**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА**

**ФАКУЛЬТЕТ УПРАВЛІННЯ ФІНАНСАМИ ТА БІЗНЕСУ**

**Кафедра цифрової економіки та бізнес-аналітики**

**КУРСОВА РОБОТА**

**з навчальної дисципліни**

**“Проектування та адміністрування БД і СД”**

**на тему:**

**«Інформаційна система для онлайн-школи»**

Галузь знань: 05 «Соціальні та поведінкові науки»

Спеціальність: 051 «Економіка»

Спеціалізація: «Інформаційні технології в бізнесі»

Освітній ступінь: бакалавр

**Науковий керівник:         Виконавець:**

к.ф.-м.н., доц. Депутат Б.Я.Демчина Х. В.        

           (прізвище, ім’я, по-батькові)                                 (прізвище, ім’я, по-батькові)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (підпис)                  УФЕ-31с        група

                                                                                \_\_\_\_\_\_\_\_\_\_\_\_\_\_ (підпис)

**“\_\_\_” \_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 р.                               “\_\_\_” \_\_\_\_\_\_\_\_\_\_\_\_ 2021 р.**

**Загальна кількість балів** \_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис, ПІП членів комісії)

**Зміст**

[РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ 5](#_Toc73007070)

[1.1. Постановка задачі 6](#_Toc73007071)

[1.2. Розробка моделі 6](#_Toc73007072)

[1.3. Аналіз засобів реалізації 7](#_Toc73007073)

[РОЗДІЛ 2. РОЗРОБКА БАЗИ ДАНИХ 9](#_Toc73007074)

[2.1. Що таке реляційні бази даних? 9](#_Toc73007075)

[2.2. Принципи нормалізації баз даних та логічні зв’язки 10](#_Toc73007076)

[2.3. Опис моделі даних 13](#_Toc73007077)

[2.4. Визначення типів даних 15](#_Toc73007078)

[2.5. Реалізація SQL-скрипту 18](#_Toc73007079)

[РОЗДІЛ 3. РОЗРОБКА ВЕБ-ДОДАТКУ 23](#_Toc73007080)

[3.1. Структура веб-сайту 23](#_Toc73007081)

[3.2. Макет веб-сторінки 24](#_Toc73007082)

[3.3. Програмування клієнтської частини 24](#_Toc73007083)

[ВИСНОВОК 30](#_Toc73007084)

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 31](#_Toc73007085)

Вступ

**Актуальність проблеми.** В даний час швидко розвиваються нові інформаційні технології (далі ІТ), які тягнуть за собою корінні зміни не тільки у виробничих галузях, але і в сфері пізнавальної діяльності, зокрема в освіті. Сучасний період розвитку суспільства характеризується сильним впливом на нього комп'ютерних технологій, які проникають в усі сфери людської діяльності, забезпечують поширення інформаційних потоків в суспільстві, утворюючи глобальний інформаційний простір. Невід'ємною і важливою частиною цих процесів є онлайн-освіта, особливо в період карантину, коли всі змушені перейти на дистанційну форму навчання. Інтернет-освіта дає нам безліч можливостей, напрямів та форм навчання, зокрема це онлайн-курси. Кожен може спробувати себе у різній сфері, витрачаючи мінімум часу та зусиль, щоб вивчати нову справу. За статистикою, кожен третій українець мав необхідність проходити додаткове навчання протягом останніх трьох років.

Переваги онлайн-освіти:

* Доступ до програм найкращих викладачів світу: онлайн-освіта дає можливість навчатися в тих, хто перебуває територіально далеко.
* Найновіша інформація, технології, теорії: матеріал онлайн-курсів оновлюється динамічніше, ніж офлайнові програми.
* Навчання безкоштовне або доступніше за ціною, ніж денне навчання в університеті.
* Можливість навчатись будь-де й будь-коли.
* Можливість поєднувати різні формати та підходи до навчання.

Навчання на відстані — часто вже навіть не вибір людини, а вимушений захід. Тому виникає потреба в створенні інформаційної системи для онлайн-шкіл, які мають на меті навчати людей тої чи іншої сфери в дистанційному режимі. Так, користувач з легкістю зможе знайти потрібний йому курс на просторах Інтернету, ознайомитись з програмою курсу, ціною та іншою інформацією, а також придбати чи скачати курс та навчатись, не виходячи з дому.

**Мета та завдання.** Метою даної роботи є обгрунтування теоретичних основ та реалізації інформаційної системи на прикладі сайту онлайн-освіти. Для досягнення мети курсової роботи, необхідно вирішити такі завдання:

* коротко охарактеризувати інформаційну систему для сайту онлайн освіти;
* проаналізувати предметну область;
* спроектувати структуру бази даних;
* вибрати мову програмування та технології для програмної реалізації описаного продукту;
* розробити веб-сайт «Онлайн-курси» для продажу пакетів онлайн-освіти.

**Об’єкт дослідження**. Особливість розробки інформаційної системи для продажу пакетів онлайн освіти за допомогою системи управління базами даних MySQL.

**Предмет досліджень**. Застосування технологій створення веб-орієнтованого програмного забезпечення для розробки веб-сайту «Онлайн-курси».

**Практичне значення отриманих результатів.** Програма може використовуватись юридичною особою, яка хоче розмістити інформацію про свій пакет онлайн освіти в мережі інтернет з метою залучення клієнтів.

**Використане програмне забезпечення**. В процесі розробки використовуватимуться наступні мови програмування та технології:

* Java - об'єктно-орієнтована мова програмування.
* HTML - мова тегів, якою пишуться гіпертекстові документи для мережі Інтернет.
* CSS - спеціальна мова стилю сторінок, що використовується для опису їхнього зовнішнього вигляду.
* Bootstrap - безкоштовний набір інструментів з відкритим кодом, призначений для створення веб-сайтів та веб-додатків, який містить шаблони CSS та HTML для типографіки, форм, кнопок, навігації та інших компонентів інтерфейсу, а також додаткові розширення JavaScript.
* MySQL - вільна система керування реляційними базами даних.

**Структура роботи**. Курсова робота складається з трьох розділів(«Аналіз предметної області», «Розробка бази даних» та «Розробка веб-додатку»), висновків, списку використаних джерел та додатків.

Загальний обсяг роботи − 35 сторінок.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

**1.1. Постановка задачі**

Основним завданням даної роботи є створення повноцінної інформаційної системи для онлайн-школи продажу курсів. Функціональна структура ІС має орієнтуватися на ті інформаційні потреби кінцевих користувачів, які змінюються в умовах ринку, та відображати зміст і специфіку функцій управління конкретним економічним об’єктом. Вимоги, які пред’являються користувачами до ІС, це: гнучка структура і відкритість системи, тобто можливість вносити необхідні зміни у розроблену модель і забезпечувати нарощування функціональних можливостей в міру необхідності. Отож, в даній роботі нам слід створити ІС для онлайн-школи у вигляді веб-сайту.

Зокрема, повинна бути головна сторінка – де буде розміщена головна інформація про курс, який пропонує дана онлайн-школа. Необхідно створити розділ, на якому буде розміщений перелік/опис послуг які надає дана організація. Також веб-сайт повинен містити розділ з контактними даними та цінами. Всі сторінки необхідно пов’язати між собою за допомогою меню ( „шапки” сторінки). Дані про клієнтів необхідно внести в базу даних та при створені сторінки використовувати не сталі дані, а дані з бази даних. При оновлені даних у базі даних (БД) необхідно щоб вони автоматично оновлювались на сторінці сайту. Також даний сайт повинен містити форму зворотнього звязку. Тобто щоб клієнт переглянувши запропоновані йому послуги міг онлайн заповнити заявку вказавши свої контактні дані емейл чи телефон, щоб адміністрація школи могла зв'язатись з клієнтом.

**1.2. Розробка моделі**

Система онлайн-школи має три елементи керування:

* Будь-який користувач;
* Авторизований користувач;
* Адміністратор;

Не зареєстрований користувач має змогу переглядати головну сторінку, щоб переглянути список можливих курсів та взяти участь, користувач повинен зареєструватись. Головний користувач сайту – це адміністратор. Він має змогу наповнювати БД системи новою інформацією, тобто даними про курси, що з’явилися, їх опис тощо, а також додавати чи видаляти користувачів.

**1.3. Аналіз засобів реалізації**

Для реалізації даної роботи було вибрано наступні технології:

1. HTML5, СSS3, Bootstrap (дані технології для реалізації Front-end частини);
2. Java Spring Boot, MySQL(для реалізації Back-end частини).

Bootstrap було використано, тому що ця технологія надає такі переваги:

1. Адаптивність (Bootstrap дозволяє створювати адаптивні сайти, тобто дизайн сайту буде коректно відображатися на екранах пристроїв різних розмірів незалежно від їх діагоналі);
2. Кросбраузерність (сайти, зроблені з використанням Bootstrap, будуть однаково відображатися у всіх сучасних браузерах);
3. Легкість у використанні і швидкість в освоєнні (Bootstrap легко використовувати в розробці, в ньому легко розібратися, а ті, хто ще не знайомий з Bootstrap, зможуть швидко навчитися працювати з цим інструментом);
4. Зрозумілий код (Bootstrap дозволяє писати якісний і зрозумілий код, який легко зрозуміє інший розробник, що значно спрощує розробку в команді.).

Для Back-end частини було обрано Spring Boot, тому що ця технологія надає такі переваги:

1. Spring надає каркас для майбутньої аплікації, щось на подобі заготовки, в якій вже є готова архітектура і необхідно вбудувати бажану функціональність;
2. В аплікаціях на основі Spring об’єкти слабозв’язані завдяки використанню Dependency Injection (відповідно, немає потреби зв’язувати об’єкти вручну – Spring виконує це сам завдяки аннотаціям);
3. Spring Boot використовує starters, які дозволяють писати код швидко та легко. Вони застосовуються методом вставки готового коду, так званої dependency в файлі pom.xml, що дозволяє з легкістю додавати нові шаблони та видаляти непотрібні (наприклад, spring-boot-starter-jpa надає готовий комплект для використання JPA);
4. Легко взаємодіє з Bootstrap, який включається в конфігурацію, що покращує читабельність коду та збільшує швидкодію.

**РОЗДІЛ 2. РОЗРОБКА БАЗИ ДАНИХ**

**2.1. Що таке реляційні бази даних?**

**База даних** — це інтегрована сукупність структурованих і взаємозалежних даних, організована за певними правилами, які передбачають загальні принципи опису, зберігання і обробки даних.

**Реляційна (від анг. relation) база даних**— це тип бази даних, що зберігає інформацію в електронних таблицях і здійснює пошук даних в одній таблиці на підставі визначених ключових полів іншої таблиці. Реляційна база даних є сукупністю елементів даних, організованих у вигляді набору формально описаних таблиць, з яких дані можуть бути доступними або повторно зібрані багатьма різними способами без необхідності реорганізації таблиць бази даних. Використання реляційних БД було запропоноване Едгаром Коддом в 1970 році.

Для того щоб створити власну СУБД, слід скористатися одним з інструментів моделювання, продумати, з якою інформацією вам необхідно працювати, спроектувати таблиці і реляційні одно- і множинні зв'язки між даними, заповнити осередки сутностей і встановити первинний, зовнішні ключі.

Моделювання таблиць і проектування реляційних баз даних здійснюється за допомогою безкоштовних інструментів, таких як Workbench, PhpMyAdmin, Case Studio, dbForge Studio. Після детального проектування слід зберегти графічно готову реляційну модель і перевести її в готовий SQL-код. На цьому етапі можна починати роботу з сортуванням даних, їх обробку і систематизацію.

Реляційна база даних складається з наступних компонентів:

1. Сутність. Таблиця реляційної бази даних може бути одна, а може бути цілий набір з таблиць, які характеризують описані об'єкти. У них фіксована кількість полів і змінне число записів. Таблиця реляційної моделі баз даних складається з рядків, атрибутів і макета.
2. Запис - змінне число рядків, що відображають дані, що характеризують описуваний об'єкт. Нумерація записів проводиться системою автоматично.
3. Атрибути - дані, що демонструють собою опис стовпців суті.
4. Поле. Являє собою стовпець суті. Їх кількість - фіксована величина, що встановлюється під час створення або зміни таблиці.

Тепер, знаючи складові елементи таблиці, можна переходити до властивостей реляційної моделі баз даних:

* Суті реляційної БД двовимірні. Завдяки цій властивості з ними легко проробляти різні логічні і математичні операції.
* Порядок проходження значень атрибутів і записів в реляційній таблиці може бути довільним.
* Стовпець в межах однієї реляційної таблиці повинен мати свою індивідуальну назву.
* Всі дані в стовпці суті мають фіксовану довжину і однаковий тип.
* Будь-який запис по суті вважається одним елементом даних.
* Складові компоненти рядків єдині в своєму роді. У реляційної сутності відсутні однакові рядки.

Виходячи з властивостей реляційної СУБД, зрозуміло, що значення атрибутів повинні бути однакового типу, довжини.

**2.2. Принципи нормалізації баз даних та логічні зв’язки**

Нормалізація - це процес ефективної організації даних у базі даних. Є дві цілі процесу нормалізації: усунення зайвих даних (наприклад, зберігання одних і тих же даних у більш ніж одній таблиці) та забезпечення залежності даних має сенс (лише зберігання пов’язаних даних у таблиці). Обидва ці цілі - це гідні цілі, оскільки вони зменшують обсяг місця, який займає база даних, і забезпечують логічне зберігання даних.

Спільнота баз даних розробила низку вказівок щодо забезпечення нормалізації баз даних. Вони називаються нормальними формами і нумеруються від однієї (найнижча форма нормалізації, яка називається першою нормальною формою або 1NF) через п'ять (п'ята нормальна форма або 5NF). У практичних програмах часто зустрічаються 1NF, 2NF і 3NF, а також епізодичні 4NF. П'ята нормальна форма зустрічається дуже рідко .

Важливо зазначити, що принципи нормалізації є лише вказівками. Іноді виникає необхідність відхилятися від них, щоб задовольнити практичні вимоги бізнесу. Однак коли виникають зміни, надзвичайно важливо оцінити всі можливі наслідки, які вони могли б мати у вашій системі, та врахувати можливі невідповідності.

Перша нормальна форма (1NF) встановлює дуже основні правила для організованої бази даних:

* Видалити дублікати стовпців з тієї ж таблиці.
* Створити окремі таблиці для кожної групи пов’язаних даних та визначити кожен рядок унікальним стовпцем або набором стовпців (первинний ключ).

Друга нормальна форма (2NF) додатково стосується концепції видалення дублюючих даних:

* Відповідати всім вимогам першої нормальної форми.
* Видалити підмножини даних, які застосовуються до декількох рядків таблиці, і помістити їх в окремі таблиці.
* Створити зв’язки між цими новими таблицями та їх попередниками за допомогою використання зовнішніх ключів.

Третя нормальна форма (3NF) йде на один великий крок далі:

* Відповідати всім вимогам другої нормальної форми.
* Видалити стовпці, які не залежать від первинного ключа.

Нормальна форма Бойса-Кодда додає ще одну вимогу:

* Відповідати всім вимогам третьої нормальної форми.
* Кожна детермінанта повинна бути ключовим кандидатом.

Нарешті, четверта нормальна форма (4NF) має ще одну додаткову вимогу:

* Відповідати всім вимогам третьої нормальної форми.
* Відношення є в 4NF, якщо воно не має багатозначних залежностей.

Ці норми щодо нормалізації є сукупними. Щоб база даних знаходилася в 2NF, вона повинна спочатку виконати всі критерії бази 1NF.

Хоча нормалізація бази даних часто є хорошою ідеєю, це не є абсолютною вимогою. Насправді, є деякі випадки, коли навмисне порушення правил нормалізації є хорошою практикою.

Зв’язок встановлюється між двома загальними полями (стовпцями) двох таблиць. Існують зв’язки з відношенням «один-до-одного»*,*«один-до-багатьох»і«багато-до-багатьох».

Відносини, які можуть існувати між записами двох таблиць:

1. **один — до — одного** — кожному запису з однієї таблиці відповідає один запис у іншій таблиці;
2. **один — до — багатьох** — кожному запису з однієї таблиці відповідає кілька записів у іншій таблиці;
3. **багато — до — одного** — безлічі записів з однієї таблиці відповідає один запис у іншій таблиці;
4. **багато — до — багатьох** — безлічі записів з однієї таблиці відповідає кілька записів в іншій таблиці.

Відношення «один-до-багатьох» створюється в тому випадку, коли тільки одне з полів є полем первинного ключа або унікального індексу. Відношення «один-до-одного» створюється в тому випадку, коли обидва поля є ключовими або мають унікальні індекси. Відношення «багато-до-багатьох» фактично є двома відносинами «один-до-багатьох» з третьої таблицею, первинний ключ якої складається з полів зовнішнього ключа двох інших таблиць.

**Ключ** — це стовпець (може бути декілька стовпців), що додається до таблиці і дозволяє встановити зв’язок із записами в іншій таблиці. Існують ключі двох типів: первинні і вторинні (зовнішні). **Первинний ключ** — це одне або кілька полів (стовпців), комбінація значень яких однозначно визначає кожний запис у таблиці. Первинний ключ не допускає значень null і завжди повинен мати унікальний індекс. Первинний ключ використовується для зв’язування таблиці з зовнішніми ключами в інших таблицях. **Зовнішній (вторинний) ключ** — це одне або кілька полів (стовпців) у таблиці, що містять посилання на поле або поля первинного ключа в іншій таблиці. Зовнішній ключ визначає спосіб об’єднання таблиць.

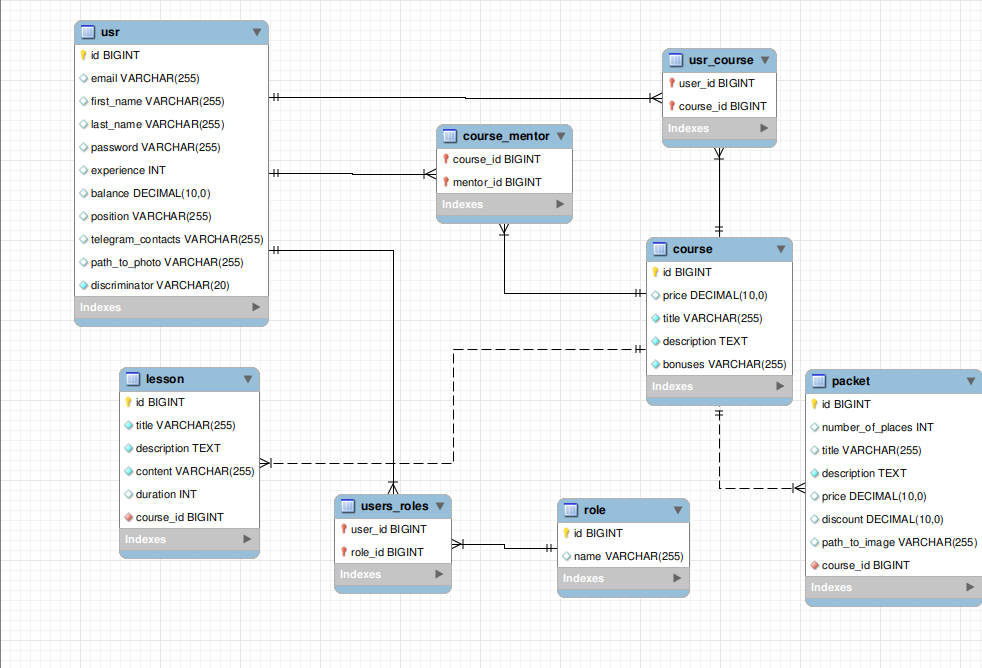
З двох логічно пов’язаних таблиць одну називають таблицею первинного ключа або головною таблицею, а іншу таблицею вторинного (зовнішнього) ключа або підпорядкованою таблицею. СУБД дозволяють зіставити споріднені записи з обох таблиць і спільно вивести їх у формі, звіті або запиті.

**2.3. Опис моделі даних**

Для розробки своєї бази даних було обрано саме реляційну модель через ряд її переваг, а саме:

1. вона проста і доступна для розуміння користувачем; єдиною використовуваною інформаційною конструкцією є «таблиця»;
2. суворі правила проектування, які базуються на математичному апараті;
3. повна незалежність даних; зміни в прикладній програмі при зміні реляційної БД мінімальні;
4. для організації запитів і написання прикладного ПЗ немає необхідності знати конкретну організацію БД у зовнішній пам'яті.

У ході практичного завдання була побудована наступна модель бази даних (Рис.2.3.1).

Рис. 2.3.1. Діаграма бази даних

Можна побачити, що дана модель складається з восьми таблиць: usr (дані користувачів), role(інформація про ролі), course (інформація про курс), packet (інформація про пакети курсів), lesson (інформація про уроки курсів) course\_mentor (інформація про викладачів курсів), user\_roles (ролі користувачів), usr\_course(інформація про куплені курси користувачами). Таблиця користувачів складається з id (первинного ключа сутності), набору даних про користувача. Таблиця курсів міститть id (первинний ключ), набір даних. Таблиця usr\_course є таблицею, яка зв’язує користувача і курс. Вона містить складений первинний ключ(user\_id, course\_id) та два зовнішні ключі user\_id та course\_id.

Також можна побачити зв’язки між таблицями:

* usr — role - багато-до-багатьох
* course — usr - багато-до-багатьох ( куплені курси в таблиці usr\_course )
* course — usr — багато-до-багатьох (інформація про менторів на курсах )
* course — lesson - один-до-багатьох
* course — packet - один до-багатьох

Важливо зазначити, що база даних розбита на декілька логічних зі змістом таблиць. Таблиці з’єднані між собою за допомогою зовнішніх ключів, і містять унікальні поля та описують конкретні сутності. З цього можна зробити висновок, що дана модель відповідає принципам нормалізації.

**2.4. Визначення типів даних**

При визначенні таблиці для всіх її стовпців необхідно вказати тип даних. Тип даних визначає діапазон значень, які можуть зберігатися в стовпці, скільки вони будуть займати місця в пам'яті. MySQL підтримує широку палітру різних типів даних, які умовно можна розділити на підгрупи: чисельні, символьні, логічні, дата і час, бінарні і ряд інших.

Числові типи даних:

* serial: представляє числове значення, яке автоінкрементується та займає 4 байти і може зберігати числа від 1 до 2147483647; значення даного типу утворюється шляхом автоінкремента значення попереднього рядка; тому, як правило, даний тип використовується для визначення ідентифікаторів рядків;
* smallserial: представляє числове значення, яке автоінкрементується та займає 2 байти і може зберігати числа від 1 до 32767; аналог типу serial для невеликих чисел;
* bigserial: представляє числове значення, яке автоінкрементується та займає 8 байтів і може зберігати числа від 1 до 9223372036854775807; аналог типу serial для великих чисел;
* smallint: зберігає числа від -32768 до +32767; займає 2 байти; має псевдонім int2;
* integer: зберігає числа від -2147483648 до +2147483647; займає 4 байти; має псевдоніми int і int4;
* bigint: зберігає числа від -9223372036854775808 до +9223372036854775807; займає 8 байтів; має псевдонім int8;
* numeric: зберігає числа з фіксованою точністю, які можуть мати до 131072 знаків в цілій частині і до 16383 знаків після коми; даний тип може приймати два параметри precision і scale: numeric (precision, scale); параметр precision вказує на максимальну кількість цифр, які може зберігати число; параметр scale представляє максимальну кількість цифр, які може містити число після коми; це значення повинно знаходитися в діапазоні від 0 до значення параметра precision; за замовчуванням воно дорівнює 0; наприклад, для числа 23.5141 precision дорівнює 6, а scale - 4.
* decimal: зберігає числа з фіксованою точністю, які можуть мати до 131072 знаків в цілій частині і до 16383 знаків у дробовій частині; те ж саме, що і numeric.
* real: зберігає числа з плаваючою точкою з діапазону від 1E-37 до 1E + 37. Займає 4 байти; має псевдонім float4.
* double precision: зберігає числа з плаваючою точкою з діапазону від 1E-307 до 1E + 308; займає 8 байт; має псевдонім float8.

Символьні типи:

* character (n): представляє рядок з фіксованої кількості символів; за допомогою параметра задається кількість символів в рядку; має псевдонім char (n);
* character varying (n): представляє рядок з фіксованої кількості символів; за допомогою параметра задається кількість символів в рядку; має псевдонім varchar (n);
* text: представляє текст довільної довжини.

Для роботи с грошовими одиницями визначений тип money, який може приймати значення в діапазоні від -92233720368547758.08 до +9223372036854758.07 і займає 8 байтів.

Для зберігання бінарних даних визначено тип bytea. Він зберігає дані у вигляді бінарних рядків, які представляють послідовність октетів або байт.

Типи для роботи з датами і часом:

* timestamp: зберігає дату і час; займає 8 байтів; для дат найнижче значення - 4713 р. до н.е., найбільш верхнє значення - 294 276 р. н.е.;
* timestamp with time zone: те ж саме, що і timestamp, тільки додає дані про часовий пояс;
* date: представляє дату від 4713 р. до н.е. до 5874897 р. н.е.; займає 4 байти;
* time: зберігає час з точністю до 1 мікросекунди без вказівки часового поясу; приймає значення від 00:00:00 до 24:00:00; займає 8 байтів;
* time with time zone: зберігає час з точністю до 1 мікросекунди із зазначенням часового поясу; приймає значення від 00:00:00 +1459 до 24:00:00-1459; займає 12 байт;
* interval: представляє часовий інтервал; займає 16 байт.

Логічний тип boolean може зберігати одне з двох значень: true або false. Замість true можна вказувати наступні значення: TRUE, 't', 'true', 'y', 'yes', 'on', '1'. Замість false можна вказувати наступні значення: FALSE, 'f', 'false', 'n', 'no', 'off', '0'.

Типи для подання інтернет-адрес:

* cidr: інтернет-адреса в форматі IPv4 і IPv6 (наприклад, 192.168.0.1.); займає від 7 до 19 байтів;
* inet: інтернет-адреса в форматі cidr / y, де cidr це адреса в форматі IPv4 або IPv6, а /y - кількість біт в адресі (якщо цей параметр не вказано, то використовується 34 для IPv4, 128 для IPv6) (наприклад, 192.168.0.1/24 або 2001: 4f8: 3: ba: 2e0: 81ff: ​​fe22: d1f1 / 128); займає від 7 до 19 байтів.
* macaddr: зберігає MAC-адресу; займає 6 байтів;
* macaddr8: зберігає MAC-адресу у форматі EUI-64; займає 8 байтів.

Геометричні типи:

* point: представляє точку на площині в форматі (x, y); займає 16 байтів;
* line: представляє лінію невизначеної довжини в форматі {A, B, C}; займає 32 байти;
* lseg: представляє відрізок в форматі ((x1, y1), (x2, y2)); займає 32 байти;
* box: представляє прямокутник в форматі ((x1, y1), (x2, y2)); займає 32 байти;
* path: представляє набір з’єднаних точок; у форматі ((x1, y1), ...) шлях є закритим (перша і остання точка з'єднуються лінією) і фактично являє багатокутник; у форматі [(x1, y1), ...] шлях є відкритим; займає 16 + 16n байтів;
* polygon: представляє многокутник в форматі ((x1, y1), ...); займає 40 + 16n байтів;
* circle: представляє круг в форматі <(x, y), r>; займає 24 байти.

Решта типів даних:

* json: зберігає дані json в текстовому вигляді;
* jsonb: зберігає дані json в бінарному форматі;
* uuid: зберігає універсальний унікальний ідентифікатор (UUID), наприклад, a0eebc99-9c0b-4ef8-bb6d-6bb9bd380a11, займає 32 байти;
* xml: зберігає дані в форматі XML.

В даному випадку використовуються наступні типи даних: serial – для первинних та зовнішніх ключів, тобто для ідентифікаторів; int2 – для поля user.age; money – для поля course.price; varchar(n) – для текстових полів (наприклад, description або email).

**2.5. Реалізація SQL-скрипту**

Раніше Java використовувала JDBC API(Java Database Connection) для менеджменту базою даних. Проте даний спосіб вимагає писати однотипний і непотрібний “обслуговуючий” код для тривіальних операцій зберігання Java об’єктів в базу даних і навпаки. Рішенням даної проблеми є ORM – Object Relational Mapping, тобто об’єктно-реляційне відображення. Це технологія програмування, яка зв'язує бази даних з концепціями об'єктно-орієнтованих мов програмування. Якщо спростити, то ORM це зв'язок Java об'єктів і записів в БД(Рис.2.5.1).

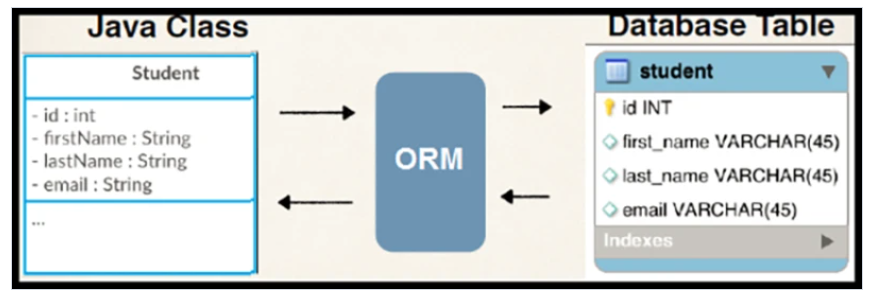


Рис.2.5.1. Ілюстрація концепції ORM

ORM - це по суті концепція про те, що Java об'єкт можна представити як дані в БД (і навпаки). Вона знайшла втілення у вигляді специфікації JPA - Java Persistence API. Специфікація - це вже опис Java API, яке виражає цю концепцію. Специфікація розповідає, якими засобами ми повинні бути забезпечені (тобто через які інтерфейси ми зможемо працювати), щоб працювати по концепції ORM. І як ці засоби використовувати. Реалізацію засобів специфікація не описує. Це дає можливість використовувати для однієї специфікації різні реалізації. Можна спростити і сказати, що специфікація - це опис API.

В даному випадку для того, щоб використовувати JPA API в проекті, реалізованому за допомогою Spring Boot, досить додати в залежності (файл pom.xml) бібліотеку, а точніше стартер spring-boot-starter-jpa, а також вказати бібліотеку бази даних, яку ми використовуєм, тобто mysql.

*Лістинг 2.5.1*

<dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-data-jpa</artifactId>  
</dependency>

<dependency>  
 <groupId>mysql</groupId>  
 <artifactId>mysql-connector-java</artifactId>  
 <scope>runtime</scope>  
</dependency>

Після цього необхідно в файлі application.properties підключитися до бази даних, яка була створена в СУБД MySQL, вказавши користувача та пароль, а також локалхост та деякі додаткові властивості.

*Лістинг 2.5.2*

spring.datasource.url=jdbc:mysql://localhost:3306/course?useSSL=false  
spring.datasource.username=root  
spring.datasource.password=  
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQL8Dialect  
spring.jpa.hibernate.ddl-auto=none  
spring.sql.init.schema-locations=classpath:schema.sql  
spring.sql.init.data-locations=classpath:data.sql  
spring.sql.init.enabled=true  
spring.sql.init.continue-on-error=true

В результаті було з’єднано базу даних з проектом за допомогою JPA. Після цього в папці з ресурсами проекту можна створити файл data.sql, з якого безпосередньо можна керувати базою даних за допомогою SQL-запитів. Наприклад, спочатку необхідно заповнити таблицю курсів courses за допомогою запиту INSERT:

* INSERT INTO course(id, price, title, description, bonuses) VALUES (1, 200, 'Базовий', 'Мета нашого курсу, щоб ви почали монетизувати свій профіль вже під час навчання. А також сформували особистий план розвитку. Якщо у вас невеликий профіль або немає продукту, який би ви могли продавати, ми надаємо можливість продавати наші освітні продукти, і заробляти на цьому.', '10 днів з "Як заробляти на контенті” всі уроки будуть доступні лише онлайн);
* INSERT INTO course(id, price, title, description, bonuses) VALUES (2, 200, 'Серйозний', 'Мета нашого курсу, щоб ви почали монетизувати свій профіль вже під час навчання. А також сформували особистий план розвитку. Якщо у вас невеликий профіль або немає продукту, який би ви могли продавати, ми надаємо можливість продавати наші освітні продукти, і заробляти на цьому.', '30 днів з "Як заробляти на контенті всі уроки будуть доступні в особистому кабінеті в записі + для кожного потоку є окремий чат в телеграм для мотивації та спілкування');

[Завдяки концепці](https://newbizzzz.com/#popup:premium) ORM можна створювати інтерфейси репозиторіїв, які наслідують JpaRepository, де реалізовані методи для роботи з базами даних.

*Лістинг 2.5.3*

public interface UserRepository extends JpaRepository<User, Long> {

}

Наприклад, метод save, який додаватиме авторизованого користувача до бази даних.

*Лістинг 2.5.4*

userRepository.save(user);

Також завдяки залежності mysql аплікація може використовувати будь-які SQL –запити, наприклад, SELECT:

* "SELECT \* FROM users where email = ? and password = ?"
* "SELECT \* FROM users where AGE = ?"
* "SELECT \* FROM courses where id = ?"

Нижче наведено приклад використання запиту SELECT в коді.

*Лістинг 2.5.5*

@Query(nativeQuery = true, value = "SELECT \* FROM usr LEFT OUTER JOIN usr\_course ON usr.id = usr\_course.user\_id LEFT OUTER JOIN course ON usr\_course.course\_id = course.id" +  
 " LEFT OUTER JOIN users\_roles ON usr.id = users\_roles.user\_id LEFT OUTER JOIN role ON role.id = users\_roles.role\_id" +  
 " WHERE usr.email=?1")  
 User findByEmail(String email);

**РОЗДІЛ 3. РОЗРОБКА ВЕБ-ДОДАТКУ**

**3.1. Структура веб-сайту**

В якості середовища для написання коду було обрано програмний продукт IntelliJ IDEA — використовується для редагування коду JavaScript, HTML, CSS, Java. За допомогою нього сайт розміщується на localhost.

Програмна реалізація веб-сервісу почалася з верстки шаблону сайту. Верстка виконувалася мовою розмітки гіпертексту HTML використанням фреймворку Bootstrap. Для надання розмітці стилів використовувалися каскадні таблиці стилів CSS.

Дана програма реалізована у вигляді web-сайту, який працює через будь-який доступний браузер.

Структура головної сторінки онлайн-школи (Рис.3.1.1) складається з “шапки” (меню) сайту, де узагальнені всі розділи сайту, основної частини з контентом, а також “підвалу” сайту.

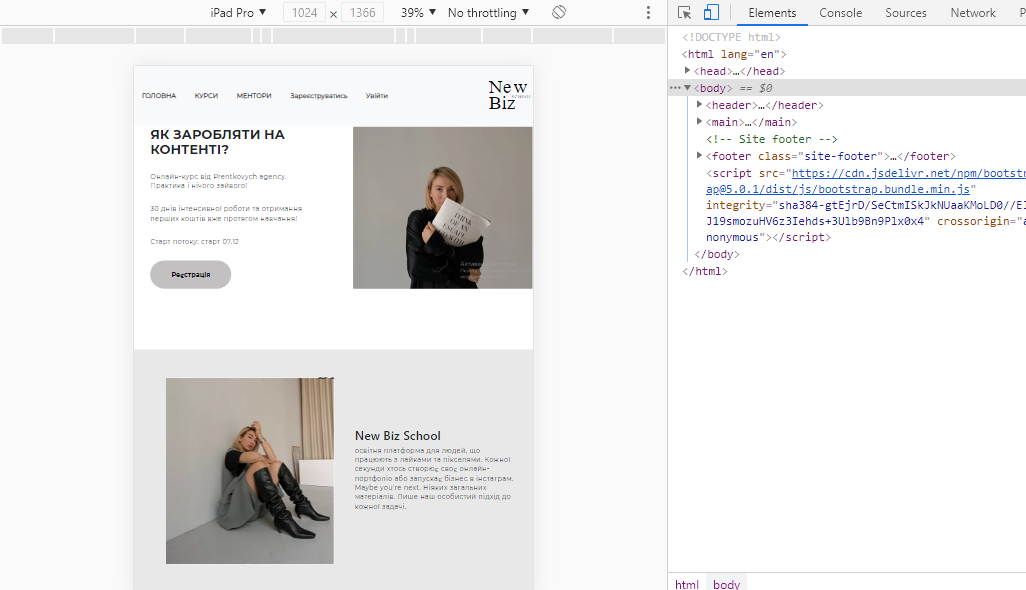


Рис.3.1.1. Структура головної сторінки онлайн-школи

Однією з основних вимог до розробки веб-сайту також є його адаптування під різні пристрої (Рис. 3.1.2), такі як телефони та планшети, адже тільки 20% користувачів переглядають сайти з ПК.



Рис. 3.1.2. Адаптація меню та контенту під планшети

**3.2. Макет веб-сторінки**

Клієнт-частина містить такі розділи:

* Головна сторінка – сторінка з основною інформацією про онлайн-школу;
* Курси – сторінка з основною інформацією про курси (Додаток А);
* Ментори – сторінка з даними про менторів (Додаток Б);
* Зареєструватись – сторінка з формою для реєстрації (Додаток В);
* Увійти – сторінка з формою реєстрації для входу в особистий кабінет (Додаток Г);
* Особистий кабінет користувача (Додаток Д);
* Особистий кабінет адміна (Додаток Е);

**3.3. Програмування клієнтської частини**

Не менш важливим є зовнішній вигляд сайту, для досягнення економічної цілі він повинен бути «user-friendly», тобто інтуїтивно зрозумілий для користувача, привабливий та сучасний, з урахуванням останніх тенденцій веб-дизайну. Розглянемо частину коду головної сторінки (Лістинг 3.3.1)

*Лістинг 3.3.1*

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<link rel="stylesheet" href="style.css">

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/css/bootstrap.min.css" rel="stylesheet"

integrity="sha384-+0n0xVW2eSR5OomGNYDnhzAbDsOXxcvSN1TPprVMTNDbiYZCxYbOOl7+AMvyTG2x" crossorigin="anonymous">

<link rel="preconnect" href="https://fonts.gstatic.com">

<link href="https://fonts.googleapis.com/css2?family=Montserrat&display=swap" rel="stylesheet">

<title>NewBizzz School</title>

</head>

<body>

<header>

<nav class="navbar navbar-expand-lg navbar-light bg-light">

<div class="container-fluid">

<button class="navbar-toggler" type="button" data-bs-toggle="collapse"

data-bs-target="#navbarSupportedContent" aria-controls="navbarSupportedContent"

aria-expanded="false" aria-label="Toggle navigation">

<span class="navbar-toggler-icon"></span>

</button>

<div class="collapse navbar-collapse" id="navbarSupportedContent">

<ul class="navbar-nav me-auto mb-2 mb-lg-0">

<li class="nav-item">

<a class="nav-link" aria-current="page" href="index.html">ГОЛОВНА</a>

</li>

<li class="nav-item">

<a class="nav-link" href="courses.html">КУРСИ</a>

</li>

<li class="nav-item">

<a class="nav-link" href="mentors.html">МЕНТОРИ</a>

</li>

<li class="nav-item">

<a class="nav-link" href="registration.html">Зареєструватись</a>

</li>

<li class="nav-item">

<a class="nav-link" href="login.html">Увійти</a>

</li>

</ul>

<a class="navbar-brand" href="#">

<img src="img/nb\_logo\_1000.svg" alt="" width="120" height="120">

</a>

</div>

</div>

</nav>

</header>

<main>

<section class="howToEarn">

<div class="textHowToEarn">

<h1>Як заробляти на контенті?</h1>

<h2>Онлайн-курс від Prentkovych agency. <br> Практика і нічого зайвого!</h2>

<h2>30 днів інтенсивної роботи та <b>отримання перших коштів вже протягом навчання!</b>

</h2>

<h2>Старт потоку: <b>старт 07.12</b></h2>

<button type="button">Реєстрація</button>

</div>

<img src="img/content.png" class="img-fluid" alt="Не знайдено">

</section>

<section class="newBizSchool">

<img src="img/newbiz.png" class="img-fluid" alt="Не знайдено">

<div>

<h2>New Biz School</h2>

<h3>освітня платформа для людей, що працюють з лайками та пікселями. Кожної секунди хтось створює своє

онлайн-портфоліо або запускає бізнес в інстаграм. Maybe you're next. Ніяких загальних матеріалів.

Лише наш особистий підхід до кожної задачі.</h3>

</div>

</section>

<section class="request">

<div class="boxRequest">

<h2>New Biz – це перші запити про співпрацю</h2>

<h3>Бартер, огляди, сторіз. Вже декілька десятків студентів отримали свої перші запити на співпрацю

завдяки нашому курсу та підтримці. Ми навчимо, як правильно оцінити свою роботу та як продати себе

</h3>

</div>

<div class="boxRequest">

<h2>New Biz – це перші проекти</h2>

<h3>Мрієш фотографувати контент для інших профілів? Ми навчимо структурувати, знаходити свої фішки,

комунікувати з блогерами та створювати кайфовий контент</h3>

</div>

</section>

<section class="program">

<div class="programDesc">

<h1>ПРОГРАМА КУРСУ</h1>

<h2>30 днів з "Як заробляти на контенті"

всі уроки будуть доступні в особистому кабінеті в записі + для кожного потоку є окремий чат в

телеграм для мотивації та спілкування</h2>

</div>

<div class="module">

<div class="nameOfModule">

<h3>модуль #1</h3>

<h4>Віжуал понад усе!</h4>

</div>

<div class="module">

<div class="nameOfModule">

<h3>модуль #2</h3>

<h4>Хто вміє говорити, той зможе написати!</h4

</main>

<!-- Site footer -->

<footer class="site-footer">

<div class="container">

<div class="row">

<div class="col-md-8 col-sm-6 col-xs-12 emaill">

<p>newbizzz@gmail.com </p>

<p>+380735674534</p>

</div>

<div class="col-md-4 col-sm-6 col-xs-12">

<ul class="social-icons">

<li><a class="facebook" href="#"><i class="fa fa-facebook"></i></a></li>

<li><a class="twitter" href="#"><i class="fa fa-twitter"></i></a></li>

<li><a class="dribbble" href="#"><i class="fa fa-dribbble"></i></a></li>

<li><a class="linkedin" href="#"><i class="fa fa-linkedin"></i></a></li>

</ul>

</div>

</div>

</div>

</footer>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.1/dist/js/bootstrap.bundle.min.js"

integrity="sha384-gtEjrD/SeCtmISkJkNUaaKMoLD0//ElJ19smozuHV6z3Iehds+3Ulb9Bn9Plx0x4"

crossorigin="anonymous"></script>

</body>

</html>

**3.4. Програмування користувацької частини**

Все, що відбувається на сервері ( взаємодія з базами даних, реєстрація користувачів, покупка курсів) , реалізовано за допомогою мови програмування JAVA. На лістингу 3.4.1 зображена реєстрація користувача, додавання його в базу даних та надання йому ролі авторизованого користувача.

*Лістинг 3.4.1*

@Controller

public class RegistrationController {

private final UserRepo userRepo;

public RegistrationController(UserRepo userRepo) {

this.userRepo = userRepo;

}

@GetMapping("/registration")

public String registration(){

return "registration";

}

@PostMapping("/registration")

public String addUser(User user, Map<String, Object> model){

User userFromDB = userRepo.findByUsername(user.getUsername());

if(userFromDB != null){

model.put("message", "User exists");

return "registration";

}

user.setRoles(Collections.singleton(Role.USER));

user.setEnabled(true);

userRepo.save(user);

return "redirect:/login";

}

}

**ВИСНОВОК**

У процесі виконання дослідження мета була досягнута, а завдання вирішені, а саме успішно виконано теоретичні і практичне завдання (побудова інформаційної системи (веб сайту) для онлайн освіти). Результати проведеного дослідження дають підставу зробити такі висновки:

1) З вище опрацьованих джерел використаної літератури та з власного аналізу інформаційної системи, безумовно можна стверджувати про актуальність веб сайту для онлайн освіти.

2) Особливість будь якого веб-сайту полягає в його функціоналі. Перш ніж верстати сайт необхідно детально проаналізувати призначення сайту, зробити його макет. Якшо це інтернет-магазин необхідна реєстрація. В даному випадку під час аналізу було вирішено створити систему авторизації, оскільки немає доцільності в формі зворотнього зв’язку.

3) Правильно підібрані технологої для роботи значно полегшують розробку веб-сайту. В даному випадку, можна стверджувати, що обравши технології Java Spring Boot, MySQL, Bootstrap було усунено будь які бар`єри як зі сторони Front-end частини, так і зі сторони Back-end частини.

Отож, враховуючи усе вище сказане, можна стверджувати про важливу роль інформаційних систем та веб сайтів, зокрема побудована інформаційна система позитивно впливає на розвиток бізнесу, а саме клієнт має змогу онлайн ознайомитись із запропонованими пакетами, їх цінами, та викладачами, які ведуть той чи ынший курс. В наш час 9 з 10 підприємств мають власний веб сайт чи профіль в соціальних мережах. Це дозволяє залучити більше клієнтів та зекономити час.

**СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ**

1. Мартин Грабер :SQL; Видавництво «Лори», 2016. -643 с.

2. Моррисон М. : Изучаем JavaScript; Видавництво «O`REILLY», 2012. – 606 c.

3. Веб-дизайн, структура сайту [Електронний ресурс]. – Режим доступу: https://webstudio2u.net/ua/design-web/403-structure-models.html .

4. Bootstrap [Електронний ресурс]. – Режим доступу: <https://timeweb.com/ru/community/articles/plyusy-i-minusy-bootstrap-1>.

5. Моделі даних [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D0%B5%D0%BB%

D1%96\_%D0%B1%D0%B0%D0%B7\_%D0%B4%D0%B0%D0%BD%D0%B8

%D1%85.

6. Spring документація [Електронний ресурс]. – Режим доступу: <https://spring.io/guides/gs/serving-web-content/>

7. MySQL [Електронний ресурс]. – режим доступу: [https://www.mysql.com/](https://www.postgresql.org/).

ДОДАТКИ

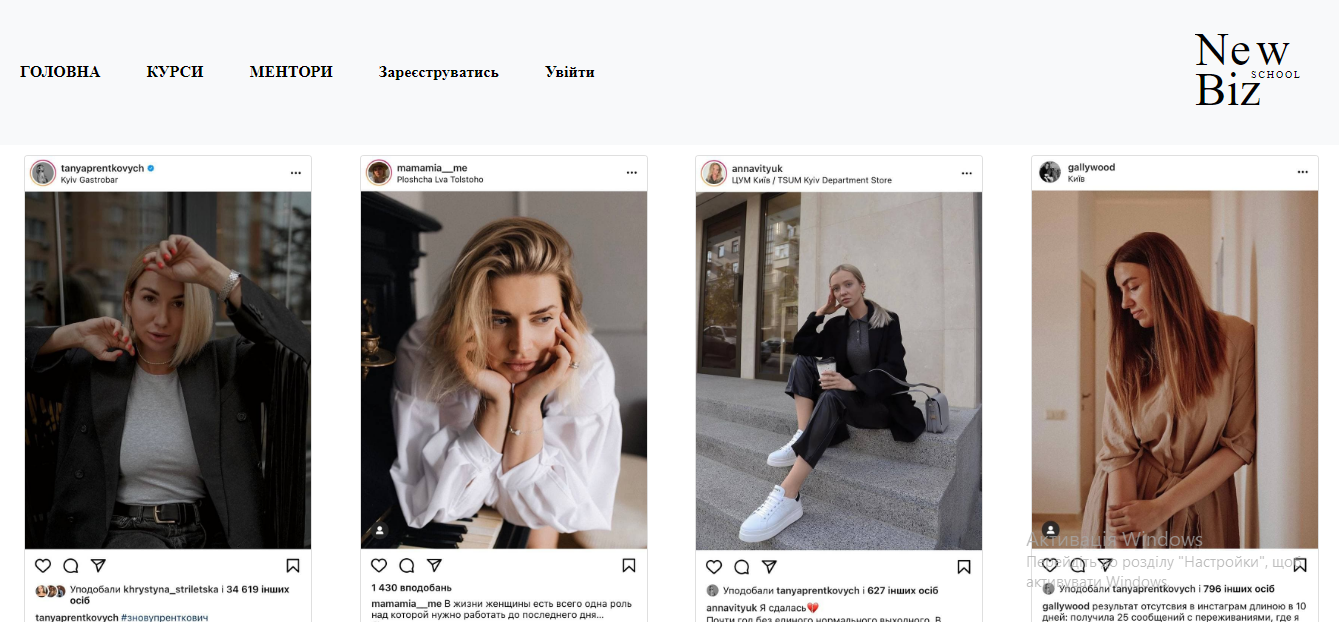
Додаток А

Курси



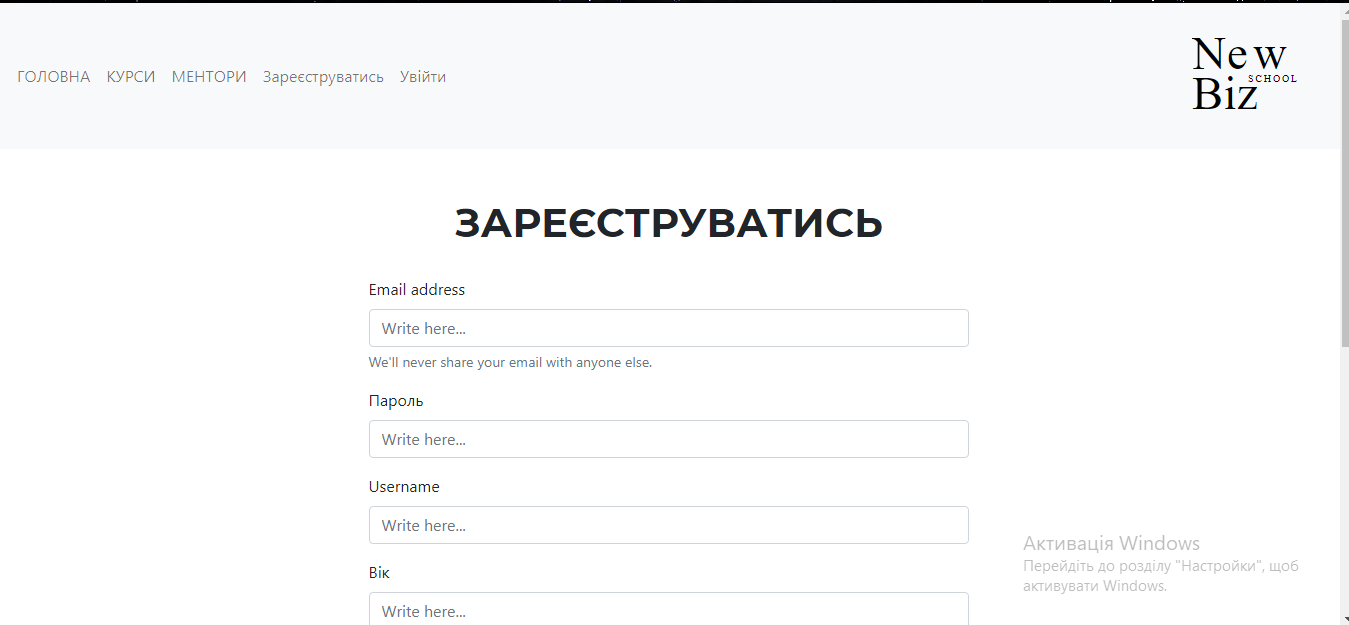
Додаток Б

Ментори



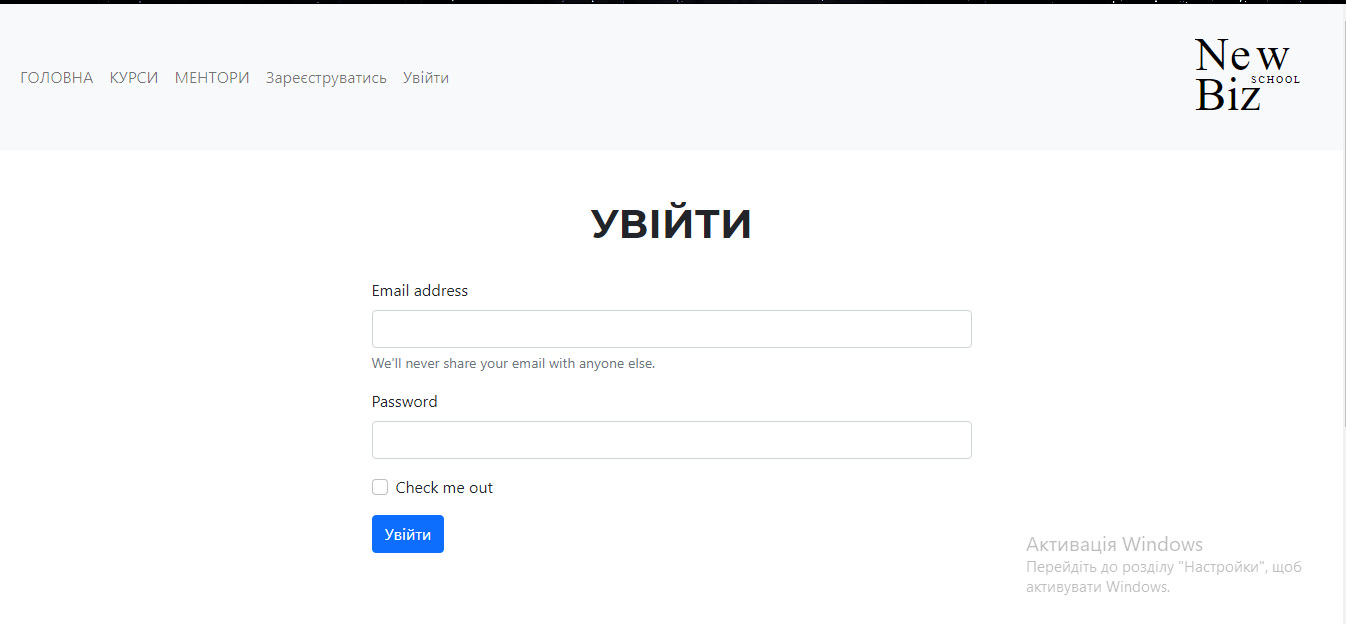
Додаток В

Сторінка реєстрації



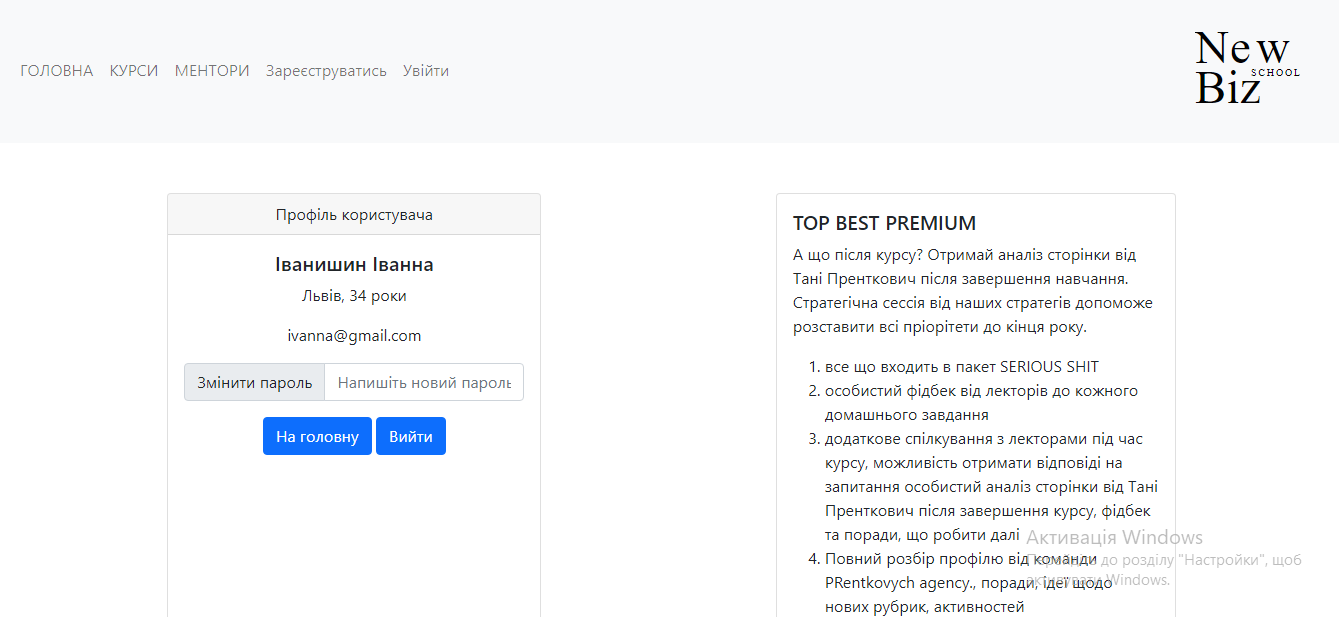
Додаток Г

Сторінка входу



Додаток Д

Особистий кабінет користувача



Додаток Е

Особистий кабінет адміністратора

