Міністерство освіти і науки України
Рівненський державний гуманітарний університет
Кафедра інформаційних технологій та моделювання

Кваліфікаційна робота

за освітнім ступенем «бакалавр»

на тему:

ЕЛЕКТРОННИЙ ЖУРНАЛ ДЛЯ НАВЧАЛЬНИХ КУРСІВ СПЕЦІАЛЬНОСТІ “КОМП’ЮТЕРНІ НАУКИ”

Виконав:
здобувач IV курсу
групи КН-41
спеціальності 122 «Комп’ютерні науки»
Токар Роман Іванович
Науковий керівник:
к. т. н., доц. Сінчук А.М.

Рівне – 2024

Зміст

ВСТУП

Сучасні інформаційні технології проникають у всі сфери людської діяльності, включаючи освіту. Важливою складовою освітнього процесу є ведення навчальної документації, яка потребує систематизації та ефективного управління. Одним з інструментів, що дозволяє вирішити ці завдання, є електронний журнал для навчальних курсів. Особливо актуальним є створення таких журналів для спеціальності "Комп'ютерні науки", де високі вимоги до точності, надійності та зручності користування програмними засобами.

Метою даної роботи є дослідження технологій розробки електронного журналу для оптимізації процесу ведення навчальної документації у спеціальності "Комп'ютерні науки" за допомогою сучасних інформаційних технологій. Аналізуючи сучасний цифровий освітній простір, стає зрозумілим, як інноваційна стратегія може позитивно вплинути на різні аспекти функціонування навчального закладу: оптимізацію процесів обліку успішності студентів, раціональне використання ресурсів, тощо. Найбільш ефективною методологією проектування такого електронного журналу є об'єктно-орієнтований підхід для керування відповідною базою даних. Такий підхід забезпечить швидкий доступ до найбільш важливої інформації та дозволить зберігати великі обсяги даних для ретроспективного аналізу.

Об'єктом дослідження є технологічний процес розробки електронного журналу для навчальних курсів спеціальності "Комп'ютерні науки".

Предметом дослідження є використання інноваційних технологій для розробки електронного журналу у навчальній сфері.

Завдання дослідження:

- Використання веб-технологій для створення зручного та функціонального інтерфейсу.
- Інтеграція з існуючими освітніми системами та базами даних.
- Забезпечення безпеки та конфіденційності даних студентів.
- Оптимізація системи для швидкого доступу та обробки даних.
- Проведення тестування та оцінка ефективності розробленого електронного журналу.

У процесі виконання дослідження зібрано та аналізовано великий обсяг даних. На основі цього була розроблена програма для ведення електронного журналу, яка забезпечує зручний облік успішності студентів та автоматизацію багатьох рутинних процесів.

Викладені в дипломній роботі матеріали є основою для розробки та проектування відповідних електронних систем. Результати роботи можуть використовуватись у навчальному процесі.

Апробація результатів дослідження

Звітна науково-практична конференція професорсько-викладацького складу Рівненського державного гуманітарного університету, яка відбулася 16-17 травня 2024 року, Рівне.

Структура роботи

Дипломна робота складається зі вступу, трьох розділів, висновку та списку використаних джерел. Повний обсяг роботи складає 46 сторінок машинного тексту, включаючи 15 рисунків.

РОЗДІЛ 1

АНАЛІЗ ЗАСОБІВ РОЗРОБКИ СУЧАСНИХ ДОДАТКІВ

1.1 Основні технології та інструменти

Мови програмування, Історія та розвиток

Мови програмування є основою розробки програмного забезпечення. З моменту появи перших комп'ютерів, потреба у мовах програмування зросла для створення інструкцій, які могли б виконувати машини. Перші мови були низькорівневими, наприклад, асемблер, але з часом розробники перейшли до високорівневих мов, що дозволяють писати код, більш зрозумілий людям. У цьому розділі розглянемо чотири популярні мови програмування: JavaScript, Python, Java і C#.

Аналіз існуючих систем

Інші системи, такі як Edmodo, Schoology та Canvas, також пропонують різний набір функцій та можливостей. Edmodo орієнтована на соціальну взаємодію між викладачами та студентами, але її функціонал може бути обмеженим для більш складних навчальних процесів. Schoology пропонує потужні інструменти для управління курсами та оцінками, але може бути складною у налаштуванні.

Canvas є однією з найбільш гнучких систем, яка дозволяє налаштовувати різні аспекти навчального процесу. Вона пропонує розширені інструменти для інтеграції з іншими системами та сервісами, але її вартість та складність налаштування можуть бути недоліками.

На основі аналізу існуючих систем можна зробити висновок, що нова система повинна поєднувати в собі простоту використання, широкий функціонал та гнучкість налаштувань. Важливо також враховувати вартість впровадження та підтримки, щоб забезпечити доступність системи для різних навчальних закладів.

JavaScript Історія та розвиток мови

JavaScript був створений Бренданом Ейхом у 1995 році під час роботи в компанії Netscape. Початкова назва мови була Mocha, потім LiveScript, і лише пізніше вона була перейменована на JavaScript для маркетингових цілей. JavaScript був створений для того, щоб зробити веб-сторінки інтерактивними.

Основні можливості та області застосування

JavaScript є мовою програмування з відкритим вихідним кодом, яка дозволяє створювати динамічні веб-сторінки. Основні можливості включають:

Маніпуляції з DOM: дозволяють змінювати структуру HTML-документу.
Обробка подій: надає можливість реагувати на дії користувачів.
Асинхронні запити: дозволяють робити запити до сервера без перезавантаження сторінки (AJAX).

Серверна розробка: з використанням Node.js можна створювати серверні додатки.

JavaScript використовується для створення інтерактивних елементів на веб-сторінках, односторінкових додатків (SPA), серверних додатків та мобільних додатків за допомогою фреймворку React Native.

Переваги та недоліки

Переваги JavaScript:

Широка підтримка у браузерах: всі сучасні браузери підтримують JavaScript.

Динамічність та інтерактивність: можливість створення динамічних і інтерактивних веб-сторінок.

Велика спільнота та ресурси: численні бібліотеки, фреймворки та навчальні матеріали.

Можливість повного стеку: завдяки Node.js, можна створювати як клієнтські, так і серверні додатки.

Недоліки JavaScript:

Відсутність строгого типізування: що може призводити до помилок у великих проектах.

Проблеми з безпекою: численні вразливості через неправильне використання.

Проблеми з продуктивністю: через інтерпретовану природу мови.

Python Історія та розвиток мови

Python був створений Гвідо ван Россумом у 1991 році. Мова була розроблена для спрощення програмування та забезпечення читабельності коду. Python швидко став популярним завдяки своїй простоті та універсальності.

Основні можливості та області застосування

Python відомий своєю універсальністю та простотою у використанні. Основні можливості включають:

Висока читабельність коду: зрозумілий синтаксис, який легко навчитися.

Широкий спектр стандартних бібліотек: для роботи з файлами, мережею, базами даних та інше.

Підтримка різних парадигм програмування: об'єктно-орієнтоване, процедурне, функціональне програмування.

Інтерактивний інтерпретатор: полегшує тестування та налагодження коду.

Python активно використовується в наукових [1] дослідженнях, машинному навчанні, веб-розробці (з фреймворками Django, Flask), автоматизації, розробці ігор та інших областях.

Переваги та недоліки

Переваги Python:

Простота синтаксису: висока читабельність та простота у навчанні.

Універсальність: підходить для багатьох видів завдань.
Активна спільнота: численні ресурси для навчання та підтримка.
Широка підтримка бібліотек: для різних галузей.
Недоліки Python:

Повільніша швидкість виконання: через інтерпретовану природу.
Високе споживання пам'яті: у порівнянні з іншими мовами.
Проблеми з багатозадачністю: через глобальне блокування інтерпретатора (GIL).

Java Історія та розвиток мови

Java була створена в 1995 році компанією Sun Microsystems (нині належить Oracle) під керівництвом Джеймса Гослінга. Мова була розроблена для створення портативного, високопродуктивного і безпечного програмного забезпечення. Однією з головних особливостей Java є її незалежність від платформи завдяки використанню віртуальної машини (JVM).

Основні можливості та області застосування

Java є мовою з сильною типізацією і підтримкою об'єктно-орієнтованого програмування. Основні можливості включають:

Багатопотокові програми: можливість створення багатопотокових додатків.
Високий рівень безпеки: управління пам'яттю та безпека на рівні мови.
Підтримка розподілених систем: можливість створення мережевих додатків.
Портативність: можливість запуску на різних платформах завдяки JVM.
Java активно використовується для розробки корпоративних додатків, мобільних додатків (особливо для Android), веб-додатків (з використанням Spring, Hibernate), ігрових додатків та багато іншого.

Переваги та недоліки

Переваги Java:

Портативність: незалежність від платформи завдяки JVM.
Висока продуктивність: оптимізований код та управління пам'яттю.
Безпека: вбудовані механізми безпеки.
Велика кількість інструментів та бібліотек: підтримка різних технологій та фреймворків.
Недоліки Java:

Складний синтаксис: у порівнянні з деякими іншими мовами.
Високе споживання пам'яті: порівняно з іншими мовами.
Необхідність написання значної кількості коду: для вирішення простих завдань.

C# Історія та розвиток мови

C# був розроблений компанією Microsoft у 2000 році як частина платформи .NET. Мова була створена Андерсом Гейлсбергом і його командою з метою забезпечити по [2] тужний інструмент для розробки різних типів додатків, включаючи веб-додатки, мобільні додатки та десктопні програми.

Основні можливості та області застосування

C# є мовою з сильною типізацією і підтримкою об'єктно-орієнтованого програмування. Основні можливості включають:

Асинхронне програмування: підтримка асинхронних операцій.
Інтеграція з .NET: доступ до великої кількості бібліотек та інструментів.
Підтримка різних парадигм програмування: об'єктно-орієнтоване, процедурне, функціональне.
Кросплатформеність: можливість створення додатків для Windows, Linux та macOS за допомогою .NET Core.

C# активно використовується для розробки десктопних додатків (з використанням Windows Forms, WPF), веб-додатків (з використанням ASP.NET), ігор (з використанням Unity) та мобільних додатків (з використанням Xamarin).

Переваги та недоліки

Переваги C#:

Продуктивність та безпека: високий рівень безпеки та продуктивності.
Глибока інтеграція з .NET: доступ до потужних інструментів та бібліотек.
Кросплатформеність: можливість розробки для різних платформ.
Активна спільнота: велика кількість ресурсів для навчання та підтримки.
Недоліки C#:

Залежність від Windows: деякі інструменти та бібліотеки краще працюють на Windows.
Високе споживання ресурсів: у порівнянні з іншими мовами.
Складний синтаксис: для новачків.
Фреймворки та бібліотеки
Фреймворки та бібліотеки дозволяють розробникам швидше створювати додатки, надаючи готові компоненти та рішення для типових задач. Розглянемо чотири популярні фреймворки: React, Angular, Django та Spring.

React Історія та розвиток
React був розроблений компанією Facebook у 2013 році для спрощення створення динамічних та інтерактивних користувацьких інтерфейсів. React швидко здобув популярність завдяки своїй простоті та ефективності.

Основні можливості та області застосування

React є бібліотекою JavaScript для створення користувацьких інтерфейсів. Основні можливості включають:

Компонентний підхід: дозволяє розбивати інтерфейс на незалежні, повторно використовувані компоненти.

Віртуальний DOM: забезпечує ефективне оновлення та рендеринг інтерфейсу.

Односторінкові додатки (SPA): підтримка створення динамічних веб-додатків.

React використовується для створення динамічних веб-сторінок, інтерактивних користувацьких інтерфейсів та мобільних додатків (з використанням React Native).

Переваги та недоліки

Переваги React:

Простота у використанні: компонентний підхід спрощує розробку.

Висока продуктивність: завдяки віртуальному DOM.

Велика кількість бібліотек та інструментів: що розширюють можливості React.

Активна спільнота: підтримка від Facebook та розробників.

Недоліки React:

Необхідність додаткових інструментів: для реалізації повноцінного додатку (наприклад, Redux для управління станом).

Високий поріг входження: для новачків через складну екосистему.

Часті зміни та оновлення: що вимагають постійного навчання та адаптації.

Angular Історія та розвиток

Angular був розроблений компанією Google у 2010 році як фреймворк для створення динамічних веб-додатків. Спочатку відомий як AngularJS, фреймворк був значно оновлений у 2016 році і перейменований на Angular, що включає нову архітектуру та поліпшені можливості.

Основні можливості та області застосування

Angular є повноцінним фреймворком для створення односторінкових додатків (SPA) [3]. Основні можливості включають:

Декларативне зв'язування даних та шаблонів: спрощує створення динамічних інтерфейсів.

Компонентна архітектура: дозволяє створювати модульні та повторно використовувані компоненти.

Вбудовані інструменти для тестування та налагодження: полегшують розробку та підтримку додатків.

Angular використовується для створення складних веб-додатків, корпоративних рішень та мобільних додатків (з використанням Ionic).

Переваги та недоліки

Переваги Angular:

Повна інтеграція: з інструментами та бібліотеками для створення додатків.

Висока продуктивність та масштабованість: завдяки потужній архітектурі.

Підтримка TypeScript: забезпечує статичну типізацію та покращує якість коду.

Активна підтримка: від Google та великої спільноти розробників.

Недоліки Angular:

Високий поріг входження: через складний синтаксис та архітектуру.

Велика кількість інструментів та бібліотек. можуть ускладнювати процес розробки.

Часті зміни та оновлення: що вимагають постійного навчання та адаптації.

Django Історія та розвиток

Django був розроблений у 2003 році як фреймворк для веб-розробки на мові Python. Він був створений для спрощення розробки складних веб-додатків та зниження часу, необхідного для створення прототипів.

Основні можливості та області застосування

Django є повноцінним фреймворком для створення веб-додатків. Основні можливості включають:

ORM (Object-Relational Mapping): для роботи з базами даних.

Вбудована система аутентифікації та авторизації: для безпечного доступу.

Інструменти для автоматизації адміністрування: дозволяють швидко створювати панелі адміністратора.

Django використовується для створення веб-сайтів, корпоративних додатків, API та інших веб-рішень.

Переваги та недоліки

Переваги Django:

Швидкий старт: завдяки вбудованим інструментам та бібліотекам.

Висока продуктивність та масштабованість: завдяки оптимізованому коду.

Вбудована система безпеки: допомагає уникнути поширених вразливостей.

Активна спільнота: велика кількість ресурсів для навчання та підтримки.

Недоліки Django:

Великий розмір фреймворку: може призводити до перевантаження для простих додатків.

Висока складність налаштування: для новачків.

Деякі обмеження у гнучкості: через вбудовані рішення.

Spring Історія та розвиток

Spring був створений у 2003 році як фреймворк для розробки корпоративних додатків на мові Java. Його головна мета – спрощення створення [4] складних та масштабованих додатків через використання інверсії управління та аспектно-орієнтованого програмування.

Основні можливості та області застосування

Spring є потужним фреймворком для створення веб-додатків та корпоративних рішень. Основні можливості включають:

Контейнер інверсії управління (IoC): для управління об'єктами та їх життєвим циклом.

Інтеграція з різними технологіями для доступу до даних: JPA, Hibernate.

Підтримка аспектно-орієнтованого програмування (AOP): для розширення функціональності додатків.

Spring використовується для створення масштабованих веб-додатків, мікросервісів, корпоративних систем та інших складних рішень.

Переваги та недоліки

Переваги Spring:

Висока продуктивність та масштабованість: завдяки потужним інструментам та оптимізаціям.

Гнучкість: завдяки модульній архітектурі та підтримці різних технологій.

Підтримка широкого спектра технологій: та інтеграція з іншими інструментами.

Активна спільнота: велика кількість ресурсів для навчання та підтримки.

Недоліки Spring:

Висока складність налаштування та конфігурації: потребує значних знань та досвіду.

Потреба у великій кількості коду: для налаштування простих завдань.

Відносно високий поріг входження: для новачків.

Інтегровані середовища розробки (IDE)

Інтегровані середовища розробки (IDE) є важливими інструментами для програмістів, оскільки вони надають зручні інструменти для написання, налагодження [5] та тестування коду. Розглянемо три популярні IDE: Visual Studio, IntelliJ IDEA та PyCharm.

Visual Studio Історія та розвиток

Visual Studio був розроблений компанією Microsoft у 1997 році як інтегроване середовище розробки для створення програмного забезпечення на різних мовах програмування. З тих пір Visual Studio став одним з найпопулярніших інструментів для розробників завдяки своїй потужності та гнучкості.

Основні можливості та області застосування

Visual Studio підтримує широкий спектр мов програмування (C#, VB.NET, C++, Python, JavaScript та інші). Основні можливості включають:

Редагування коду: з підтримкою підсвічування синтаксису та автодоповнення.

Інструменти для налагодження та тестування: дозволяють виявляти та виправляти помилки.

Інтеграція з системами контролю версій: Git, SVN.

Інтеграція з хмарними сервісами та платформами: Azure.

Visual Studio використовується для розробки веб-додатків, десктопних додатків, мобільних додатків та інших типів програмного забезпечення.

Переваги та недоліки

Переваги Visual Studio:

Потужні інструменти для редагування та налагодження коду: забезпечують високу продуктивність.

Підтримка широкого спектра мов програмування та технологій: дозволяє працювати з різними проектами.

Інтеграція з хмарними сервісами та платформами: спрощує розробку та розгортання додатків.

Велика кількість плагінів та розширень: розширюють функціональність IDE.

Недоліки Visual Studio:

Високе споживання ресурсів системи: може спо вільнювати роботу на слабких комп'ютерах.

Відносно висока складність для новачків: потребує часу для освоєння.

Платні версії з додатковими функціями: можуть бути недоступні у безкоштовній версії.

IntelliJ IDEA Історія та розвиток

IntelliJ IDEA був розроблений компанією JetBrains у 2001 році як інтегроване середовище розробки для мов програмування на основі Java. З часом IntelliJ IDEA став одним з найпопулярніших IDE завдяки своїй інтуїтивності та потужності.

Основні можливості та області застосування

IntelliJ IDEA підтримує широкий спектр мов програмування (Java, Kotlin, Groovy, Scala, Python, JavaScript та інші). Основні можливості включають:

Розширене автодоповнення та аналіз коду: підвищує продуктивність розробки.

Інструменти для налагодження та тестування: допомагають швидко знаходити та виправляти помилки.

Інтеграція з системами контролю версій: Git, SVN.

Підтримка різних фреймворків та платформ: Spring, Android, Java EE.

IntelliJ IDEA використовується для розробки веб-додатків, мобільних додатків, корпоративних систем та інших типів програмного забезпечення.

Переваги та недоліки

Переваги IntelliJ IDEA:

Інтуїтивний інтерфейс та розширені можливості автодоповнення: полегшують роботу розробників.

Підтримка широкого спектра мов програмування та фреймворків: дозволяє працювати з різними проектами.

Висока продуктивність та масштабованість: забезпечують ефективну роботу навіть з великими проектами.

Велика кількість плагінів та розширень: розширюють функціональність IDE.

Недоліки IntelliJ IDEA:

Високе споживання ресурсів системи: може сповільнювати роботу на слабких комп'ютерах.

Відносно висока вартість ліцензії для професійної версії: може бути недоступна для індивідуальних розробників.

Складність для новачків: через велику кількість налаштувань та функцій.

PyCharm Історія та розвиток

PyCharm був розроблений компанією JetBrains у 2010 році як інтегроване середовище розробки для мови програмування Python. PyCharm швидко здобув популярність завдяки своїм спеціалізованим можливостям для Python-розробників.

Основні можливості та області застосування

PyCharm підтримує тільки мову Python, але надає потужні інструменти для розробки на цій мові. Основні можливості включають:

Розширене автодоповнення та аналіз коду: підвищує продуктивність розробки.

Інструменти для налагодження та тестування: допомагають швидко знаходити та виправляти помилки.

Інтеграція з системами контролю версій: Git, SVN.

Підтримка популярних фреймворків: Django, Flask.

PyCharm використовується для розробки веб-додатків, наукових досліджень, машинного навчання та інших типів програмного забезпечення на Python.

Переваги та недоліки

Переваги PyCharm:

Потужні інструменти для розробки на Python: спеціалізовані функції та інтеграції.

Інтеграція з популярними фреймворками та бібліотеками: підвищує продуктивність.

Висока продуктивність та масштабованість: забезпечують ефективну роботу навіть з великими проектами.

Велика кількість плагінів та розширень: розширюють функціональність IDE.

Недоліки PyCharm:

Високе споживання ресурсів системи: може сповільнювати роботу на слабких комп'ютерах.

Відносно висока вартість ліцензії для професійної версії: може бути недоступна для індивідуальних розробників.

Обмеження у підтримці інших мов програмування: спеціалізація лише на Python [6].

1.3 Мобільні платформи

Розробка для Android

Операційна система Android була створена компанією Android Inc., яку придбала Google у 2005 році. Перший пристрій на Android з'явився у 2008 році. Сьогодні Android є найпопулярнішою мобільною операційною системою у світі, використовується на мільярдах пристроїв різних виробників.

Kotlin

Історія та розвиток мови

Kotlin був розроблений компанією JetBrains і офіційно випущений у 2011 році. У 2017 році Google оголосила Kotlin офіційною мовою для розробки Android-додатків нарівні з Java. Kotlin є сучасною мовою програмування, яка надає більш компактний та безпечний синтаксис у порівнянні з Java.

Основні можливості та області застосування

Kotlin є статично типізованою мовою програмування, що працює на Java Virtual Machine (JVM). Основні можливості включають:

Компактний синтаксис: менш багатослівний у порівнянні з Java.

Безпечність: запобігання NullPointerException за допомогою вбудованих механізмів.

Інтероперабельність з Java: повна сумісність з існуючим Java-кодом та бібліотеками.

Підтримка сучасних парадигм програмування: функціональне та об'єктно-орієнтоване програмування.

Kotlin активно використовується для розробки мобільних додатків для Android, серверних додатків, веб-додатків, а також для роботи з іншими платформами завдяки Kotlin/Native та Kotlin/JS.

Переваги та недоліки

Переваги Kotlin:

Компактний та читабельний код: зменшує обсяг коду та спрощує його підтримку.

Безпека: запобігання поширеним помилкам на рівні компіляції.

Інтероперабельність з Java: дозволяє використовувати існуючий Java-код без змін.

Підтримка від JetBrains та Google: гарантує активний розвиток та підтримку мови.

Недоліки Kotlin:

Молодість мови: менша кількість ресурсів та прикладів у порівнянні з Java.

Час компіляції: у деяких випадках компіляція може займати більше часу, ніж для Java.

Вимоги до знань Java: для повного використання потенціалу потрібні знання Java.

Java

Історія та розвиток мови

Java, розроблена компанією Sun Microsystems у 1995 році, є однією з найпопулярніших мов програмування у світі. Завдяки своїй незалежності від платформи та потужній віртуальній машині (JVM), Java швидко стала основною мовою для розробки корпоративних додатків, веб-додатків та мобільних додатків.

Основні можливості та області застосування

Java є мовою з сильною типізацією і підтримкою об'єктно-орієнтованого програмування. Основні можливості включають:

Портативність: можливість запуску коду на будь-якій платформі з JVM.

Багатопоточність: підтримка багатопоточних додатків.

Безпека: вбудовані механізми безпеки та управління пам'яттю.

Велика кількість бібліотек та фреймворків: для вирішення різних завдань.

Java активно використовується для розробки мобільних додатків для Android, корпоративних додатків, веб-додатків та серверних додатків.

Переваги та недоліки

Переваги Java:

Портативність: код можна запускати на різних платформах без змін.

Стабільність та продуктивність: забезпечують надійну роботу великих систем.

Велика спільнота та кількість ресурсів: допомагають швидко вирішувати проблеми та знаходити інформацію.

Інтеграція з інструментами та бібліотеками: забезпечує швидку розробку додатків.

Недоліки Java:

Багатослівність коду: для реалізації простих задач може знадобитися багато коду.

Швидкість виконання: в деяких випадках нижча, ніж у мов, що компілюються до нативного коду.

Проблеми з управління пам'яттю: незважаючи на наявність збирача сміття, може виникати проблема витоків пам'яті.

Розробка для iOS Історія та розвиток

Операційна система iOS була розроблена компанією Apple і вперше представлена у 2007 році разом з першим iPhone. З того часу iOS стала однією з найпопулярніших мобільних платформ у світі, відома своєю стабільністю, безпекою та зручністю використання.

Swift Історія та розвиток мови

Swift був представлений компанією Apple у 2014 році як сучасна мова програмування для розробки додатків для iOS, macOS, watchOS та tvOS. Swift був створений як більш безпечна та продуктивна альтернатива Objective-C.

Основні можливості та області застосування

Swift є мовою програмування з сильною типізацією і підтримкою об'єктно-орієнтованого програмування. Основні можливості включають:

Сучасний синтаксис: простий та зрозумілий.
Безпека: запобігання поширеним помилкам, таким як null-значення.
Висока продуктивність: швидкість виконання коду близька до нативної.
Інтерактивні середовища: підтримка Playground для експериментів з кодом у реальному часі.
Swift активно використовується для розробки мобільних додатків для iOS, додатків для macOS, watchOS та tvOS.

Переваги та недоліки

Переваги Swift:

Сучасний та зрозумілий синтаксис: спрощує розробку та підтримку коду.
Висока продуктивність: швидкість виконання коду наближена до нативної.
Безпека: запобігання поширеним помилкам на рівні мови.
Інтерактивні середовища: дозволяють експериментувати з кодом у реальному часі.
Недоліки Swift:

Молодість мови: менша кількість ресурсів та прикладів у порівнянні з Objective-C.
Зміни в синтаксисі: часті оновлення [7] мови можуть вимагати адаптації існуючого коду.
Залежність від екосистеми Apple: використання мови обмежене платформами Apple.

Кросплатформенні рішення

Кросплатформенні фреймворки дозволяють розробляти додатки, які можуть працювати на різних мобільних платформах, таких як Android та iOS, з використанням єдиного базового коду. Розглянемо два популярні кросплатформенні рішення: Flutter та React Native.

Flutter Історія та розвиток

Flutter був розроблений компанією Google і вперше представлений у 2017 році. Він є відкритим фреймворком для розробки кросплатформенних мобільних додатків з використанням мови програмування Dart.

Основні можливості та області застосування

Flutter дозволяє створювати високопродуктивні кросплатформенні додатки з єдиним базовим кодом. Основні можливості включають:

Висока продуктивність: майже нативна швидкість роботи додатків.

Єдиний базовий код: для Android та iOS.

Hot Reload: можливість миттєвого оновлення коду під час розробки.

Розширені можливості UI: велика кількість готових віджетів для створення користувацьких інтерфейсів.

Flutter використовується для розробки мобільних додатків для Android та iOS, а також для веб-додатків та десктопних д одатків.

Переваги та недоліки

Переваги Flutter:

Висока продуктивність: майже нативна швидкість роботи додатків.

Єдиний базовий код: спрощує розробку та підтримку додатків для різних платформ.

Hot Reload: підвищує ефективність розробки.

Розширені можливості UI: дозволяють створювати привабливі та інтуїтивно зрозумілі інтерфейси.

Недоліки Flutter:

Великий розмір додатків: у порівнянні з нативними додатками.

Молодість фреймворку: менша кількість бібліотек та плагінів у порівнянні з нативними платформами.

Вимоги до знань Dart: для повного використання потенціалу потрібні знання мови Dart.

React Native Історія та розвиток

React Native був розроблений компанією Facebook і вперше представлений у 2015 році. Фреймворк дозволяє створювати кросплатформенні мобільні додатки з використанням мови програмування JavaScript та бібліотеки React.

Основні можливості та області застосування

React Native дозволяє створювати високопродуктивні кросплатформенні додатки з єдиним базовим кодом. Основні можливості включають:

Єдиний базовий код: для Android та iOS.

Використання компонентів React: дозволяє створювати повторно використовувані компоненти.

Hot Reload: можливість миттєвого оновлення коду під час розробки.

Доступ до нативних модулів: можливість використовувати нативні функції платформи.

React Native використовується для розробки мобільних додатків для Android та iOS.

Переваги та недоліки

Переваги React Native:

Єдиний базовий код: спрощує розробку та підтримку додатків для різних платформ.

Hot Reload: підвищує ефективність розробки.

Велика спільнота та кількість ресурсів: численні бібліотеки та плагіни.

Використання компонентів React: дозволяє створювати інтуїтивні та повторно використовувані компоненти.

Недоліки React Native:

Проблеми з продуктивністю: у порівнянні з нативними додатками.

Обмеження у використанні нативних функцій: іноді потрібні додаткові модулі.

Залежність від екосистеми JavaScript: вимагає знання мови JavaScript та екосистеми React.

1.4 Веб-розробка

Frontend технології **Історія та розвиток**

Frontend розробка охоплює всі аспекти створення інтерфейсу користувача для веб-додатків. Це включає в себе верстку сторінок, створення інтерактивних елементів, а також забезпечення зручного та привабливого користувацького досвіду. Основні технології для фронтенд розробки включають HTML, CSS та JavaScript.

HTML Історія та розвиток

HTML (HyperText Markup Language) був створений Тімом Бернерсом-Лі у 1991 році як мова розмітки для створення веб-сторінок. З того часу HTML пройшов через численні оновлення і зараз існує у версії HTML5, яка була офіційно випущена у 2014 році.

Основні можливості та області застосування

HTML є основною мовою розмітки для створення веб-сторінок. Основні можливості включають:

Структуризація контенту: визначення заголовків, абзаців, списків, зображень та інших елементів.

Зв'язування сторінок: створення гіперпосилань для переходу між сторінками.

Вбудовані мультимедійні елементи: підтримка відео, аудіо та графіки.

Форми: створення форм для введення даних користувачами.

HTML використовується для створення структурованого контенту на веб-сторінках і є основою для всіх веб-додатків.

Переваги та недоліки

Переваги HTML:

Простота та зручність: легкий для вивчення та використання.

Універсальність: підтримується всіма веб-браузерами.

Стандартизація: існують чітко визначені стандарти W3C.

Недоліки HTML:

Обмеженість функціональності: без допомоги CSS та JavaScript HTML не може створювати динамічний контент.

Відсутність інтерактивності: для додавання інтерактивності потрібні додаткові технології.

CSS Історія та розвиток

CSS (Cascading Style Sheets) був створений Хоконом Віум Лі у 1996 році для поліпшення презентації веб-сторінок. CSS дозволяє відокремити структуру HTML від її представлення, надаючи стилі для різних елементів.

Основні можливості та області застосування

CSS використовується для стилізації HTML-документів. Основні можливості включають:

Застосування стилів до елементів: визначення кольорів, шрифтів, розмірів та інших властивостей.

Розташування елементів: управління розташуванням елементів на сторінці з використанням Flexbox, Grid та інших технологій.

Анімації та трансформації: створення анімацій та трансформацій елементів для покращення користувацького досвіду.

Медіа-запити: адаптація стилів для різних пристроїв та розмірів екранів.

CSS використовується для створення візуально привабливих та зручних у використанні веб-сторінок.

Переваги та недоліки

Переваги CSS:

Відокремлення стилів від структури: спрощує підтримку та оновлення коду.

Гнучкість та масштабованість: дозволяє створювати адаптивні дизайни.

Стандартизація: існують чітко визначені стандарти W3C.

Недоліки CSS:

Складність у великих проектах: управління стилями може стати складним у великих проектах.

Проблеми з сумісністю браузерів: деякі стилі можуть відображатися по-різному в різних браузерах.

JavaScript Історія та розвиток

JavaScript був створений Бренданом Ейхом у 1995 році під час роботи в компанії Netscape. JavaScript дозволяє створювати динамічні та інтерактивні веб-сторінки, додаючи функціональність до HTML і CSS.

Основні можливості та області застосування

JavaScript є мовою програмування, яка використовується для додавання інтерактивності до веб-сторінок. Основні можливості включають:

Маніпуляції з DOM: дозволяє змінювати структуру HTML-документу у реальному часі.

Обробка подій: надає можливість реагувати на дії користувачів.

Асинхронні запити: дозволяє робити запити до сервера без перезавантаження сторінки (AJAX).

Розширення можливостей браузера: дозволяє створювати розширення та додатки для браузерів.

JavaScript використовується для створення інтерактивних елементів на веб-сторінках, односторінкових додатків (SPA), серверних додатків за допомогою Node.js та мобільних додатків з використанням фреймворку React Native.

Переваги та недоліки

Переваги JavaScript:

Широка підтримка у браузерах: всі сучасні браузери підтримують JavaScript.

Динамічність та інтерактивність: можливість створення динамічних і інтерактивних веб-сторінок.

Велика спільнота та ресурси: численні бібліотеки, фреймворки та навчальні матеріали.

Можливість повного стеку: завдяки Node.js, можна створювати як клієнтські, так і серверні додатки.

Недоліки JavaScript:

Відсутність строгого типізування: що може призводити до помилок у великих проектах.

Проблеми з безпекою: численні вразливості через неправильне використання.

Проблеми з продуктивністю: через інтерпретовану природу мови.

Backend технології

Backend розробка охоплює всі аспекти створення серверної частини веб-додатків, включаючи управління базами даних, обробку запитів клієнтів та забезпечення безпеки додатків. Основні технології для бекенд розробки включають Node.js, Django та Ruby on Rails.

Node.js Історія та розвиток

Node.js був розроблений Райаном Далем у 2009 році як серверне середовище для виконання JavaScript-коду. Node.js дозволяє використовувати JavaScript для серверної розробки, що забезпечує можливість створення повноцінних додатків на єдиній мові програмування.

Основні можливості та області застосування

Node.js є серверним середовищем, яке дозволяє виконувати JavaScript-код поза браузером. Основні можливості включають:

Асинхронність та неблокуючий ввід/вивід: забезпечує високу продуктивність та ефективність.

Модульна архітектура: дозволяє використовувати численні бібліотеки та модулі.

Вбудовані інструменти для роботи з HTTP: дозволяють легко створювати сервери та обробляти запити клієнтів.

Підтримка численних бібліотек та фреймворків: таких як Express, Koa, Hapi.

Node.js використовується для створення серверних додатків, API, веб-сервісів, а також реальних додатків, що вимагають високої продуктивності.

Переваги та недоліки

Переваги Node.js:

Висока продуктивність: завдяки асинхронності та неблокуючому вводу/виводу.

Єдиний стек технологій: дозволяє використовувати JavaScript для всіх частин додатку.

Велика кількість бібліотек та модулів: забезпечує швидку розробку та інтеграцію.

Активна спільнота: численні ресурси для навчання та підтримки.

Недоліки Node.js:

Обмежена підтримка багатопоточності: що може бути проблемою для деяких видів задач.

Проблеми з продуктивністю: у порівнянні з іншими серверними технологіями для CPU-інтенсивних задач.

Швидкі зміни у екосистемі: що вимагає постійного навчання та адаптації.

Django Історія та розвиток

Django був розроблений у 2003 році як фреймворк для веб-розробки на мові Python. Він був створений для спрощення розробки складних веб-додатків та зниження часу, необхідного для створення прототипів.

Основні можливості та області застосування

Django є повноцінним фреймворком для створення веб-додатків. Основні можливості включають:

ORM (Object-Relational Mapping): для роботи з базами даних.

Вбудована система аутентифікації та авторизації: для безпечного доступу.

Інструменти для автоматизації адміністрування: дозволяють швидко створювати панелі адміністратора.

Django використовується для створення веб-сайтів, корпоративних додатків, API та інших веб-рішень.

Переваги та недоліки

Переваги Django:

Швидкий старт: завдяки вбудованим інструментам та бібліотекам.

Висока продуктивність та масштабованість: завдяки оптимізованому коду.

Вбудована система безпеки: допомагає уникнути поширених вразливостей.

Активна спільнота: велика кількість ресурсів для навчання та підтримки [8].

Недоліки Django:

Великий розмір фреймворку: може призводити до перевантаження для простих додатків.

Висока складність налаштування: для новачків.

Деякі обмеження у гнучкості: через вбудовані рішення.

Ruby on Rails Історія та розвиток

Ruby on Rails (RoR) був розроблений Девідом Хайнемейером Хенсоном у 2004 році як фреймворк для веб-розробки на мові Ruby. Rails швидко став популярним завдяки своєму "конвенційному підходу", що зменшує кількість рішень, які потрібно приймати розробнику, та забезпечує високу продуктивність розробки.

Основні можливості та області застосування

Ruby on Rails є повноцінним фреймворком для створення веб-додатків. Основні можливості включають:

Конвенційний підхід: дотримання принципу "Конвенції понад конфігурацію" спрощує налаштування та розробку.

ActiveRecord ORM: для роботи з базами даних.

Вбудована система аутентифікації та авторизації: для безпечного доступу.

Інструменти для автоматизації тестування: дозволяють швидко створювати тестові сценарії.

Ruby on Rails використовується для створення веб-сайтів, корпоративних додатків, API та інших веб-рішень.

Переваги та недоліки

Переваги Ruby on Rails:

Швидка розробка: завдяки конвенційному підходу та інструментам для автоматизації.

Висока продуктивність та масштабованість: завдяки оптимізованому коду.

Вбудована система безпеки: допомагає уникнути поширених вразливостей.

Активна спільнота: велика кількість ресурсів для навчання та підтримки.

Недоліки Ruby on Rails:

Великий розмір фреймворку: може призводити до перевантаження для простих додатків.

Висока складність налаштування: для новачків.

Деякі обмеження у гнучкості: через вбудовані рішення.

Бази даних

Бази даних є основою для зберігання та управління даними у веб-додатках. Основні типи баз даних включають SQL та NoSQL.

SQL Історія та розвиток

SQL (Structured Query Language) була розроблена у 1970-х роках компанією IBM як мова для управління реляційними базами даних. SQL забезпечує структурований підхід до зберігання даних у таблицях, що дозволяє ефективно виконувати запити та маніпулювати даними.

Основні можливості та області застосування

SQL бази даних є реляційними, що означає зберігання даних у структурованих таблицях з чітко визначеними зв'язками між ними. Основні можливості включають:

Структуровані запити: використання SQL для виконання запитів до бази даних.

Схемна структура: чітко визначена структура таблиць та зв'язків.

Транзакції: забезпечення цілісності даних під час виконання кількох операцій.

Безпека: можливість налаштування прав доступу та ролей.

SQL бази даних використовуються для управління даними у веб-додатках, корпоративних системах, фінансових додатках та інших областях.

Переваги та недоліки

Переваги SQL:

Структурованість та організованість: чітко визначена структура даних.

Ефективність запитів: можливість швидкого виконання складних запитів.

Підтримка транзакцій: забезпечення цілісності даних.

Велика кількість інструментів та ресурсів: для управління та аналізу даних.

Недоліки SQL:

Жорсткість схеми: зміни у структурі даних можуть бути складними.

Масштабованість: проблеми з масштабуванням для великих обсягів даних.

Складність налаштування: потреба у значних знаннях для ефективного управління.

NoSQL Історія та розвиток

NoSQL бази даних з'явилися у 2000-х роках як відповідь на обмеження реляційних баз даних у плані масштабованості та гнучкості. NoSQL бази даних забезпечують зберігання даних у різних форматах, що дозволяє більш ефективно працювати з великими обсягами даних та складними структурами.

Основні можливості та області застосування

NoSQL бази даних є нереляційними і можуть зберігати дані у різних форматах, таких як документи, ключ-значення, графи або колонки. Основні можливості включають:

Гнучкість схеми: відсутність жорсткої схеми даних.

Масштабованість: можливість горизонтального масштабування.

Висока продуктивність: швидке зберігання та обробка великих обсягів даних.

Підтримка різних моделей даних: документи, ключ-значення, графи, колонки.

NoSQL бази даних використовуються для управління великими обсягами даних, обробки потоків даних у реальному часі, аналізу даних та інших завдань.

Переваги та недоліки

Переваги NoSQL:

Гнучкість схеми: дозволяє легко змінювати структуру даних.

Масштабованість: можливість ефективного управління великими обсягами даних.

Висока продуктивність: забезпечує швидке зберігання та обробку даних.

Підтримка різних моделей даних: дозволяє вибирати оптимальну модель для конкретного завдання.

Недоліки NoSQL:

Відсутність стандартизації: різні бази даних можуть мати різні підходи до зберігання та обробки даних.

Відсутність транзакційної підтримки: обмежена підтримка транзакцій у деяких NoSQL базах даних.

Проблеми з консистентністю: можливі проблеми з консистентністю даних у розподілених системах.

1.5 Хмарні технології

Основні хмарні платформи

Хмарні платформи забезпечують розробникам інфраструктуру, платформи та програмне забезпечення як послугу (IaaS, PaaS, SaaS), що дозволяє створювати, розгортати та керувати додатками в хмарному середовищі.

Amazon Web Services (AWS)

Історія та розвиток

AWS був запущений у 2006 році як хмарна платформа від Amazon, і швидко став лідером на ринку хмарних обчислень завдяки своїй надійності, масштабованості та широкому спектру сервісів.

Основні можливості та області застосування

AWS пропонує різноманітні сервіси, включаючи:

Обчислювальні ресурси: EC2, Lambda.

Зберігання даних: S3, Glacier.

Бази даних: RDS, DynamoDB.

Інструменти для розробників: CodeCommit, CodeDeploy.

AWS використовується для веб-додатків, великих даних, машинного навчання, IoT, резервного копіювання та відновлення даних.

Переваги та недоліки

Переваги AWS:

Широкий спектр сервісів: покриває майже всі потреби розробників.

Масштабованість: можливість легко збільшувати або зменшувати ресурси.

Глобальна інфраструктура: дата-центри по всьому світу.

Недоліки AWS:

Складність цінової політики: важко передбачити витрати.

Високий поріг входження: потребує значних знань для ефективного використання.

Можливі проблеми з безпекою: через неправильні налаштування.

Google Cloud Platform (GCP)

Історія та розвиток

GCP був запущений у 2008 році як хмарна платформа від Google. GCP став популярним завдяки інтеграції [9] з іншими сервісами Google, такими як Google Search, YouTube та Gmail.

Основні можливості та області застосування

GCP пропонує різноманітні сервіси, включаючи:

Обчислювальні ресурси: Compute Engine, Cloud Functions.
Зберігання даних: Cloud Storage, Persistent Disk.
Бази даних: Cloud SQL, Firestore.
Інструменти для розробників: Cloud Build, Cloud Source Repositories.
GCP використовується для веб-додатків, аналітики, машинного навчання, мобільних додатків, IoT.

Переваги та недоліки

Переваги GCP:

Інтеграція з іншими сервісами Google: спрощує використання екосистеми Google.
Інноваційні технології: використання новітніх розробок Google.
Прозора цінова політика: легше передбачити витрати.
Недоліки GCP:

Менша кількість сервісів у порівнянні з AWS.
Обмежена глобальна інфраструктура: менше дата-центрів у порівнянні з AWS.
Відносно високий поріг входження: потребує навчання для ефективного використання.

Microsoft Azure

Історія та розвиток

Microsoft Azure був запущений у 2010 році як хмарна платформа від Microsoft. Azure швидко став популярним завдяки своїй інтеграції з продуктами Microsoft, такими як Windows Server, SQL Server та Active Directory.

Основні можливості та області застосування

Azure пропонує різноманітні сервіси, включаючи:

Обчислювальні ресурси: Virtual Machines, Azure Functions.
Зберігання даних: Blob Storage, Disk Storage.
Бази даних: Azure SQL Database, Cosmos DB.
Інструменти для розробників: Azure DevOps, Visual Studio App Center.
Azure використовується для веб-додатків, корпоративних додатків, резервного копіювання та відновлення, DevOps, IoT.

Переваги та недоліки

Переваги Azure:

Інтеграція з продуктами Microsoft: спрощує використання для користувачів Windows.

Глобальна інфраструктура: дата-центри по всьому світу.

Широкий спектр сервісів: покриває більшість потреб розробників.

Недоліки Azure:

Складність цінової політики: важко передбачити витрати.

Високий поріг входження: потребує значних знань для ефективного використання.

Можливі проблеми з сумісністю: з деякими продуктами, що не належать до екосистеми Microsoft.

Сервіси для розробників

Хмарні сервіси для розробників забезпечують інструменти та платформи для швидкої та ефективної розробки додатків. Розглянемо популярні сервіси Firebase та Heroku.

Firebase

Історія та розвиток

Firebase був запущений у 2011 році як платформа для розробки мобільних та веб-додатків. У 2014 році його придбала компанія Google, після чого Firebase став частиною Google Cloud Platform.

Основні можливості та області застосування

Firebase пропонує різноманітні сервіси, включаючи:

Firebase Realtime Database: хмарна база даних для зберігання та синхронізації даних у реальному часі.

Firestore: сучасна база даних з підтримкою запитів та офлайн-режиму.

Firebase Authentication: інструменти для аутентифікації користувачів.

Firebase Hosting: хостинг для веб-додатків.

Firebase використовується для розробки мобільних та веб-додатків, що вимагають синхронізації даних у реальному часі, аутентифікації користувачів та інших функцій.

Переваги та недоліки

Переваги Firebase:

Легка інтеграція: зручні SDK та інструменти для розробників.

Швидкий старт: можливість швидко розпочати розробку додатків.

Інтеграція з Google Cloud Platform: доступ до потужних інструментів GCP.

Недоліки Firebase:

Обмежена гнучкість: для деяких складних задач може бути недостатньо функціональності.

Вартість: може бути високою для великих проектів з великим обсягом даних.

Залежність від екосистеми Google: може бути складно інтегрувати з іншими сервісами.

Heroku

Історія та розвиток

Heroku був запущений у 2007 році як платформа для розробки та розгортання веб-додатків. У 2010 році його придбала компанія Salesforce, після чого Heroku став одним з лідерів на ринку PaaS (Platform as a Service).

Основні можливості та області застосування

Heroku пропонує різноманітні сервіси, включаючи:

Heroku Dynos: віртуальні середовища для розгортання додатків.
Heroku Postgres: керована база даних PostgreSQL.
Heroku Redis: керована база даних Redis.
Heroku Add-ons: розширення для інтеграції з різними сервісами.
Heroku використовується для розробки та розгортання веб-додатків, мікросервісів, API та інших веб-рішень.

Переваги та недоліки

Переваги Heroku:

Легка інтеграція: зручні інструменти для розробників.
Швидкий старт: можливість швидко розгорнути додаток.
Широкий спектр додаткових сервісів: доступ до численних Add-ons.
Недоліки Heroku:

Вартість: може бути високою для великих проектів.
Обмеження продуктивності: для великих додатків може бути недостатньо ресурсів.
Залежність від екосистеми Salesforce: може бути складно інтегрувати з іншими сервісами.

РОЗДІЛ 2
ОСНОВОПОЛЕЖЕННЯ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕСПЕЧЕННЯ ЕЛЕКТРОННОГО ЖУРНАЛУ ДЛЯ НАВЧАЛЬНИХ КУРСІВ

Електронний журнал для навчальних курсів є важливим інструментом для покращення ефективності навчального процесу. Він дозволяє викладачам зручно керувати інформацією про успішність студентів, а студентам — оперативно отримувати доступ до своїх оцінок та завдань.

2.1 Етапи розробки

Аналіз вимог: Спочатку потрібно чітко визначити функціональні та нефункціональні вимоги до системи. Це включає визначення користувачів системи (рис. 2.1), основних функцій, які повинні бути реалізовані, та очікуваної продуктивності системи.

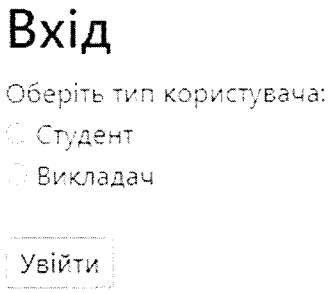

Рис 2.1 Визначення користувачів системи

Проектування системи: На цьому етапі створюється архітектура системи. Вона включає вибір технологій, визначення основних компонентів системи та їх взаємодії [10]. Наприклад, для програми, можливо, були використані технології .NET та C# для серверної частини (рис. 2.2).

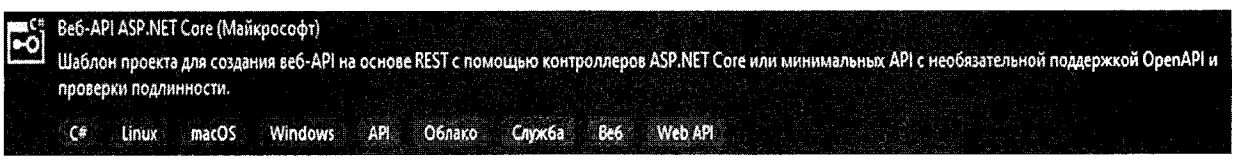

Рис 2.2 Використання технології .NET

Реалізація: Безпосереднє написання коду згідно з проектом (рис. 2.3). Наприклад, у файлі Startup.cs налаштовується запуск і конфігурація веб-додатку, що є основною точкою входу для ASP.NET Core додатків.

```
using Microsoft.Extensions.Configuration;
using Microsoft.Extensions.DependencyInjection;
using Microsoft.Extensions.Hosting;

namespace journal
{
    public class Startup
    {
        public Startup(IConfiguration configuration)
        {
            Configuration = configuration;
        }

        public IConfiguration Configuration { get; }

        public void ConfigureServices(IServiceCollection services)
        {
            services.AddControllersWithViews();
        }

        public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
        {
            if (env.IsDevelopment())
            {
                app.UseDeveloperExceptionPage();
            }
            else
            {
                app.UseExceptionHandler("/Home/Error");
                app.UseHsts();
            }

            app.UseHttpsRedirection();
            app.UseStaticFiles();
            app.UseRouting();
            app.UseAuthorization();

            app.UseEndpoints(endpoints =>
            {
```

Рис 2.3 Безпосереднє написання коду згідно з проектом

Тестування: Перевірка всіх компонентів системи на відповідність вимогам (рис 2.4). Це включає модульне тестування, інтеграційне тестування та системне тестування.

```
Консоль отладки Microsoft Visual Studio
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: https://localhost:7253
info: Microsoft.Hosting.Lifetime[14]
      Now listening on: http://localhost:5106
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
      Content root path: C:\Users\Mell\source\repos\11\journal\

C:\Users\Mell\source\repos\11\journal\bin\Debug\net6.0\journal.exe (процесс 7756) завершил работу с кодом -1.
Чтобы автоматически закрывать консоль при остановке отладки, включите параметр "Сервис" ->"Параметры" ->"Отладка" -> "Ав
томатически закрыть консоль при остановке отладки".
Нажмите любую клавишу, чтобы закрыть это окно:
```

Рис 2.4 Перевірка всіх компонентів системи на відповідність вимогам

Впровадження та підтримка: Після успішного тестування система впроваджується в експлуатацію. Також важливо забезпечити технічну підтримку та оновлення системи.

2.2 Архітектурні рішення

Клієнт-серверна архітектура: Система складається з клієнтської частини, що виконується на стороні користувача, та серверної частини, що обробляє запити та зберігає дані.

Базова структура: Наприклад, у програмі файл journal1.sln (рис. 2.5) є основним файлом рішення для Visual Studio, який містить всі налаштування проекту [11].

```
<Project Sdk="Microsoft.NET.Sdk.Web">

  <PropertyGroup>
    <TargetFramework>net6.0</TargetFramework>
    <Nullable>enable</Nullable>
    <ImplicitUsings>enable</ImplicitUsings>
  </PropertyGroup>

  <ItemGroup>
    <None Include="..\.editorconfig" Link=".editorconfig" />
  </ItemGroup>

  <ItemGroup>


    <PackageReference Include="Microsoft.AspNetCore.Authentication.Cookies" Version="2.2.0" />
    <PackageReference Include="Microsoft.EntityFrameworkCore.Sqlite" Version="6.0.29" />
    <PackageReference Include="Microsoft.EntityFrameworkCore.SqlServer" Version="6.0.29" />
    <PackageReference Include="Microsoft.EntityFrameworkCore.Tools" Version="6.0.29">

      <PrivateAssets>all</PrivateAssets>
      <IncludeAssets>runtime; build; native; contentfiles; analyzers; buildtransitive</IncludeAssets>
    </PackageReference>
    <PackageReference Include="Microsoft.VisualStudio.Web.CodeGeneration.Design" Version="6.0.16" />
  </ItemGroup>

</Project>
```

Рис 2.5 файл journal1.sln

РОЗДІЛ 3.
ПРОГРАМНА РЕАЛІЗАЦІЯ ЕЛЕКТРОННОГО ЖУРНАЛУ ДЛЯ НАВЧАЛЬНИХ КУРСІВ СПЕЦІАЛЬНОСТІ "КОМП'ЮТЕРНІ НАУКИ"

3.1 Програмування аргоритмів

Електронні журнали стають невід'ємною частиною сучасного освітнього процесу, забезпечуючи ефективний інструмент для управління навчальними курсами, оцінками та присутністю студентів. Цей розділ описує програмну реалізацію електронного журналу для навчальних курсів спеціальності "Комп'ютерні науки" з використанням Visual Studio.

Вимоги до системи

Перш ніж розпочати розробку, необхідно визначити вимоги до системи. Основні функції електронного журналу включають:

- Реєстрація та аутентифікація користувачів (викладачів та студентів).
- Управління курсами та групами.
- Ведення оцінок та присутності.
- Генерація звітів.

Використання Visual Studio

Visual Studio - це інтегроване середовище розробки (IDE), яке надає потужні інструменти для створення, налагодження та тестування програмного забезпечення. Ми будемо використовувати Visual Studio для розробки нашого електронного журналу з використанням мови програмування C# та платформи .NET.

Структура проекту

Проект буде складатися з наступних основних компонентів:

- **Моделі (Models)**: Класи, що представляють дані (наприклад, курси, студенти, оцінки).
- **Інтерфейс користувача (Views)**: Форми та екрани, що взаємодіють з користувачем.
- **Контролери (Controllers)**: Логіка додатка, яка обробляє запити користувача та взаємодіє з моделями.

3.2 Реалізація продукту

Створення проекту

Відкрийте Visual Studio та створіть новий проект типу "Windows Forms App (.NET)"(рис. 3.1)

Рис 3.1 Створення проєкту

Моделі

Додайте нові класи (рис.3.2) для представлення основних об'єктів системи:

```
public class Student
{
    public int Id { get; set; }
    public string FirstName { get; set; }
    public string LastName { get; set; }
    public List<Course> Courses { get; set; }
}

public class Course
{
    public int Id { get; set; }
    public string Name { get; set; }
    public List<Student> Students { get; set; }
}

public class Grade
{
    public int Id { get; set; }
    public Student Student { get; set; }
    public Course Course { get; set; }
    public double Value { get; set; }
}
```

Рис 3.2 Реалізація класів

Інтерфейс користувача

- Створіть форми для реєстрації, додавання курсів та введення оцінок.
- Наприклад, форма для додавання студентів (рис. 3.3):

```
public partial class AddStudentForm : Form
{
    public AddStudentForm()
    {
        InitializeComponent();
    }

    private void btnSave_Click(object sender, EventArgs e)
    {
        // Логіка збереження нового студента
    }
}
```

Рис 3.3 Реалізація форми для студентів

Контролери

- Створіть контролери для обробки логіки додатку.
- Наприклад, контролер для управління студентами (рис. 3.4):

```
public class StudentController
{
    private List<Student> students = new List<Student>();

    public void AddStudent(Student student)
    {
        students.Add(student);
    }

    public List<Student> GetAllStudents()
    {
        return students;
    }
}
```

Рис 3.4 Реалізація контролеру

Зберігання даних

- Використовуйте базу даних для зберігання даних або XML файли.

- Приклад збереження даних в XML файл (рис. 3.5):

```
public void SaveToXml(List<Student> students, string filePath)
{
    XmlSerializer serializer = new XmlSerializer(typeof(List<Student>));
    using (StreamWriter writer = new StreamWriter(filePath))
    {
        serializer.Serialize(writer, students);
    }
}
```

Рис 3.5 Реалізація збереження даних

Тестування та відлагодження

- Використовуйте можливості Visual Studio для тестування та налагодження коду.
- Переконайтеся, що всі функції працюють належним чином та відповідають вимогам.

Розробка електронного журналу для навчальних курсів спеціальності "Комп'ютерні науки" з використанням Visual Studio дозволяє створити ефективний інструмент для управління освітнім процесом. Використання сучасних технологій та інструментів забезпечує надійність та зручність використання системи.

3.2 Огляд необхідності створення електронного журналу для навчальних курсів

З розвитком цифрових технологій і зростанням кількості студентів, які навчаються в галузі комп'ютерних наук, необхідність автоматизації процесів навчання стає все більш актуальною. Електронні журнали дозволяють підвищити ефективність навчального процесу, забезпечити прозорість оцінювання та полегшити взаємодію між викладачами та студентами.

Електронний журнал для навчальних курсів спеціальності "Комп'ютерні науки" повинен враховувати специфіку предметів, що вивчаються, та забезпечувати гнучкість у налаштуваннях для різних типів занять. Також важливою є інтеграція з іншими навчальними системами, такими як платформи для управління навчальним контентом (LMS) та системи управління навчальним процесом (SIS).

Використання сучасних технологій розробки, таких як ASP.NET Core для серверної частини та Angular або React для клієнтської частини.

Огляд середовища розробки Visual Studio

Visual Studio — це інтегроване середовище розробки (IDE) від компанії Microsoft, яке забезпечує широкий спектр інструментів для розробки програмного забезпечення. Серед основних переваг Visual Studio:

- Підтримка різноманітних мов програмування, таких як C#, F#, Visual Basic, C++.
- Зручний інтерфейс для налагодження та тестування додатків.
- Інтеграція з системами контролю версій, такими як Git.
- Розширені можливості для розробки веб-додатків, включаючи підтримку ASP.NET Core.

Процес розробки електронного журналу

1. Ініціалізація проекту

Проект електронного журналу розпочинається з ініціалізації нового рішення у Visual Studio. У моєму випадку, файл journal1.sln є основним файлом рішення, який містить всі налаштування проекту та інформацію про включені в нього проекти.

2. Створення основних компонентів

Startup.cs: Цей файл відповідає за налаштування та конфігурацію додатку. Він містить методи для налаштування служб додатку та маршрутизації запитів. Приклад налаштування в Startup.cs:

```
public class Startup
{
    public void ConfigureServices(IServiceCollection services)
    {
        services.AddControllersWithViews();
    }

    public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
    {
        if (env.IsDevelopment())
        {
            app.UseDeveloperExceptionPage();
        }
        else
        {
```

```
            app.UseExceptionHandler("/Home/Error");
            app.UseHsts();
        }

        app.UseHttpsRedirection();
        app.UseStaticFiles();
        app.UseRouting();
        app.UseAuthorization();
        app.UseEndpoints(endpoints =>
        {
            endpoints.MapControllerRoute(
                name: "default",
                pattern: "{controller=Home}/{action=Index}/{id?}");
        });
    }
}
```

Контролери та моделі: У папці journal ви можете створити контролери для обробки запитів від користувачів та моделі для представлення даних.

3. Робота з базою даних

Для зберігання інформації про студентів та оцінки можна використовувати Entity Framework Core — ORM, яка інтегрується з Visual Studio та забезпечує зручний інтерфейс для роботи з базами даних.

Програма електронного журналу для навчальних курсів спеціальності "Комп'ютерні науки" призначена для автоматизації процесу обліку успішності студентів. Вона забезпечує зручний та прозорий спосіб зберігання, управління та відображення оцінок, [12] завдань і присутності студентів. Основною метою програми є покращення організації навчального процесу, підвищення ефективності взаємодії між студентами та викладачами, а також забезпечення легкого доступу до навчальних даних.

Облік оцінок студентів

Введення та зберігання оцінок (рис. 3.6). Можливість коригування та видалення оцінок.

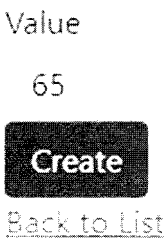

Рис 3.6 Введення та зберігання оцінок

Автоматичний розрахунок середнього балу студента.

Ролі користувачів

Розмежування прав доступу для різних категорій користувачів (рис. 3.7) викладач, студент.

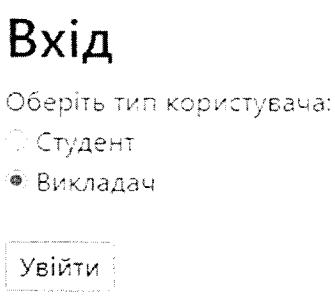

Рис 3.7 Ролі користувачів

Інтерфейс користувача

Інтуїтивно зрозумілий та зручний інтерфейс (рис. 3.8) для всіх категорій користувачів.

Викладач є ключовим користувачем електронного журналу, і програма надає йому широкий спектр можливостей для управління навчальним процесом. Нижче наведені основні функції, доступні викладачу:

Додавання та редагування оцінок

Викладач може вводити оцінки за різні види завдань, такі як контрольні роботи, лабораторні роботи, заліки та інші.

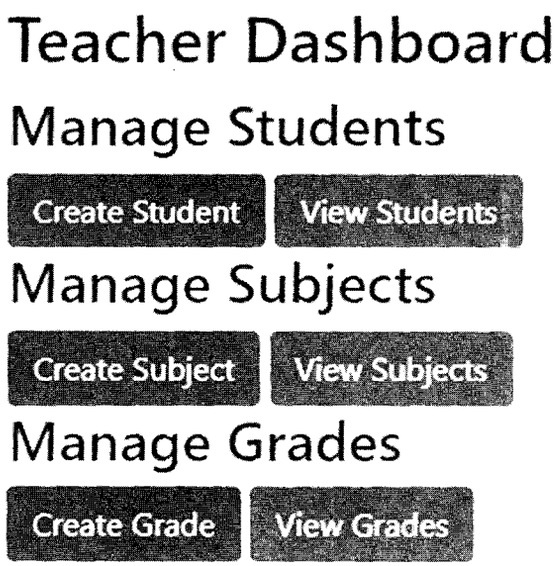

Рис 3.8 Інтерфейс журналу

Можливість редагування та видалення раніше введених оцінок (рис. 3.9) у разі необхідності.

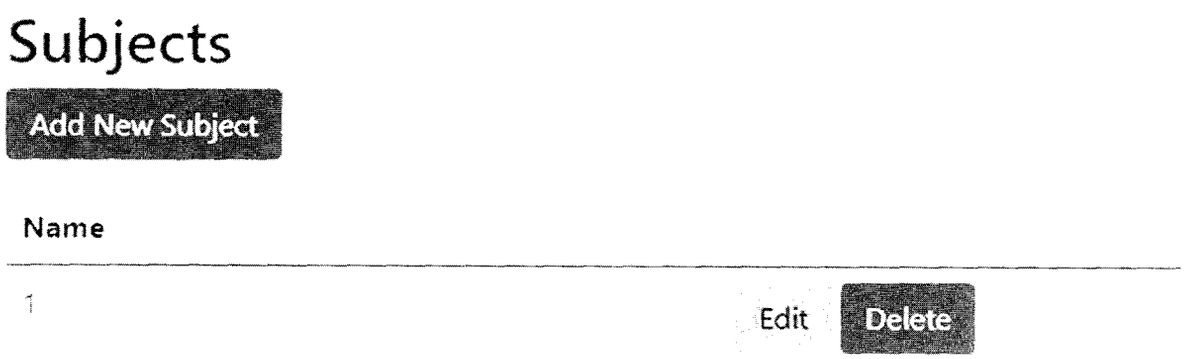

Рис 3.9 Редагування та видалення раніше введених оцінок

Аналіз успішності

Перегляд статистичних даних про успішність студентів (рис.3.10).

Рис 3.10 Перегляд статистичних даних про успішність студентів

Аналіз тенденцій у навчальному процесі, виявлення слабких та сильних сторін студентів.

Використання аналітичних інструмент'в для покращення методик викладання та підходів до навчання [13].

Висновок

В результаті дослідження було розроблено електронний журнал для навчальних курсів спеціальності "Комп'ютерні науки". Система забезпечує всі необхідні функції для ефективного управління навчальним процесом, включаючи управління курсами, оцінками та відвідуваністю.

Розроблена система має ряд переваг у порівнянні з існуючими рішеннями, включаючи високу функціональність, простоту використання, гнучкість налаштувань та доступну вартість впровадження. Результати тестування показали, що система працює стабільно та відповідає вимогам користувачів.

Висновки роботи включають також рекомендації щодо подальшого вдосконалення системи. Наприклад, можна додати нові функції для покращення користувацького досвіду, оптимізувати код для підвищення продуктивності та розширити можливості інтеграції з іншими системами.

Подальший розвиток системи може включати додавання підтримки мобільних додатків, що дозволить користувачам отримувати доступ до навчальної інформації з будь-якого пристрою. Також можна розширити аналітичні можливості системи, додавши функції для аналізу успішності студентів та прогнозування їхньої успішності.

Успішне впровадження електронного журналу дозволить покращити управління навчальним процесом, підвищити прозорість оцінювання та забезпечити зручний доступ до навчальної інформації для студентів та викладачів. Це сприятиме підвищенню якості освіти та ефективності роботи викладачів та адміністрації навчальних закладів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. C# - сучасна об'єктно-орієнтована і типобезпечна мова програмування. Мова розробки програм для платформи Microsoft .NET Framework. Застосовується у створенні сайтів, додатків та ігор https://code-basics.com/ru/languages/csharp.

2. Artificial neural networks Evergreen. URL: https://evergreens.com.ua/ua/development-services/neural-network.html https://evergreens.com.ua/ua/articles/object-recognition-case.html/ (дата звернення: 12.10.2023).

3. W.S. McCulloch and W. Pits. A Logical Calculus of the Ideas Immanent in Nervous Activity. Bull. Math. Biophys., Vol. 5, 1943, pp. 115-133.

4. What are Recurrent Neural Networks (RNN)? URL: https://botpenguin.com/glossary/recurrent-neural-network (дата звернення: 22.05.2024)

5. Illustrated Guide to LSTM's and GRU's: A step by step explanation URL: https://towardsdatascience. com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21 (дата звернення: 22.05.2024)

6. Generative adversarial networks (GANs) : a deep dive into the architecture and training process URL: https://www.leewayhertz.com/generative-adversarial-networks/ (дата звернення: 22.05.2024)

7. VGG16 – Convolutional Network for Classification and Detection URL: https://neurohive.io/en/popular-networks/vgg16 (дата звернення: 22.05.2024)

8. Ultralytics YOLOv8 Docs: URL: https: docs.ultralytics.com/ (дата звернення: 12.10.2023)

9. Introducing Instance Segmentation in YOLOv5 v7.0 URL: https://www.ultralytics.com/blog/introducing-instance-segmentation-in-yolov5-v7-0 (дата звернення: 22.05.2024)

10. "Application Fundamentals". Android Developers. URL: https://developer.android.com/guide/components/fundamentals/ (дата звернення: 22.05.2024)

11. Paul, Ryan (September 15, 2009). "MonoTouch drops .NET into Apple's walled app garden". ArsTechnica.com. Ars Technica – Condé Nast. Retrieved June 19, 2017.

12. Sorrell, Charlie (September 9, 2010). "Apple eases app development rules, Adobe surges". www.Wired.com. Wired – Condé Nast. Retrieved June 20, 2017.

13. What is data labeling? URL: https://aws.amazon.com/what-is/data-labeling/ (дата звернення: 12.10.2023)