та маніпулювання документами HTML, обробка подій, анімація та Ajax, стають набагато простіші. Завдяки поєднанню універсальності та розширюваності, jQuery змінив спосіб, яким мільйони людей пишуть JavaScript. [1]

Bootstrap - це інструментарій із відкритим кодом для розробки за допомогою HTML, CSS та JS. Дозволяє швидко прототипувати веб-застосунок або побудувати весь додаток за допомогою змінних та сумішей Sass. Складається з широких попередньо вбудованих компонентів та потужних плагінів, побудованих на jQuery. [2]

Material Design - Створений та розроблений Google, Material Design - це мова дизайну, яка поєднує класичні принципи успішного дизайну разом із інноваціями та технологіями. Мета Google - розробити систему дизайну, яка б дозволила забезпечити єдиний досвід користувачів для всіх їх продуктів на будь-якій платформі. [3]

2.3. Серверна Частина

Для написання серверної частини WEB-застосунку була використана мова програмування PHP.

Hypertext Preprocessor (PHP) - популярна мова сценаріїв загального призначення, яка особливо підходить для веб-розробки. Швидкий, гнучкий та прагматичний, PHP надає повноваження на все, від вашого блогу до найпопулярніших веб-сайтів у світі. [4]

PHP-код, як правило, обробляється на веб-сервері інтерпретатором PHP, реалізованим як модуль. На веб-сервері результат інтерпретаційного та виконаного PHP-коду - може бути будь-яким типом даних, наприклад, згенерованим HTML. PHP-код може сформувати всю або частину відповіді HTTP.

Існують різні системи веб-шаблонів, системи управління веб-контентом та веб-рамки, які можна використовувати для контролю або полегшення формування такої відповіді.

Крім того, PHP може використовуватися для багатьох завдань програмування поза межами веб-контексту, таких як окремі графічні програми та роботизовані системи управління.

Довільний код PHP також може бути інтерпретований та виконаний через інтерфейс командного рядка (CLI).

Для роботи з PHP був використаний потужний фреймворк Yii. Yes it Is! (Yii) - це об'єктно-орієнтований фреймворк на основі компонентів MVC PHP. [5] Більше про Yii буде описано в наступних розділах.

2.4. База Даних

Для роботи з даними була вибрана система управління реляційними базами даних MySQL. [6]

MySQL здатний тиражувати дані та таблиці розділів для кращої продуктивності та довговічності. Користувачам MySQL не потрібно вивчати нові команди, вони можуть отримати доступ до своїх даних за допомогою стандартних команд SQL.

MySQL надає унікальну зручність у використанні реляційних баз даних та має багато властивостей, які відрізняють її серед її конкуретнів:

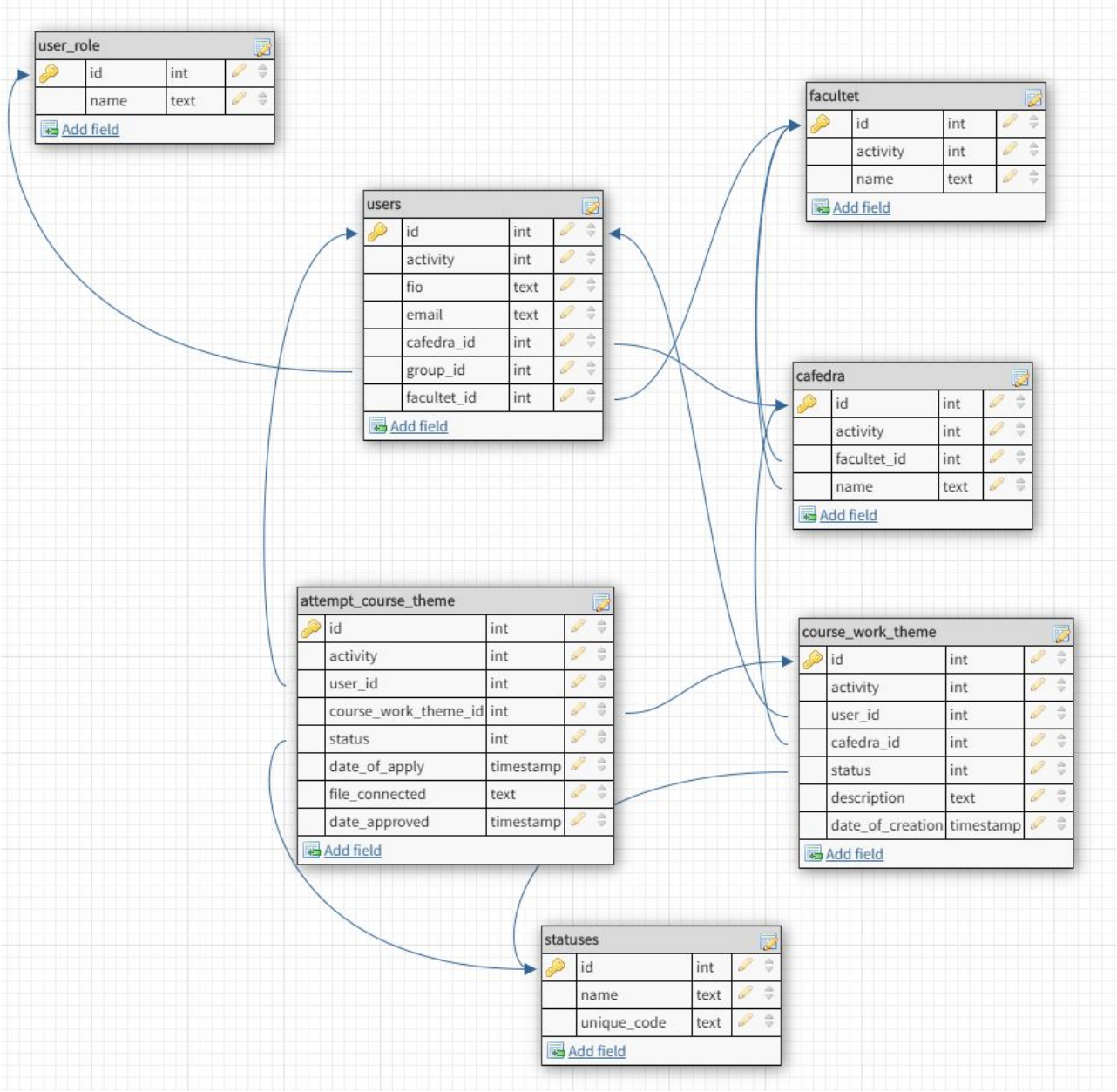
- висока швидкість виконання запитів
- просте встановлення та використання
- можливість збереження великих об’ємів даних
- ефективна система безпеки

Для безпеки MySQL використовує зашифровану систему паролів, яка перевіряється на основі хосту. Клієнти MySQL можуть підключитися до MySQL Server за допомогою декількох протоколів, включаючи сокети TCP / IP на будь-якій платформі. MySQL також підтримує ряд клієнтських та утилітних програм, програм командного рядка та інструментів адміністрування, таких як MySQL Workbench. [7]

Для зручного керування базою даних використовується інструмент phpMyAdmin.

phpMyAdmin - це безкоштовний програмний інструмент, написаний на PHP, призначений для управління та адмініструванням MySQL через Інтернет. phpMyAdmin підтримує широкий спектр операцій на MySQL. Часто використовувані операції (керування базами даних, таблицями, стовпцями, відносинами, індексами, користувачами, дозволами тощо) можуть виконуватися через користувальницький інтерфейс, хоча ви все ще маєте можливість безпосередньо виконувати будь-який оператор SQL. [8]

2.5. Модель Бази Даних



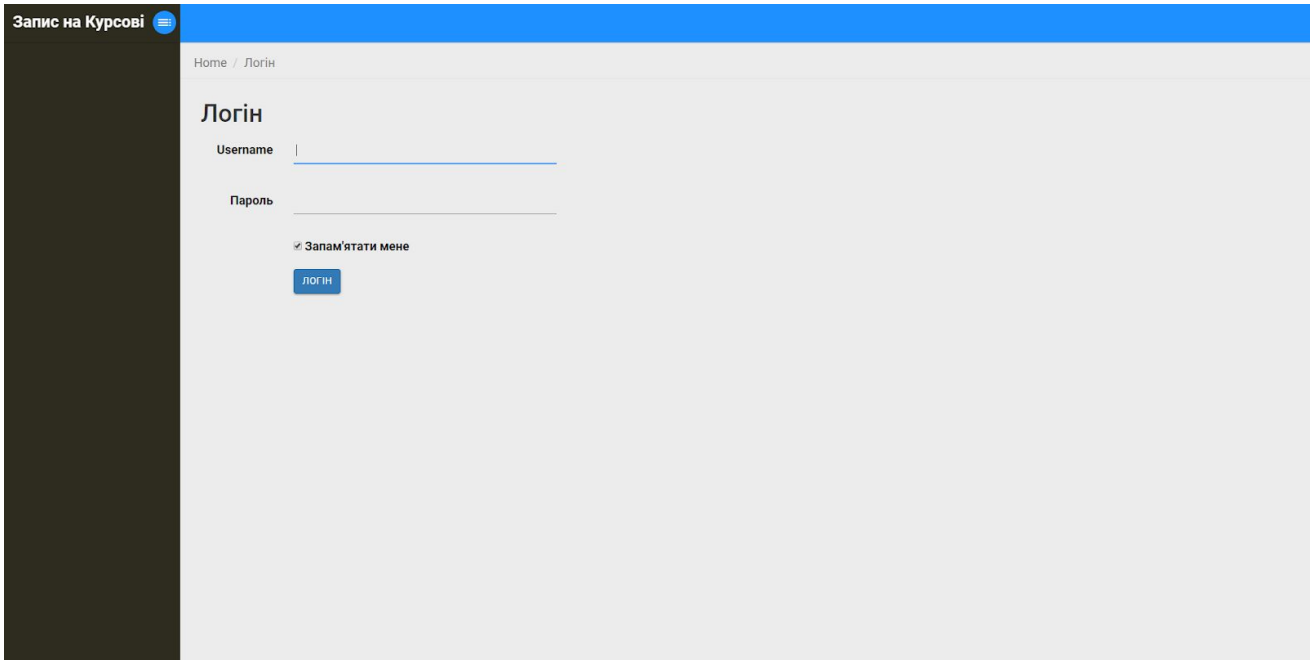
3. Опис практичної частини

3.1. Загальний опис

Система була розроблена на мові програмування PHP, та виконується у вигляді веб-застосунку (сайт).

Кожен Користувач має бути залогінений для доступу до програми. Для логіну потрібно знати свій username та пароль. Для безпеки системи, всі паролі зберігаються захешованими.

Ось вигляд сторінки логіну:



Правила доступу різних ролей Користувачів до окремих сторінок контролюється як через базу даних, так і через код.

Реалізовано захист від перегляду недоступних Тем Курсових Робіт. Наприклад:

- Викладачам від перегляду Тем інших Викладачів
- Студентам від перегляду Тем інших Факультетів

3.2. Архітектура

Архітектура системи розроблена за принципом моделі MVC.

Model - View - controller (MVC) - розділена на три взаємопов’язані частини, що у українському перекладі називаються модель, представлення та контролер. Усі три вищевказані компоненти створені для обробки певних аспектів будь-якої розробки веб-додатків. У розробці програми MVC контролер отримує всі запити потрібні для роботи веб-додатку, а потім інструктує модель підготувати будь-яку інформацію, необхідну для представлення. У представленні використовуються дані, підготовлені контролером, для отримання остаточного результату.

Model - це рівень, який визначає, де зберігаються об’єкти даних програми. Модель нічого не знає про види та контролери. Отже, коли в моделі будуть внесені зміни, вона автоматично сповіщатиме користувачів про внесення змін. Модель може представляти собою один об'єкт або структуру об'єктів.

View - це візуальне зображення моделі MVC. Цей рівень створює інтерфейс, щоб показати фактичний вивід даних користувачеві. Однак view нічого не відображатиме сам по собі. Саме контролер або модель надають дані, що побачить користувач у кінцевому результаті. Він також обробляє запити від користувача та інформує контролер. Представлення підключається до його моделі та отримує необхідні для показу дані, задаючи певні питання. Іноді представлення також може оновлювати модель, надсилаючи відповідні повідомлення.

Controller - це рівень, який діє як мозок всієї системи MVC. Контролер також діє як зв’язок між користувачем та системою. У більшості випадків саме Контролер передає дані представленням для показу. Також він надає користувачеві можливість введення, показуючи йому необхідні представлення з вже підготованими формами, щоб відповідним чином показати їх на екрані. Контролер розуміє введення користувача, перетворює дані на відповіді у належні повідомлення та передає їх представленням.

Головні переваги у використанні цієї моделі:

- Архітектура MVC дозволяє максимально обмежити дублювання коду, оскільки вона відокремлює дані та бізнес-логіку від дисплея.

- У моделі MVC можна створити декілька представлень (з різним дизайном, можливостями) для моделі. Щодня зростає попит на нові способи користування додатками, і для цього розробка MVC, безумовно, чудове рішення.
- Архітектура MVC також може бути інтегрована з JavaScript Framework. Це означає, що додатки MVC можуть працювати з файлами PDF, з віджетами на робочому столі або утворювати повномаштабні інтерактивні сторінки.
- MVC підтримує асинхронну техніку завантаження змісту веб-додатку, яка допомагає розробникам розробити структуру продукту таким чином, що він генерується дуже швидко.
- Для будь-якої веб-програми користувальницький інтерфейс, як правило, змінюється частіше, ніж логіка роботи додатку, як-от зміна кольорів, шрифтів, макетів зовнішнього представлення та додавання нових, унікальних функцій для мобільних телефонів і планшетів. Більше того, додати новий тип представлення дуже просто в MVC-схемі, оскільки частина моделі не залежить від частини view. Тому будь-які зміни у представленні не вплинуть на всю архітектуру продукту.
- Шаблон MVC повертає дані, не застосовуючи форматування. Отже, одні і ті ж компоненти можна використовувати і викликати для використання з будь-яким інтерфейсом. Наприклад, будь-який тип даних може бути відформатований за допомогою HTML, але він також може бути відформатований за допомогою перегляду Macromedia Flash або Dream.
- Платформа MVC підтримує розробку SEO-дружніх веб-сторінок або веб-додатків. За допомогою цієї платформи дуже легко розробити зручні для SEO URL-адреси, щоб генерувати більше відвідувань у певному додатку.

Недоліків у використанні моделі MVC майже немає, але, як і усюди, вони присутні.

Одна з проблем полягає в тому, що структура MVC може бути неймовірно об'ємною. Робота веб-додатку може потребувати великої кількості файлів, щоб все працювало належним чином. Окрім моделей, представлень та контролерів, скоріше за все ви будете використовувати шаблони та файли помічників. Тепер одна веб-сторінка може вимагати до 5 файлів лише для неї. Якщо ви не будете обережні, проекти MVC можуть швидко вийти з ладу. Це також може вплинути

на продуктивність. Тож модель MVC має бути використана лише для масштабних проектів, а не для кількох сторінок веб-додатку.

Зважаючи на плюси та мінуси, модель MVC, безумовно, є чудовим підходом для створення веб-додатків. Перш за все, здатність керувати кількома представленнями робить MVC найкращою схемою архітектури для розробки інтерактивних та легко редагованих проектів. По-друге, проста у реалізації інтеграція з різними бібліотеками та фреймворками дозволяє розробити сучасний веб-додаток з безліччю запозиченими функціями.

3.3. Опис вибраного Фреймворку та його конкурентів

Для написання практичної частини був обраний фреймворк Yii.

Особливості Yii включають:

- Шаблон дизайну model - view - controller (MVC).
- Генерація складних специфікацій сервісу WSDL та управління обробкою запитів веб-служб.
- Інтернаціоналізація та локалізація (I18N та L10N), що включає переклад повідомлень, форматування дати та часу, форматування чисел та локалізацію інтерфейсу.
- Багатошарова схема кешування, яка підтримує кешування даних, сторінок, фрагментів та динамічного вмісту.
- Обробка помилок та ведення логів. Логи можна класифікувати, фільтрувати та перенаправляти до різних напрямків.
- Заходи безпеки включають запобігання міжсайтових сценаріїв (XSS), підробку запитів на різних веб-сайтах (CSRF) та підробку файлів cookie.
- Тестування блоків коду та функціоналу на основі PHPUnit та Selenium.
- Автоматичне генерування коду для скелетів програм та додатків CRUD за допомогою інструмента Gii. Код, згенерований компонентами Yii та інструментами командного рядка, відповідає стандарту XHTML.

Yii - це суто ООП фреймворк, який використовує деякі більш вдосконалені функції PHP, зокрема класи та інтерфейси SPL та анонімні функції. Усі класи

розділені на імена, що дозволяє скористатися їх автоматичним завантажувачем, сумісним з PSR-4. Це означає, що включити HTML-код помічника Yii до вашого веб-застосунку дуже просто:

use yii\helpers\Html;

Не буде помилкою сказати, що безпека є стандартом для Yii. Стандарти безпеки включають в себе перевірку вхідних даних, фільтрування вихідних даних, введення SQL, запобігання виконання коду від запит з інших сайтів та інші. Програма Yii працює на функціях, що робить програму максимально захищеною. Нижче наведено деякі функції:

Аутентифікація - це процес підтвердження особи користувача. Вона працює на основі ідентифікатора (наприклад, імені користувача або адреси електронної пошти) та секретного маркера (наприклад, пароля або маркера доступу) для підтвердження користувача. Аутентифікація є обов'язковим кроком для входу. Yii має розгалужену систему аутентифікації, яка включає спеціальні компоненти для підтримки входу.

Авторизація - це процес перевірки та дозволу, що користувач має доступ та може працювати над певною частиною програми. Yii надає два способи авторизації: Access Control Filter (ACF) та Role-Based Access Control (RBAC). Access Control Filter (ACF) - це простий метод авторизації. Його використовують програми, яким потрібен лише простий контроль доступу, це фільтр дій, який можна використовувати в контролері або модулі. ACF працює, перевіряючи список правил доступу, коли користувач просить виконати дію. Role-Based Access Control (RBAC) забезпечує централізований контроль доступу. Yii реалізує загальну ієрархічну RBAC, яка слідує за моделлю NIST RBAC, що забезпечує функціональність RBAC через компонент програми authManager. Використання RBAC включає два основні етапи: Перший - створення даних авторизації RBAC, а другий крок - використання цих даних авторизації для виконання перевірки доступу в різних місцях, де це потрібно. Клієнти Auth в Yii також надають офіційні розширення, що дозволяють аутентифікувати та авторизувати за допомогою зовнішніх служб.

Механізм криптографії фреймворку Yii ідеально підходить для захисту шифрування важливих даних. Наприклад, коли користувач намагається скинути пароль електронною поштою, він здійснює покроковий механізм генерування маркера, зберігає його в базу даних, надсилає його користувачу електронною поштою, що дозволяє змінити пароль. Важливо, щоб подібні дані - маркер та інші дані були сильно кодовані, щоб зловмисник не міг здогадатися, передбачити

чи розшифрувати їх. У таких ситуаціях Yii генерує псевдовипадкові дані, а також забезпечує функцію підтримки шифрування та дешифрування цих даних за допомогою секретного ключа. Yii також надав функцію підтвердження цілісності даних та перевірки того, що дані не підроблені.

Короткий опис схожих фреймворків:

Laravel - це фреймворк PHP з відкритим кодом. Він, як і Yii, використовує архітектуру веб-додатку MVC. Laravel пропонує широкий набір функціональних можливостей, який включає в себе основні функції фрейму PHP, такі як CodeIgniter, Yii та інші мови програмування, такі як Ruby on Rails. Laravel має дуже багатий набір функцій, який підвищить швидкість веб-розробки. Якщо ви знайомі з Core PHP та Advanced PHP, Laravel полегшить ваше завдання. Це економить багато часу, якщо ви плануєте розробляти веб-сайт з нуля. Більше того, веб-сайт, побудований у Laravel, захищений та запобігає декільком веб-атакам.

Головними перевагами Laravel є:

- Веб-додаток стає більш масштабним та легко редагованим.
- На розробці веб-додатків економиться багато часу, оскільки Laravel використовує компоненти з інших фреймворків при розробці веб-додатків.
- Laravel включає простори імен та інтерфейси, таким чином допомагає впорядкувати та керувати ресурсами.

Codeigniter - це простий, елегантний та потужний набір інструментів з дуже не великим розміром, який використовується тими розробниками, які хочуть створити повнофункціональні веб-програми. CodeIgniter - це програма з відкритим кодом PHP. Він має дуже багатий набір функціональних можливостей, що підвищить швидкість роботи над розробкою веб-сайту.

Ключовими особливостями є:

- Codeigniter дуже простий у налаштуванні, оскільки це структура з відкритим кодом.
- Структура папок, що використовується, лінійна, тому дуже зручна та проста у використанні. Це допомагає в міграції хостингу з одного серверу на інший.
- Codeigniter дозволяє легко організовувати код, що лежить в основі веб-сторінки.

- CodeIgniter використовує систему MVC для спрощення кодування, та створення доцільного веб-додатку.

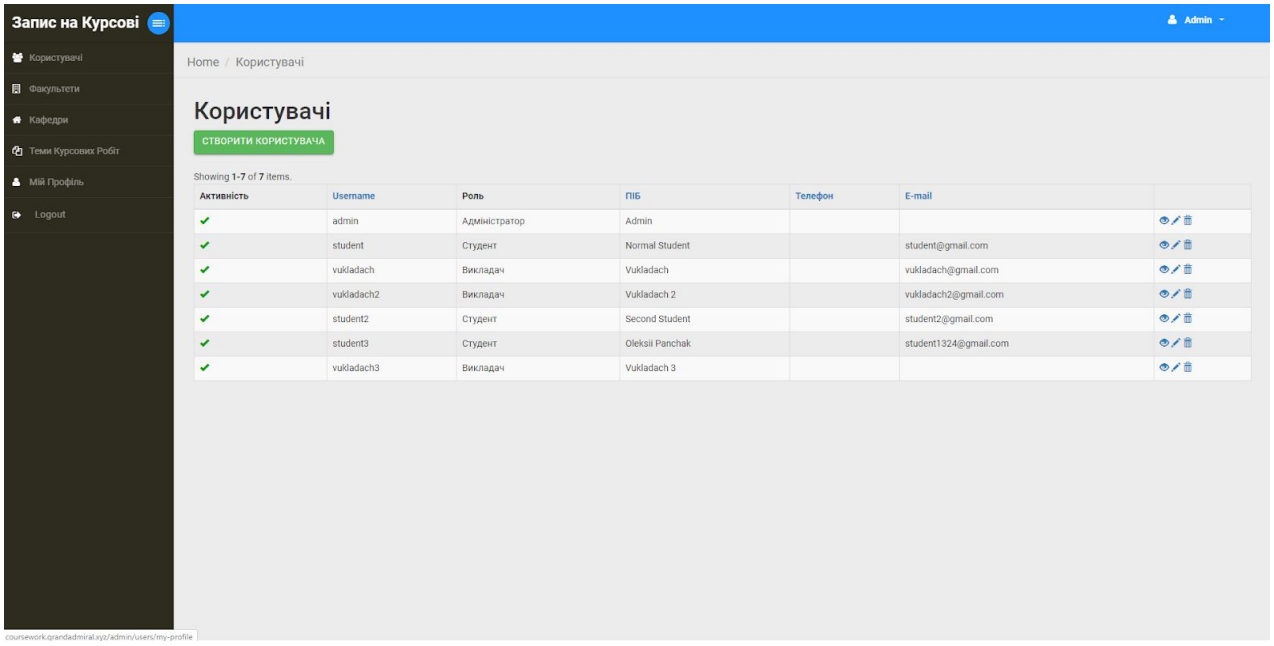
Symfony PHP - це простий у використанні фреймворк завдяки своєму методу програмування Ruby-On-Rails, чистому дизайну та читабельності коду. Symfony пропонує помічники, плагіни Ajax та інтерфейси адміністратора, що робить програмування веб-застосунків справді простими. Розробники можуть зосередитись на застосувальній логіці, не витрачаючи часу на запис файлів конфігурації XML. Symfony використовує використовує архітектуру веб-додатку MVC.

Головними перевагами Symfony є:

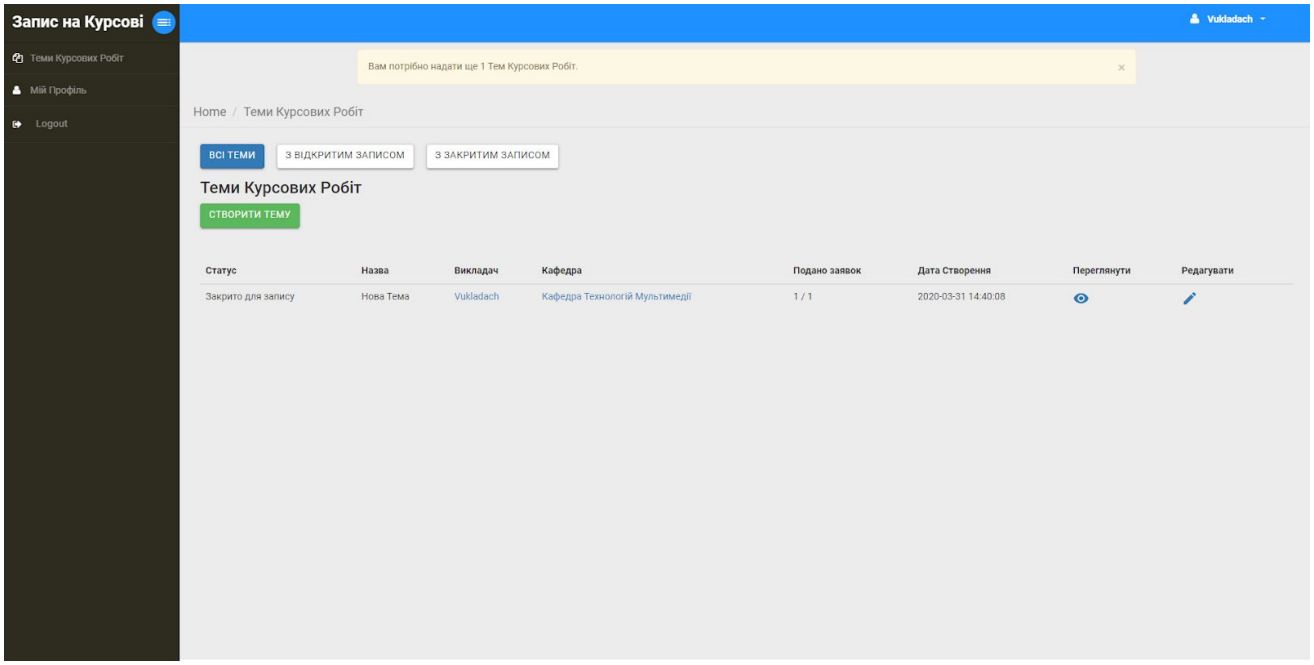
- Засоби генерації коду для прототипування та адміністрування веб-застосунку в один клік.
- Вбудований блок і функціональна структура тестування.
- Дебаг панель, яка прискорює розробку, відображаючи інформацію, необхідну розробнику, на сторінці, над якою він працює.
- Інтерфейс командного рядка, який автоматизує розгортання програми між двома серверами.

3.4. Опис Сторінок та приклад Коду

Управління сайтом виконує користувач з роллю Адміністратор. Він може створювати/редагувати Користувачів, Факультети та Кафедри. У Адміністратора є доступ до сторінок зі списком всіх елементів цих сутностей, ось приклад:

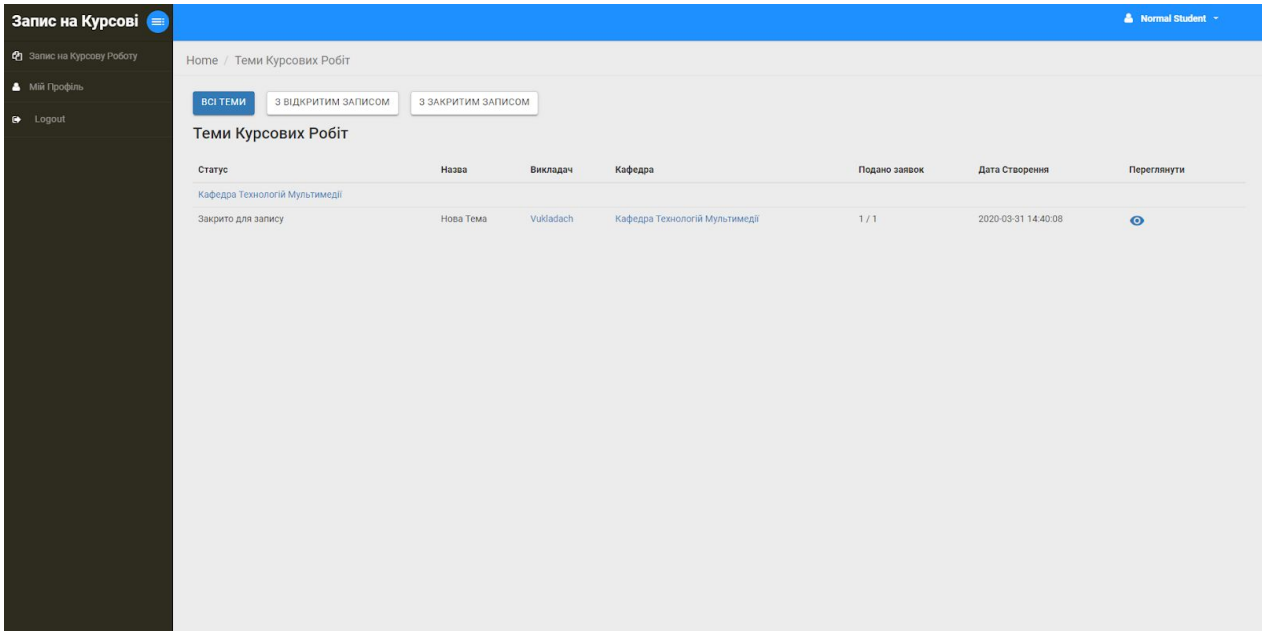


Головною сторінкою користування веб-застосунком для Викладачів є Теми Курсових Робіт. На неї передаються тільки ті теми, які створив поточний Викладач:



Також, у кожного Викладача має бути запропонована обов’язкова кількість Тем для Курсових Робіт, і якщо кількість менше або не дорівнює вже запропонованим, то тоді Викладач отримає нотифікацію, скільки ще Тем потрібно запропонувати. При створенні Теми записується Викладач, який її створив, дата створення та Кафедра.

На цій сторінці Студент бачить всі Теми Курсових Робіт свого Факультету, де може відфільтрувати їх по вже закінченим для запису, або ще відкритим. Для Студента Теми приходять відсортовані - по Кафедрам.



Для перевірки, Користувач якої Ролі використовує у даний момент веб-додаток було написано три функції, які за рахунок розширення базового класу Користувача, перевіряють його Роль, та повертають true або false:

```
function isAdmin() {
    if (Yii::$app->user->identity->role_id == 1) {
        return true;
    }
    return false;
}

function isStudent() {
    if (Yii::$app->user->identity->role_id == 2) {
        return true;
    }
    return false;
}

function isVukladach() {
    if (Yii::$app->user->identity->role_id == 3) {
        return true;
    }
    return false;
}
```

Приклад використання цих функцій у веб-додатку:

```
if (isAdmin()) {
            if ($_GET['type'] == 'open') {
                        //пошук всіх тем з відкритим записом для показу
Адміністратору
                    $items = Theme::find()->asArray()->where(['status' =>
'1'])->all();
            } else if ($_GET['type'] == 'closed') {
                //пошук всіх тем з закритим записом для показу Адміністратору
                    $items = Theme::find()->asArray()->where(['status' =>
'2'])->all();
            } else {
                //пошук всіх тем для показу Адміністратору
                $items = Theme::find()->asArray()->all();
            }
        } else if (isVukladach()) {
            if ($_GET['type'] == 'open') {
                  //пошук всіх тем з відкритим записом для показу Викладачу,
які він створив
                    $items = Theme::find()->asArray()->where(['user_id' =>
getid(), 'status' => '1'])->all();
            } else if ($_GET['type'] == 'closed') {
```

```
                //пошук всіх тем з закритим записом для показу Викладачу, які
він створив
                    $items = Theme::find()->asArray()->where(['user_id' =>
getid(), 'status' => '2'])->all();
            } else {
                //пошук всіх тем для показу Викладачу, які він створив
                    $items = Theme::find()->asArray()->where(['user_id' =>
getid()])->all();
            }

                //перевірка, чи кількість обов'язкових Тем Курсових Робіт
дорівнює створеній
                $items_check = Theme::find()->asArray()->where(['user_id' =>
getid()])->all();
                    if (Yii::$app->user->identity->needed_course_themes >
count($items_check)) {
                $amount = Yii::$app->user->identity->needed_course_themes -
count($items_check);
                 $warning_message = 'Вам потрібно надати ще '.$amount.' Тем
Курсових Робіт.';
                Yii::$app->session->setFlash('warning', $warning_message);
            }
        } else if (isStudent()) {
            //пошук всіх категорій Факультету, до якого належить Студент
                    $cafedras = Cafedra::find()->asArray()->select(['id',
'name'])->where(['faculty_id' => Yii::$app->user->identity->faculty_id,
'activity' => '1'])->all();
            $items = [];
            foreach ($cafedras as $cafedra) {
                if ($_GET['type'] == 'open') {
                    //пошук Тем з відкритим записом для Студента для кожної
Кафедри
                    $themes = Theme::find()->asArray()->where(['cafedra_id'
=> $cafedra['id'], 'status' => '1'])->all();
                } else if ($_GET['type'] == 'closed') {
                    //пошук Тем з закритим записом для Студента для кожної
Кафедри
                    $themes = Theme::find()->asArray()->where(['cafedra_id'
=> $cafedra['id'], 'status' => '2'])->all();
                } else {
                    //пошук Тем для Студента для кожної Кафедри
                    $themes = Theme::find()->asArray()->where(['cafedra_id'
=> $cafedra['id']])->all();
```

```
                }

                    //перевірка, чи є теми у Кафедри, чи потрібно виводити її
назву
                if ($themes) {
                        $cafedra_array = array('cafedra_id' => $cafedra['id'],
'cafedra' => $cafedra['name']);
                    $items[] = $cafedra_array;
                }

                foreach ($themes as $theme) {
                    $items[] = $theme;
                }
            }
        }
```

Висновки

Було реалізовано:

- Повномасштабний веб-застосунок, який майже готовий для повноцінного запуску та використання;
- Серверна частина;
- Інтерфейс програми;
- SQL база даних;
- Понад 2500 слів теорії для обґрунтування практичної частини;

Список використаної літератури

1. https://jquery.com/
2. https://getbootstrap.com/
3. https://materializecss.com/about.html
4. https://www.php.net/
5. https://www.yiiframework.com/
6. https://www.mysql.com/
7. https://www.mysql.com/products/workbench/
8. https://www.phpmyadmin.net/