МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра комп'ютерних наук

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: **«РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ОРГАНІЗАЦІЇ НАВЧАННЯ ПРОГРАМУВАННЮ УЧНІВ СТАРШОЇ ШКОЛИ»**

Виконав: студент 2 курсу, групи 8.1228-з
спеціальності 122 комп'ютерні науки
(шифр і назва спеціальності)
освітньої програми комп'ютерні науки
О.В.Чорний
(ініціали та прізвище)

Керівник доцент кафедри комп'ютерних наук, доцент, к.т.н. Матвіїшина Н.В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент завідувач кафедри програмної інженерії, доцент, к.ф.-м.н., Лісняк А.О.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра комп'ютерних наук

Рівень вищої освіти магістр

Спеціальність 122 комп'ютерні науки
(шифр і назва)

Освітня програма комп'ютерні науки

ЗАТВЕРДЖУЮ

Завідувач кафедри комп'ютерних наук, к.т.н., доцент
_____ Борю С.Ю.
(підпис)

« 30 » травня 2019 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Чорному Олексію Вікторовичу
(прізвище, ім'я та по-батькові)

1. Тема роботи (проекту) Розробка програмного забезпечення для організації навчання програмуванню учнів старшої школи

керівник роботи (проекту) Матвіїшина Надія Вікторівна, к.т.н., доцент
(прізвище, ім'я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 29 » травня 2019 року № 812-с

2. Строк подання студентом роботи 17 грудня 2019 року.

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
Огляд літератури та програмного забезпечення по темі, використання .NET Framework, опис використання розробленого програмного забезпечення.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Презентація кваліфікаційної роботи.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 30 травня 2019 р.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	15.06.19	
2.	Збір вихідних даних.	16.07.19	
3.	Обробка методичних та теоретичних	30.08.19	
	джерел.		
4.	Розробка першого розділу.	10.09.19	
5.	Розробка другого розділу.	03.10.19	
6.	Розробка третього розділу.	11.11.19	
7.	Оформлення та нормо контроль	19.11.19	
	кваліфікаційної роботи.		
8.	Захист кваліфікаційної роботи.	08.01.2020	

Студент _____ О.В. Чорний
(підпис) (ініціали та прізвище)

Керівник роботи _____ Н.В. Матвіїшина
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ О.Г. Спиця
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота магістра «Розробка програмного забезпечення для організації навчання програмуванню учнів старшої школи»: 39 с., 11 рис., 16 джерел, 1 додаток.

БАЗА ДАНИХ, ЕЛЕКТРОННЕ НАВЧАННЯ, НАВЧАЛЬНА СИСТЕМА, C#, PYTHON, .NET FRAMEWORK.

Об’єкт дослідження – електронне навчання програмуванню у старшій школі.

Мета роботи: створення програмного забезпечення, яке буде використовуватися для навчання програмуванню у старшій школі.

Метод дослідження – аналітичний, порівняння.

У роботі проведено аналіз поняття навчальної системи, розглянуто системи, які можуть використовуватися для навчанню програмування. Виділені недоліки використання даних систем у навчальному процесі.

З використанням платформи .NET Framework та мови програмування C# було розроблене програмне забезпечення для навчання програмуванню мовою Python у старшій школі, яке дозволяє учням вивчати теоретичний матеріал, вирішувати задачі з програмування не виходячи з системи, надсилати розв’язки вчителю. Вчитель має можливість перевіряти розв’язки задач, виставляти оцінки, надавати зворотній зв’язок учням у вигляді коментарів.

Розроблене програмне забезпечення планується використовувати у Комунальному закладі освіти «Спеціалізована школа №55 інформаційно-технологічного профілю» Дніпровської міської ради.

SUMMARY

Master's Qualification Thesis «Software Development for Organizing Teaching Programming for High School Students»: 39 pages, 11 figures, 16 references, 1 supplement.

DATABASE, E-LEARNING, LEARNING SYSTEM, C#, PYTHON, .NET FRAMEWORK.

The object of the study is e-learning of programming in high school.

The aim of the study is creating software that will be used to teach programming in high school.

The methods of research are analytical, comparison.

The paper analyzes the concept of the teching system, discusses the systems that can be used for learning programming. The disadvantages of using these systems in the educational process are highlighted.

A software for teaching Python programming in high school was developed. It allows students to study theoretical material, solve programming problems inside the system, and send solutions to the teacher. The teacher has the ability to check the solutions of the tasks, to evaluate solutions, to provide feedback to the students in the form of comments.

The developed software is planned to be used at the Municipal Educational Institution «Specialized School №55 of Information Technology Profile» of Dnipro City Council.

ЗМІСТ

ВСТУП

Одним із найважливіших розділів при вивчені інформатики у школі є вивчення основ алгоритмізації та програмування. Особливо важливим є цей розділ при профільному навчанні у старшій школі, оскільки у профільне вивчення інформатики передбачає підготовку учнів до участі в інтелектуальних змаганнях з програмування, формування в них стійкого інтересу до інформатики і пов’язаної з нею професійної діяльності, підготовку до навчання у вищих навчальних закладах.

Тим не менше, можна констатувати, що багато учнів мають складності з розумінням концепцій алгоритмізації та програмування. Тому є важливим використання у процесі навчання засобів, які підвищать зацікавленість учнів та сприятимуть процесу здобуття знань та навичок. Одним із таких засобів може стати система електронного навчання, яка матиме наступні можливості:

а) ознайомлення із стислим теоретичним матеріалом. Така можливість дає змогу не витрачати час уроку на написання конспектів, а також полегшує ознайомлення з матеріалом для учнів, які з певних причин пропустили зайняття;

б) наявність завдань з програмування, які можна виконувати безпосередньо у системі. Ця можливість особливо актуальна, якщо потужність апаратного забезпечення у класі інформатики робить проблематичним використання сучасних середовищ програмування;

в) надсилання розв’язків вчителю та можливість для вчителя виставляти оцінки і надавати коментарі щодо розв’язків у системі. Така можливість робить процес перевірки завдань більш швидким і простим, особливо якщо учні виконують завдання вдома.

Отже, метою даної роботи було ознайомлення з існуючими навчальними системами для навчання програмуванню, та створення власної навчальної системи з урахуванням недоліків існуючих систем.

У роботі проведено аналіз поняття навчальної системи, розглянуті найбільш популярні навчальні системи для навчання програмуванню. Розглянута технологія .NET Framework та її використання для створення власної навчальної системи у вигляді веб-додатку з використанням мови програмування C#. Описані можливості та особливості використання розробленого програмного забезпечення.

1 ОГЛЯД ЛІТЕРАТУРИ ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Наразі існує велика кількість програмних засобів, які можуть використовуватися для навчання інформатики, зокрема алгоритмізації та програмування.

Навчальна система – це апаратно-програмний комплекс, призначений для навчання користувачів.

Використання навчальних систем має наступні переваги:

а) можливість повністю перевести навчання у електронний формат, обмежити використання підручників. Особливо актуальним це є через те, що друковані підручники не завжди поставляються у школи, а підручники у електронному вигляді не мають зручної навігації;

б) збільшення активності учнів, надання можливості самостійного опанування теорії, зокрема для учнів що пропускали уроки з певних причин [1];

в) можливість швидко вносити зміни до навчального матеріалу, наприклад, при виявлені помилок;

г) адаптація до темпу роботи учня;

д) можливість інтерактивної перевірки завдань.

Всі навчальні системи в аспекті управління процесом навчання поділяються на дві групи [2]: системи, в яких управління навчальним процесом покладається безпосередньо на користувача, і системи, які здійснюють самостійне управління відповідними процесами навчання.

Системи, в яких управління навчальним процесом покладається безпосередньо на користувача, поділяються на такі підгрупи:

а) методичний посібник (електронний підручник) з аналоговою структурою, який можна фактично розглядати у якості електронної версії традиційного друкованого посібника або підручника;

б) методичний посібник (електронний підручник) з гіпертекстовою побудовою – візуалізація в електронному вигляді навчальної дисципліни забезпечується гіпертекстовою структурою;

в) база даних з повними текстами – забезпечує можливість звернення за посиланнями до авторських джерел дисципліни, що викладається, до подібних за тематикою текстів інших авторів. Вони можуть мати гіпертекстову побудову;

г) електронна бібліотека – система, яка здійснює управління електронними навчально-методичними матеріалами з різних навчальних дисциплін та надає можливість учневі здійснювати інформаційний пошук за різними параметрами, наприклад, за ключовими словами, за предметом навчання, темою тощо;

д) мультимедійний методичний посібник (електронний підручник). Викладання навчальної дисципліни здійснюється повністю або частково з демонстрацією в аудіо- або відеоформатах. Це дає можливість спостерігати розвиток предметів чи явищ, що досліджуються. Така система має можливість для розміщення всіх або декількох властивостей повнотекстових баз даних;

е) електронний методичний посібник (електронний підручник) із наявними засобами тематичного контролю – після проходження кожного розділу дисципліни формується певна оцінка, яка є основою для проведення самоконтролю самим учнем. Така система має можливість для розміщення всіх або декількох властивостей мультимедійних систем.

Навчальні системи, які самостійно здійснюють управління процесом навчання, мають у своєму складі викладення навчальної дисципліни в електронному вигляді в текстовому, графічному, аудіо- та відеоформатах [3]. Наприкінці кожної частини розділу навчальної дисципліни учневі надаються завдання для перевірки. У таких системах (на відміну від систем першої групи), відповіді і дії учня впливають на подальший хід даного процесу

навчання. Ступінь управління цим навчальним процесом насамперед залежить від ступеня адаптації системи щодо конкретного учня.

У зв'язку із цим навчальні системи даної групи діляться на підгрупи. Розподіл здійснюється за ступенем адаптивності та методами реалізації цієї адаптації:

а) автоматизована навчальна система із лінійною моделлю навчання. Характеризується тим, що структура викладення матеріалу в електронному вигляді є послідовною. Відповідно до результатів перевірки учневі надається наступна частина матеріалу. Якщо результати недостатні, він повертається, з урахуванням зауважень, для додаткового вивчення попередньої частини. Така система має всі чи декілька властивостей мультимедійних систем 1 групи;

б) автоматизована навчальна система із розгалуженою моделлю навчання. Особливістю цієї системи є те, що для кожної частини навчальної дисципліни системою визначається зразу кілька варіантів викладення матеріалу. Вони відрізняються за ступенем деталізації, глибиною викладу та мають декілька варіантів завдань для перевірки різних рівнів складності (пропонуються в кінці кожної частини). Така система адаптується за глибиною, рівнем деталізації викладу матеріалу, що досліджується, а також за рівнем складності завдань для перевірки. Система може формувати індивідуальний структурований рух навчання;

в) автоматизована навчальна система із адаптацією за формою викладу матеріалу. За даною системою учень сам має можливість обирати форму викладання дисципліни. Система може мати властивості автоматизованої навчальної системи з розгалуженою моделлю навчання;

г) автоматизована навчальна система із адаптацією за логікою викладення. Завдяки даній системі контроль за учнем здійснюється на підставі зіставлення моделей про предмет дослідження з боку вчителя (взятий за еталон) та учня.

Щодо взаємодії навчальної системи з користувачем автоматизовані навчальні системи поділяються на два базові групи: розімкнені (які не мають зворотного зв’язку) і замкнені (які мають зворотний зв’язок). Ці системи відрізняються різними підходами до процесу навчання.

Розімкнені автоматизовані навчальні системи не враховують відповіді учнів на поставлені питання і не коригують послідовність надання навчального матеріалу в залежності від ступеня засвоєння учнями відповідної теми. У них тільки виконується певна попередньо визначена програмними засобами послідовність викладення уроку чи низки контрольних питань. Найпростішими із числа розімкнених автоматизованих систем навчання є системи із презентацією, що фактично представляє собою послідовне чергування ланок «Автоматизована навчальна система» та «Учень».

Серед замкнутих систем значно поширеним видом автоматизованих навчальних систем є імітаційні автоматизовані навчальні системи. У них функції провідного елемент відіграє фактор моделювання реальної ситуації у певній сфері предметної області. Реакція учня на запропонований автоматизованою навчальною системою матеріал є елементом зворотного зв’язку та основою безперервного взаємодії у системі «Автоматизована навчальна система – Учень», оскільки будь-який вплив на систему з боку користувача відразу призводить до реакції у відповідь з боку навчальної системи. У якості прикладу таких автоматизованих навчальних систем можна назвати різного роду ігрові тренажери, імітатори тощо.

В імітаційних автоматизованих навчальних системах насамперед використовується комплексний підхід до процесу навчання. Програма не лише навчає, але ще й перевіряє фактично одночасно знання, отримані учнем на даний момент. При цьому, важливим фактором є учнівський відгук на той або інший інформаційний вплив. Навчальна система, враховуючи дані відгуку, може перебудовувати хід уроку в іншому напрямку. В

автоматизовану навчальну систему даної структури досить часто входять ігрові елементи. У даному випадку питання формується у вигляді ігрової ситуації, в результаті вирішення якої учень знаходить правильний чи неправильний вихід. У залежності від прийнятого учнем рішення, автоматизована навчальна система сформовує наступний варіант ігрової ситуації.

У рамках автоматизованих навчальних систем можуть вирішуватися такі завдання [4]:

а) завдання, пов’язані із реєстрацією та статистичним аналізом показників засвоєння навчального матеріалу: визначення часу розв’язування завдань, загального числа помилок тощо. До цієї групи також належать і завдання щодо управління навчальною діяльністю;

б) завдання, пов’язані з перевіркою рівня знань, умінь і навичок учнів перед і після навчання, визначенням їхніх індивідуальних здібностей і мотивацій;

в) завдання, пов’язані з підготовкою і поданням навчального матеріалу, адаптацією матеріалу за рівнями складності, підготовкою ілюстраційних матеріалів, контрольних завдань, лабораторних робіт, самостійних робіт тощо;

г) завдання адміністрування системи, доведення навчального матеріалу до робочих станцій і завдання зворотного зв’язку з учнями.

1.1 Використання систем управління навчанням

Одним із варіантів побудови автоматизованої навчальної системи (Learning Management System, LMS) є використання системи управління навчанням.

Системи управління навчанням являють собою платформу для розгортання електронного навчанння (e-Learning), але в ряді випадків можуть використовуватися і для адміністрування традиційного навчального процесу.

Від систем управління навчанням учень отримує можливості доступу до навчального порталу, який є початковою точкою для доставки всього навчального контенту, вибору відповідних навчальних шляхів на основі попереднього і проміжних тестувань, використання додаткових матеріалів за допомогою спеціальних посилань [5].

На даний момент найбільш популярною системою управління навчанням є Moodle [6] – модульне об’єктно-орієнтоване динамічне навчальне середовище, яка є вільною системою управління навчальним контентом. Підтримує більше 100 мов. Moodle дозволяє організувати навчання в процесі спільного вирішення навчальних завдань, здійснювати взаємний обмін знаннями, має широкі можливості для комунікації. Система підтримує обмін файлами між користувачами в різних текстових і графічних форматах. Система має ряд сервісів, систему повідомлень, сервіс–планувальник задач тощо. Одна з форм спілкування (форум) надає можливість організувати групове обговорення як навчальних проблем, так і використовувати його у вигляді системи новин. Чат дозволяє обмін повідомленнями при обговоренні, наприклад, навчальних завдань в режимі реального часу. Можна ще відзначити сервіси «Повідомлення» та «Коментар», які мають призначення індивідуальної комунікації викладача і студента: рецензування робіт, обговорення індивідуальних навчальних проблем. Учитель може створювати і використовувати в рамках курсу будь-яку систему оцінювання. Всі позначки з кожного курсу зберігаються у зведеній відомості. Moodle дозволяє контролювати активність учнів, стежити за їх навчальною роботою.

Moodle написана на PHP та розповсюджується під GNU General Public License.

Moodle можна завантажити і встановити на веб-сервер, наприклад NGIX. Ця система управління навчанням підтримує такі бази даних як PostgreSQL, MariaDB, MySQL, MSSQL.

Moodle підтримує наступні стандарти електронного навчання:

а) Sharable Content Object Reference Model (SCORM) – це сукупність стандартів та специфікацій електронного навчання, що визначають зв'язок між контентом на стороні клієнта та системою управління навчанням на стороні сервера, а також те, як слід інтегрувати у систему управління навчанням вміст, захищений авторським правом. Існує дві версії: SCORM 1.2 і SCORM 2004. Moodle сумісний з SCORM 1.2 і проходить усі тести в тестовому наборі ADL 1.2.7 для SCORM 1.2. SCORM 2004 не підтримується в Moodle;

б) стандарт AICC HACP для CMI був розроблений Комітетом авіаційної промисловості з питань комп'ютерного навчання (Aviation Industry Computer-Based Training Committee, AICC) та використовується для взаємодії з зовнішнім контентом від сторонніх авторів. Пакети AICC підтримуються в Moodle 2.1 та новіших версіях;

в) пакети IMS Common Cartridge також можна імпортувати в Moodle. Крім того, Moodle Book може експортуватися у вигляді пакетів вмісту IMS Content;

г) Learning Tools Interoperability (LTI) – це стандартний спосіб інтеграції багатофункціональних навчальних програм (часто вони розміщені віддалено та надаються через сторонні послуги) з навчальними платформами. Moodle використовує External Tool, щоб діяти як «споживач LTI» по стандарту, і може виконувати функції «постачальника LTI» за допомогою плагінів.

Claroline – це платформа e-Learning та e-Working (система управління навчанням), випущена під ліцензією GPL з відкритим кодом [7]. Платформа

використовується в більш ніж 100 країнах і доступна на 35 мовах. Написана на PHP, підтримує СУБД MySQL.

Платформа Claroline організована навколо концепції простору, пов'язаного з курсом або педагогічною діяльністю. Кожен простір курсу містить перелік інструментів, що дозволяють викладачу:

а) написати опис курсу;

б) додавати документи у будь-якому форматі;

в) управляти публічними або приватними форумами;

г) створювати шляхи навчання (сумісні з SCORM);

д) створювати групи користувачів;

е) створювати завдання (сумісні зі стандартом IMS / QTI standard2);

ж) структурувати навчальний план з завданнями та термінами;

з) розсилати сповіщення (в тому числі електронною поштою);

і) переглядати статистику відвідування та закінчення вправ;

к) використовувати вікі для написання спільних документів.

ATutor – це система управління навчанням з відкритим кодом, яка використовується для розробки та управління онлайн-курсами, а також для створення та розповсюдження контенту електронного навчання, що може використовуватися іншими середовищами [8].

Корисувачі ATutor можуть мають досуп до таких функцій:

Доступність: Широкий спектр функціональних можливостей гарантує, що користувачі з особливими потребами можуть повноцінно використовувати платформу як учні, викладачі або адміністратори. Підтримка IMS / ISO AccessForAll дозволяє учням конфігурувати середовище та вміст під свої конкретні потреби.

Соціальна мережа: усі користувачі ATutor можуть розвивати мережу контактів, створювати та приєднуватися до груп інтересів, налаштувати мережевий профіль.

Коли студент чи викладач заходять у «Мою стартову сторінку», подається список усієї поточної інформації, що забезпечує швидкий доступ до поточної діяльності на своїх курсах.

Усі користувачі системи ATutor мають папку «Вхідні», через яку вони можуть надсилати та отримувати приватні повідомлення від інших користувачів. Повідомлення, що надсилаються, зберігаються у надісланих повідомленнях, які зберігаються протягом встановленого періоду до видалення. Повідомлення можна експортувати та зберігати зовні.

Студенти можуть налаштувати свій профіль, додати особисту інформацію про себе. Фотогалерея може бути використана для створення альбому, де можна зберігати колекцію зображень.

Учні можуть пересуватися вмістом ATutor за допомогою глобальних, ієрархічних або послідовних засобів навігації. Елементи навігації можна приховати для спрощення середовища.

Учні можуть співпрацювати з іншими учасниками проектів, спілкуватися як група через форуми, обмінюватися ресурсами за допомогою утиліти File Storage та спільно розробляти документацію проекту. Вправи або завдання можна відправляти керівнику групи або інструктору курсу.

Усі користувачі системи ATutor мають власну утиліту зберігання файлів. Області зберігання файлів також можна розповсюдити по групах або по цілому курсу. Можна включити контроль версій для відстеження змін у документах.

Кожна група має доступ до свого власного блогу, до якого вони можуть публікувати повідомлення, доступні для всіх учасників курсу, або приватні повідомлення, доступні лише членам групи та викладачам. До публікацій блогу можна додавати математичні вирази Latex та мультимедійні об’єкти.

1.2 Онлайн платформи для навчання програмуванню

Наразі існує багато інтерактивних онлайн платформ, що призначені самостійного для вивчення програмування.

Загальний принцип роботи таких платформ наступний:

а) учню представляється теоретичний матеріал зайняття, і завдання, пов’язані з представленою теорією;

б) платформа має вбудоване вікно для написання коду;

в) учень відправляє результат, і виконується автоматична перевірка правильності розв’язку. Може існувати можливість надання підказок учню.

Таким чином, у порівнянні з використанням систем управління навчанням, онлайн платформи з вивчення програмування мають більшу інтерактивність взаємодії з учнями.

У 2019 році найбільш популярною мовою програмування для вивчення початківцями є Python, оскільки вона має простий синтаксис, і може використовуватися не лише у навчальних цілях, але й у комерційному програмуванні, і у наукових дослідженнях [9].

Єдиний український профільний підручник з інформатики для старшої школи автора Руденко В.Д. [10] також використовує мову Python.

Розглянемо декілька з найбільш популярних онлайн платформ, які дозволяють використовувати Python для вивчення програмування.

Англомовний сайт Codecademy [11] – популярна онлайн-платформа для навчання, розміщена в хмарі (див. рис. 1.1). Codecademy надає безкоштовні курси для різних мов комп’ютерного програмування.

Сайт також пропонує платну опцію «pro», яка надає користувачам доступ до персоналізованого плану навчання, вікторин, реалістичних проектів і допомоги від радників.

Зокрема, безкоштовним є курс вивчення Python2, але курс Python3 (який було випущено у 2008 році), є платним.

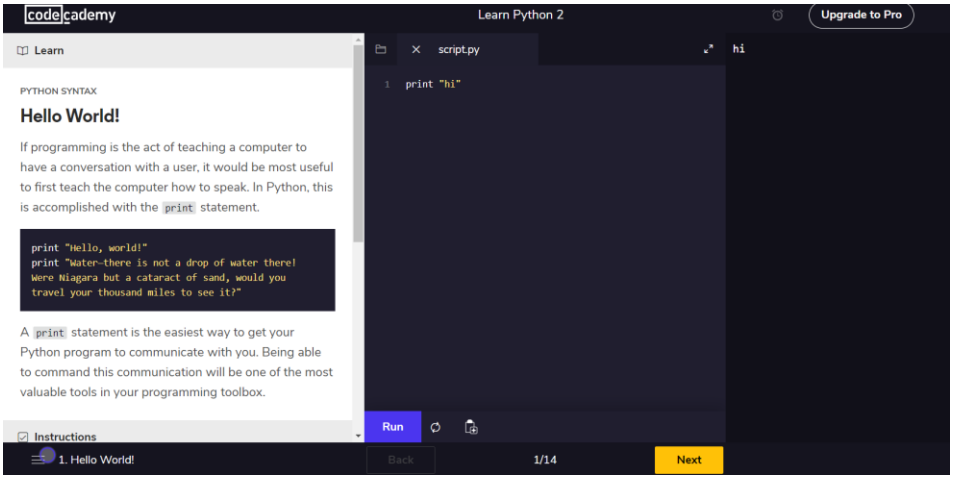

Рисунок 1.1 – Скріншот Codeacademy

Англомовний сайт learnpython.org [12] пропонує безкоштовне інтерактивне навчання Python 3 (див. рис. 1.2). Доступно подається теоретичний матеріал та пропонуються вправи з миттєвою перевіркою правильності коду. Однак не розглядаються деякі важливі теми, наприклад, рекурсія.

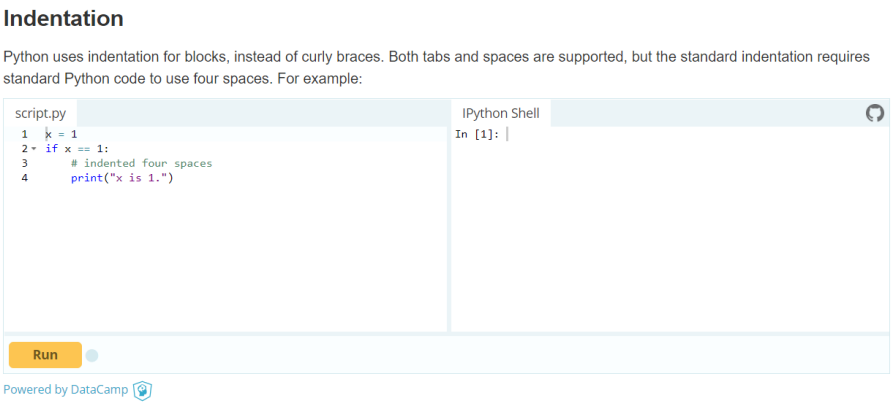

Рисунок 1.2 – Скріншот learnpython.org

Російськомовним сайтам для вивчення Python (наприклад, pythonworld.ru [13]) бракує інтерактивності (див. рис. 1.3). Акцент здебільшого робиться на теорію, а не на практику.

Эквивалент null в Python: None

Он был разработан таким образом, по двум причинам:

Многие утверждают, что слово *null* несколько эзотерично. Это не наиболее дружелюбное слово для новичков. Кроме того, **None** относится именно к требуемой функциональности - это ничего, и не имеет поведения.

Присвоить переменной значение None очень просто:

```
my_none_variable = None
```

Существует много случаев, когда следует использовать None.

Часто вы хотите выполнить действие, которое может работать либо завершиться неудачно. Используя None, вы можете проверить успех действия. Вот пример:

```
# Мы хотели бы подключиться к базе данных. Мы не знаем, верны ли логин и пароль
# Если соединение с базой будет неуспешно, то
# Он бросит исключение. Обратите внимание, что MyDatabase и DatabaseException
# НЕ являются реальными классами, мы просто используем их в качестве примеров.

try:
    database = MyDatabase(db_host, db_user, db_password, db_database)
    database_connection = database.connect()
except DatabaseException:
```

Рисунок 1.3 – Скріншот pythonworld.ru

Можна зробити висновок, що усі існуючі онлайн платформи, що використовують мову Python, навряд чи можуть використовуватися як засіб навчання програмуванню у школі, оскільки:

а) не існує платформ із локалізацією українською мовою;

б) відсутній розгляд важливих тем, що визначені програмою профільного вивчення інформатики у старшій школі [14];

в) не існує можливості редагування матеріалу;

г) не існує функціоналу оцінки завдань вчителем та отримання зворотного зв’язку від вчителя.

2 ВИКОРИСТАННЯ .NET FRAMEFORK

Для створення програмного забезпечення було обрано платформу .Net Framefork. Ця технологія розробляється корпорацією Microsoft і працює в основному на Microsoft Windows, хоча може бути використана також у Linux або macOS [15]. Вона включає в себе велику бібліотеку класів Framework Class Library (FCL) і забезпечує взаємодію різних мов програмування, таких як C#, F#, Visual Basic та інші. Кожна мова може використовувати код, написаний іншими мовами.

Можна виділити наступні переваги .Net Framefork:

а) велика кількість бібліотек, які прискорюють розробку програмного продукту;

б) забезпечення інформаційної безпеки за допомогою DataProtection API, Secret Manager, ASP.NET Identity;

в) порівняна простота розгортки додатків на сервері або у хмарі.

Одним із основних компонентів .NET Framework є Common Language Runtime (CLR). Він слугує середовищем виконання для .NET Framework і підтримує велику кількість можливостей, таких як управління пам’яттю, обробка винятків, збирання сміття, безпека та управління потоками. Усі програми, написані для .NET Framework, виконуються за допомогою CLR.

Код, написаний програмістом для .NET Framework, компілюється в код мови Common Intermediate Language (CIL). Під час виконання компілятор, що відповідає необхідній архітектурі (JIT), перетворює код CIL у машинний код.

Скомпільований код CIL зберігається у збірках. Відповідно до специфікації, збірки зберігаються у форматі файлів Portable Executable (PE), поширеного у Windows для всіх бібліотек dll та виконуваних файлів exe. Кожна збірка складається з одного або декількох файлів, один з яких повинен містити маніфест, що містить метадані збірки.

Серед мов програмування, які підтримує .NET Framework, для реалізації програмного проекту було обрано C# – об’єктно-орієнтовану мову програмування з строгою типізацією.

2.1 Використання ASP.NET MVC

ASP.NET MVC – фреймворк для створення веб-додатків з використанням патерна MVC [16]. Цей патерн поділяє додаток на 3 частини:

а) контролер (Controller) забезпечує зв’язок між користувачем і системою (приймає http-запит). За необхідності він звертається до моделі. Після обробки даних Контролер відправляє користувачу результат у вигляді Виду;

б) вид (View) – це візуальна частина додатку. Як правило, Вид являє собою html-сторінку;

в) модель (Model) описує логіку даних.

Цей патерн дозволяє реалізувати концепцію поділу відповідальності – зміни однієї компоненти не впливають на роботу інших частин програми.

Розроблений додаток має 5 Контролерів: HomeController, AccountController, StudentController, TeacherController, ExecuteCodeController.

HomeController слугує точкою входу в додаток. Після авторизації користувача від перенаправляє користувача на ту частину додатку, яка відповідає його ролі (учень або вчитель).

AccountController відповідає за вхід користувача у систему та створення нових облікових записів для учнів.

StudentController (додаток А) має три Action (методи класу). Action Index дозволяє учню продивитися загальний список уроків і оцінки за попередні уроки. Action Lesson відображає сторінку з текстом урока, завданням, формою для відладки коду програми, формою відправки

завдання. Action SubmitTask має атрибут HttpPost, який обмежує Action таким чином, щоб приймалися лише запити POST. Він дозволяє зберегти розв’язок, який відправив учень.

TeacherController має три Action. Action Index відображає сторінку з розв’язками учнів та форму для виставлення оцінок. Action SaveMark зберігає виставлену оцінку та коментар. Action AssignmentsArchive відображає архів розв’язків учнів. Action Marks відображає таблицю оцінок учнів.

ExecuteCodeController виконує код, надісланий учнем або вчителем, і повертає результат виконання коду. Виконання запиту до контролера та відображення результатів на сторінці відбувається за допомогою AJAX – технології, що дозволяє веб-додатку працювати асинхронно. Надісланий код та вхідні дані зберігаються як два тимчасових файла, потім виконується запуск інтерпретатора Python3, який зберігає результат виконання коду у третій тимчасовий файл. Вміст цього файлу передається користувачу та відображається у початковій формі.

Файл Global.asax містить налаштування роутінгу – механізму, який співставляє URL з Action котролера. Для роутінгу використовується таблиця роутів. Ця таблиця створюється при першому запуску веб-додату.

Коли веб-додаток вперше запускається, викликається метод Application_Start() із класу MvcApplication, описаному у Global.asax. Цей метод викликає метод RegisterRoutes(), а він, у свою чергу, створює таблицю роутів.

Таблиця роутів складається з одного маршруту. Цей маршрут розбиває всі вхідні запити на три сегменти. Перший сегмент відображається до імені контролера, другий сегмент відображається до імені Action, а кінцевий сегмент відображається до параметра під назвою Id.

Наприклад, URL-адреса Sudent/Lesson/2 розбивається на три параметри:

а) Controller = Student;

б) Action = Lesson;

в) Id = 3.

Таким чином, викликається Action Lesson у контролері Student, якому як параметр передається int Id = 3.

При виклику контролера система передає йому контекст запита. У цьому контексті зберігаються кукі, відправлені дані форми, строки запиту, ідентифікаційні дані користувача. Реалізація інтерфейсу IController дозволяє отримати контекст запиту в методі Execute() через параметр RequestContext.

Після виконання необхідних дій (наприклад, здійснення запитів до бази даних), контролер повертає результат, що належить типу ActionResult. ActionResult – це абстрактний клас, у якому визначений один метод ExecuteResult, що перевизначається в успадкованих класах. В більшості випадків результат належить класу ViewResult, який виконує рендерінг веб-сторінки. У випадку роботи з AJAX для виконання програм, написаних учнями, результат має тип JsonResult.

Рендерінг веб-сторінки відбувається за допомогою Razor – серверної мови розмітки, що входить до складу ASP.NET. Razor дозволяє вбудовувати серверний код C# у веб-сторінки.

Сторінка Razor має розширення cshtml, і має стандартну розмітку html, у яку вбудовується C# код, перед яким стоїть знак @. Наприклад, код для відображення списку уроків, якому як модель передається контролером List<Lesson>:

```
@foreach (Lesson lesson in Model)
<div class="list-group">
        <div class="list-group-item list-group-item-action">
            <div class="row">
                <div class="col-md-10"><h4>@lesson.Name</h4></div>
```

```
                <div class="col-md-2"><a class="btn btn-primary"
href="student/lesson?Id=@lesson.Id ">Переглянути</a></div>
            </div>
        </div>
</div>
```

2.2 Використання Entity Framework

Entity Framework - це ORM-фреймворк з відкритим кодом для платформи .NET. Він дозволяє працювати з даними, використовуючи об’єкти класів, не орієнтуючись на конкретні таблиці та поля бази даних. Entity Framework дозволяє працювати на більш високому рівні абстракції, коли обробка даних здійснюється у відповідності з об’єктно-орієнтованим підходом, а кількість коду є меншою порівняно з традиційними програмами.

Для підключення до бази даних через Entity Framework потрібен контекст даних. Контекст даних – це клас, успадкований від класу DbContext. Контекст даних містить набір типів DbSet <T>, де T представляє тип об’єкта, що окремо визначений як модель, і яких буде зберігатися в базі даних. Наприклад, таким чином додається до контексту даних модель (клас) Lesson:

```
public DbSet<Lesson> Lessons { get; set; }
```

Зв’язки бази даних створюються за допомогою навігаційних властивостей моделей. Розглянемо створення зв’язку один-до-багатьох між Lesson і Task.

```
public class Task
{       …
        public int? LessonId { get; set; }
        public Lesson Lesson { get; set; }
}
```

```
public class Lesson
{       …
        public ICollection<Task> Tasks { get; set; }
        public Lesson()
        {
           Tasks = new List<Task>();
        }
}
```

Таким чином, у класі Task Lesson є навігаційною властивістю, яка дозволяє отримати пов’язані з об’єктом дані з бази даних.

Загальна схема бази даних зображена на рисунку 2.1.

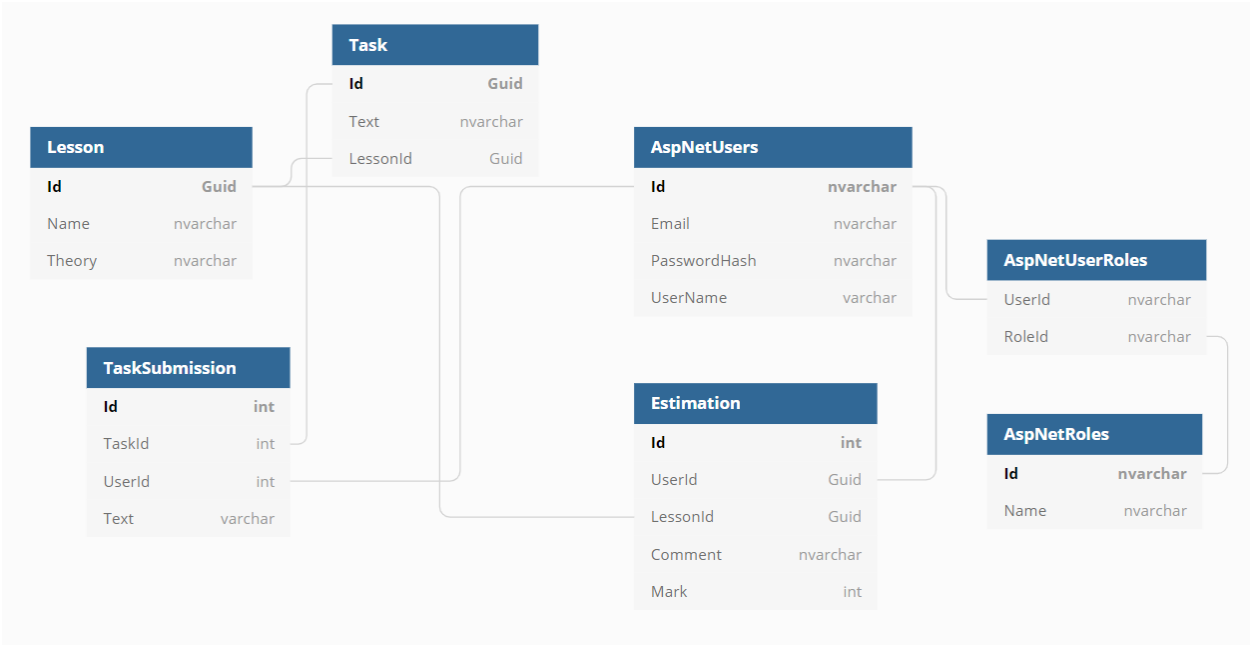

Рисунок 2.1 – Загальна схема бази даних

Для змін схеми бази даних у проекті використовувався механізм автоматичних міграцій, який забезпечує зміну схеми після зміни класів-моделей. При роботі з міграціями у консолі Visual Studio використовуються наступні команди:

а) Enable-Migrations дає можливість використовувати міграції у проекті та створює клас Configuration;

б) Add-Migration створює клас для нової міграції. Створений клас має два методи: Up() для проведення змін у базі даних та Down() для скасування змін;

в) Update-Database виконує останній файл міграції, створений командою Add-Migration, і застосовує зміни до схеми бази даних.

2.3 Використання ASP.NET Identity

Пакет ASP.NET забезпечує наступний функціонал:

а) API, який підтримує функцію входу в користувача систему;

б) керує користувачами, паролями, даними профілю, ролями, підтвердженням електронної пошти тощо.

ASP.NET Identity надає клас IdentityUser, який зберігає основну інформацію про аутентифікацію. Від базового класу IdentityUser успадковано клас ApplicationUser.

Клас IdentityRole представляє роль користувача. Роль має ім’я, з яким вона ототожнюється в системі. ASP.NET Identity забезпечує цю основну роль.

UserManager – це клас, який дозволяє керувати користувачами. Створення облікових записів користувачів, додавання ролей користувачам і подібні завдання виконуються за допомогою менеджера користувачів.

Role Manager – клас, який дозволяє керувати ролями. Створення ролі, видалення ролі, перевірка наявності ролі в системі виконуються за допомогою Role Manager.

Ролі створюються у файлі класі AppDbInitializer наступним чином:

```
var roleManager = new RoleManager<IdentityRole>(new RoleStore<IdentityRole>(context));
var role1 = new IdentityRole { Name = "teacher" };
var role2 = new IdentityRole { Name = "student" };
roleManager.Create(role1);
roleManager.Create(role2);
```

Контролери StudentController та TeacherController мають атрибут Authorize з вказанням ролі, який забезпечує надання доступу до сторінок тільки користувачам, які мають відповідні повноваження у системі.

Аутентифікацію користувача, тобто вхід у систему та вихід із системи, забезпечує Authentication Manager. Аутентифікація виконується на основі файлів cookie. Як менеджер аутентифікації ASP.NET Identity надає інтерфейс IAuthenticationManager.

3 ВИКОРИСТАННЯ РОЗРОБЛЕНОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Розроблене програмне забезпечення є клієнт-серверним веб-додатком.

При вході у систему користувач повинен заповнити форму для авторизації та аутенфікації (див. рис. 3.1). Логіни та паролі для учнів створює та видає вчитель, а логін та пароль вчителя створюються програмно при першому запуску додатку.

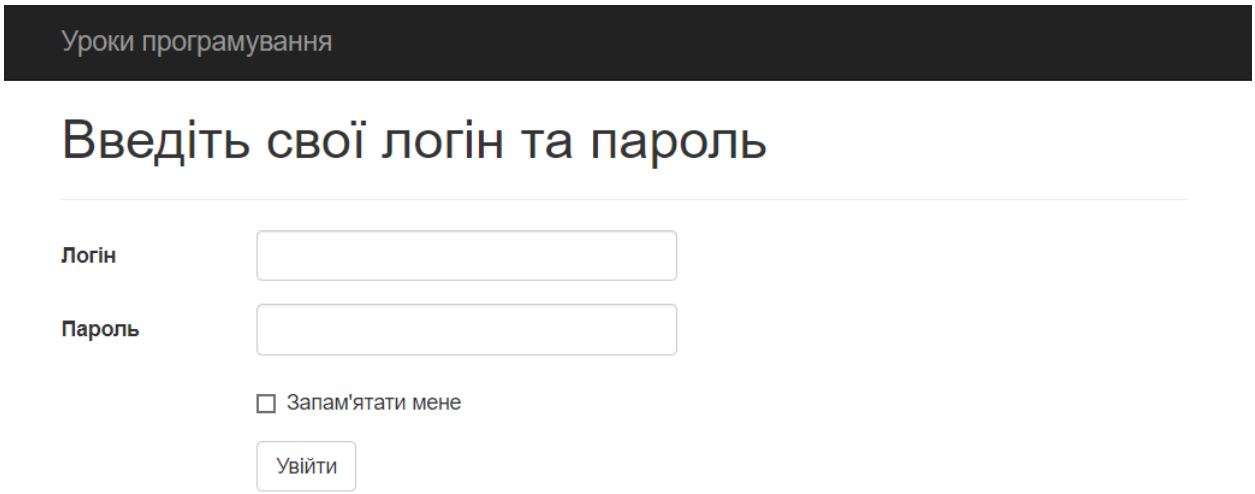

Рисунок 3.1 – Форма для входу в систему

Коли у систему входить учень, він бачить список уроків (див. рис. 3.2). Якщо учень вже отримав оцінки за попередні уроки, то вони відображаються біля відповідного уроку. Якщо вчитель додав до оцінки коментар, учень може його побачити, натиснувши на кнопку «Показати коментар вчителя». Натиснувши на кнопку «Переглянути», учень перейде на сторінку відповідного урока.

Строка меню однакова для всіх сторінок – вона дає можливість повернутися на головну сторінку або вийти з системи.

На сторінках уроків (див. рис. 3.3) учень, по-перше, ознайомлюється с теоретичним матеріалом. Якщо урок є практичною роботою, то теоретичний матеріал може бути відсутнім.

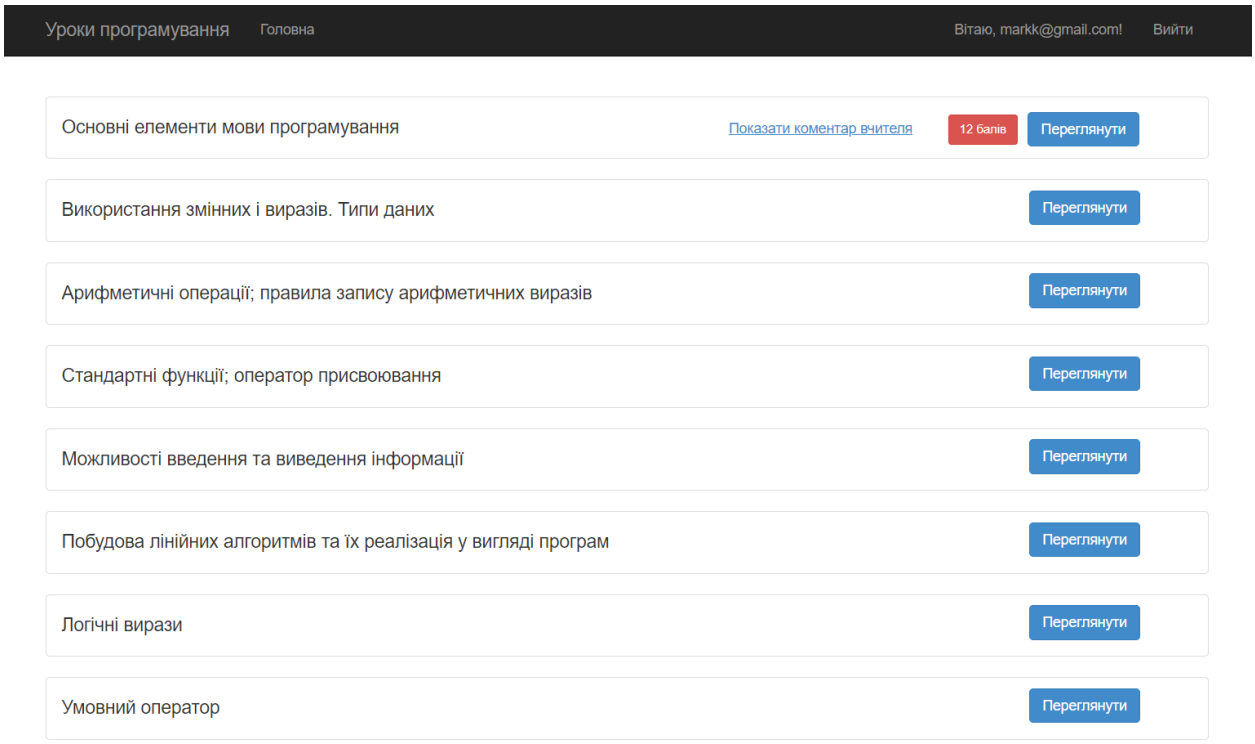

Рисунок 3.2 – Список уроків

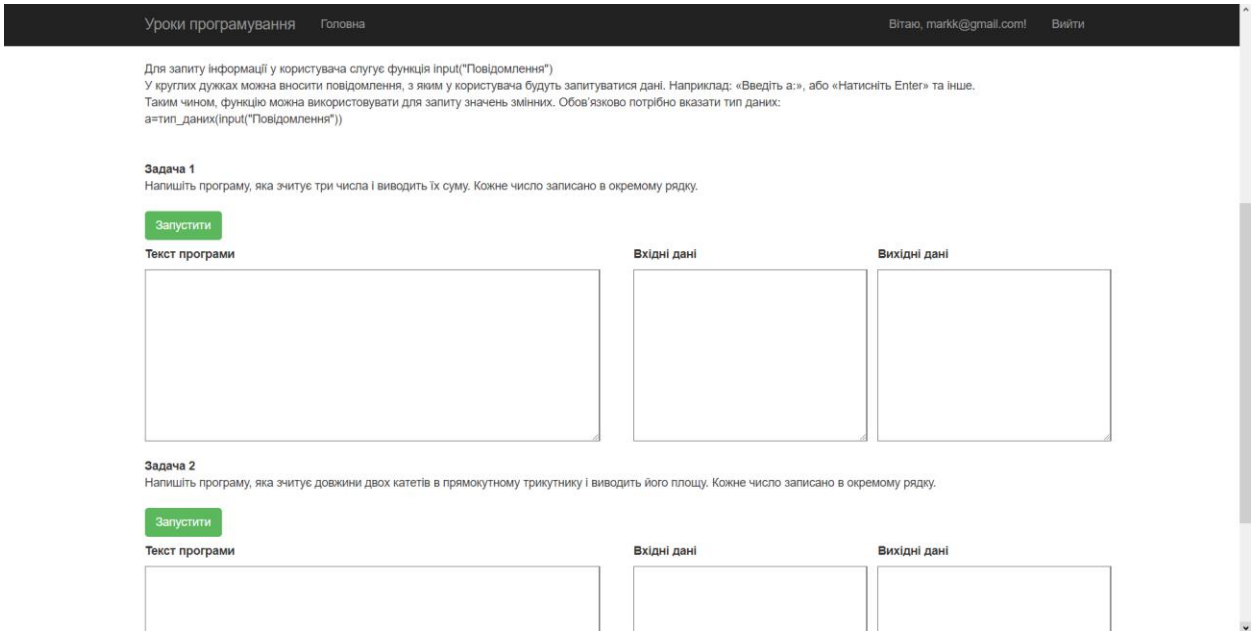

Рисунок 3.3 – Сторінка уроку

На рисунку 3.4 відображено приклад вводу учнем тексту програми та вхідних даних, та відображення вихідних даних після натиснення кнопки «Запустити».

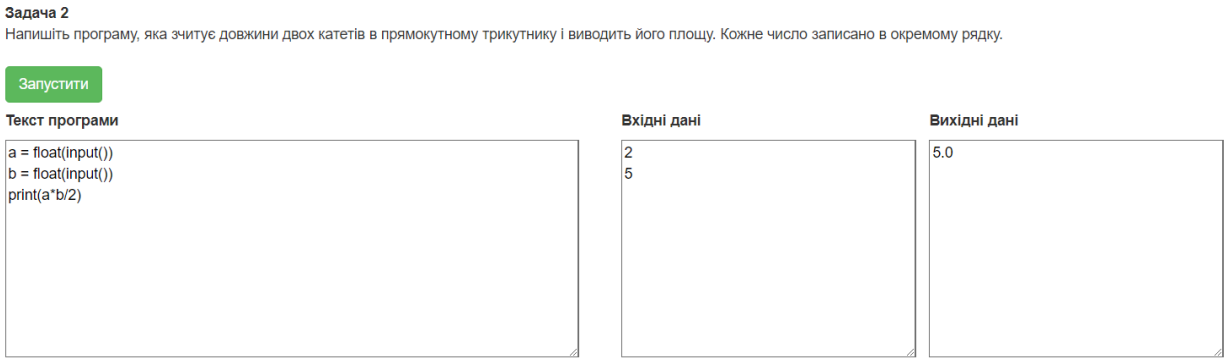

Рисунок 3.4 – Приклад вводу тексту програми та вхідних і вихідних даних

Після теоретичного матеріалу на сторінці уроку розташовані декілька завдань. Розв’язком кожного завдання є програма. Учень може писати текст програми безпосередньо у браузері. Вхідні дані заповнюються таким же чином, як вони вводились би у звичайну консольну програму. Натиснувши на кнопку «Запустити», учень може побачити результат виконання програми у полі «Вихідні дані». Закінчивши виконання завдань, учень натискає на кнопку «Відправити розв’язки вчителю», розташовану внизу сторінки.

Коли вчитель заходить у систему, він бачить розв’язки задач, які йому відправили учні (див. рис. 3.5). Розв'язки сортуються за датою за зростанням. Вчитель може запускати у браузері програми, написані учнями. Коли вчитель перевірив розв’язок, він виставляє оцінку та може написати коментар, наприклад, вказавши учню спосіб більш ефективний спосіб розв’язку задачі.

Посилання у строці меню «Додати учня» відкриває сторінку, яка дозволяє створити вчителю новий обліковий запис учня (див. рис. 3.6). Важливо вводити в одне поле спочатку Прізвище, а потім Ім’я для можливості сортування по прізвищам в подальшому.

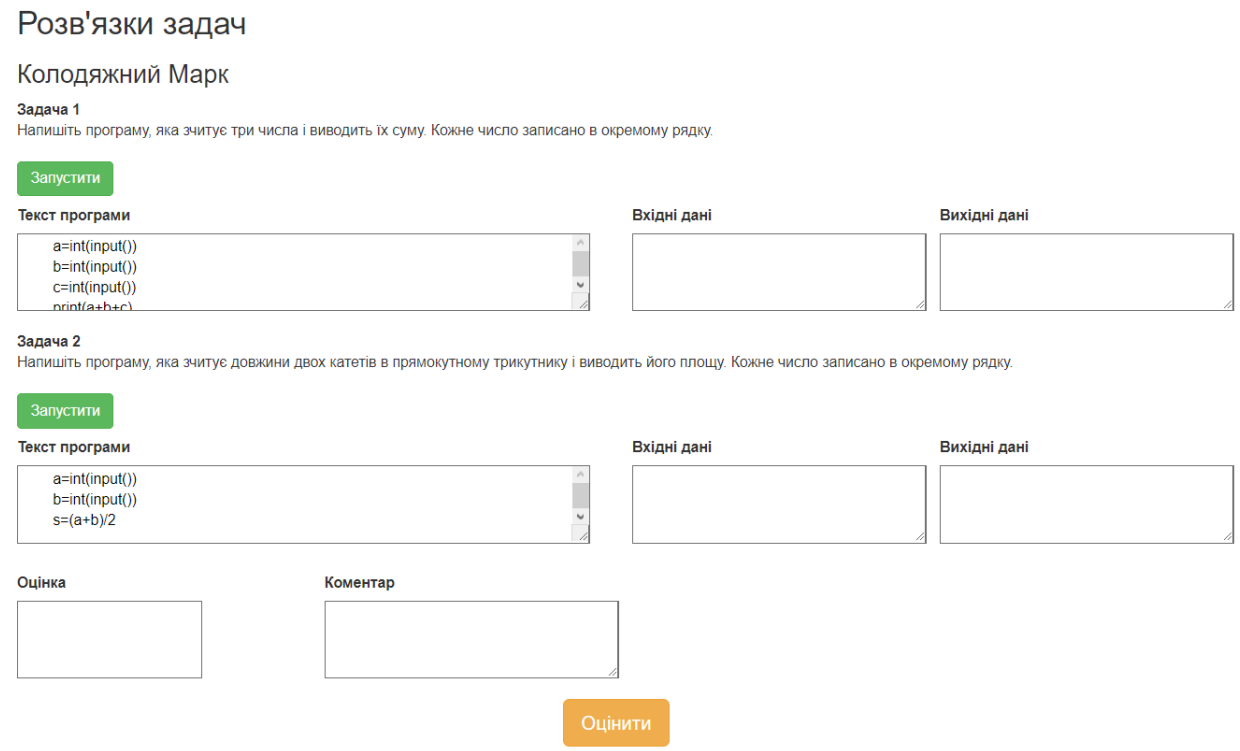

Рисунок 3.5 – Оцінювання розв’язків

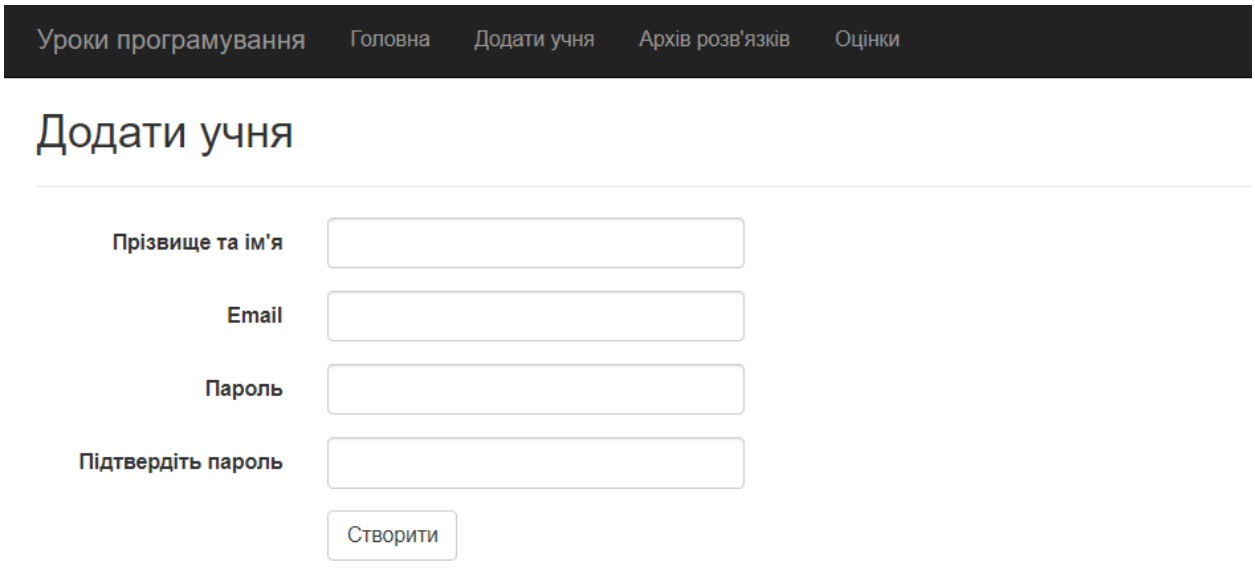

Рисунок 3.6 – Сторінка створення облікового запису учня

Посилання «Архів розв’язків» відкриває сторінку зі списком учнів (див. рис. 3.7). При виборі учня зі списку відбувається перехід на сторінку, де

відображені всі розв’язки учня, за які вчитель раніше виставив оцінку. Розв’язки мають сортування по даті спаданням.

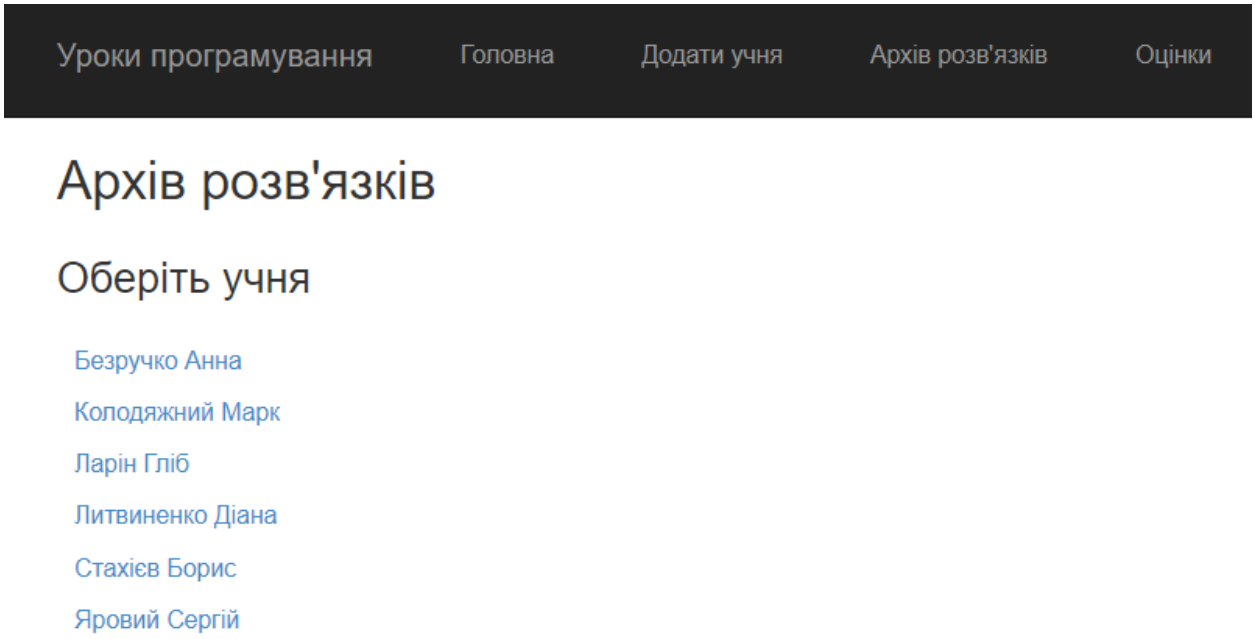

Рисунок 3.7 – Вибір учня для огляду його архіву розв’язків

Посилання «Оцінки» відкриває сторінку з таблицею, стовбці якої відповідають темам уроків, а строки – учням (див. рисунок 3.8).

Оцінки

	Основні елементи мови програмування	Використання змінних і виразів. Типи даних	Арифметичні операції; правила запису арифметичних виразів	Стандартні функції; оператор присвоювання	Можливості введення та виведення інформації	Побудова лінійних алгоритмів та їх реалізація у вигляді програм	Логічні вирази	Умовний оператор
Безручко Анна	8	9						
Колодяжний Марк	12	10						
Ларін Гліб	10							
Литвиненко Діана	7	6						
Стахієв Борис	5							
Яровий Сергій	11	11						

Рисунок 3.8 – Сторінка з таблицею оцінок

ВИСНОВКИ

У роботі проаналізовано поняття навчальної системи. Розглянуті навчальні системи, які можуть використовуватися для навчання програмуванню мовою Python. Описані недоліки використання існуючих навчальних систем у профільному навчанні інформатики у старшій школі, такі як відсутність отримання зворотного зв’язку від вчителя, відсутність україномовної локалізації, певна невідповідність навчального матеріалу навчальним програмам Міністерства освіти та науки України.

Описана розробка програмного забезпечення у вигляді веб-додатку з використанням .NET Framework, C# та пов’язаних технологій, призначеного для навчання програмуванню учнів старшої школи. Розроблене програмне забезпечення дає можливість учням вивчати теоретичний матеріал та вирішувати задачі з програмування, налагоджувати власні програми, отримувати зворотній зв’язок від вчителя щодо своїх розв’язків. Для вчителя дана навчальна система може полегшити процес перевірки завдань. Також систему можна використовувати для перевірки знань, наприклад, практичних або контрольних робіт.

При незначних модифікаціях розроблену систему можна адаптувати для використання у процесі навчання іншої мови програмування.

Планується використання розробленого програмного забезпечення при викладанні інформатики на профільному рівні у Комунальному закладі освіти «Спеціалізована школа №55 інформаційно-технологічного профілю» Дніпровської міської ради.

ПЕРЕЛІК ПОСИЛАНЬ

1. Сейфуллина А. О. Автоматизированные обучающие системы в образовательном процессе высших учебных заведений Казахстана. *Вопросы науки и техники* : сб. науч. трудов. Норильск : ЭКОР-книга, 2012. С. 115–122.
2. Бакалов В. П., Крук Б. И., Журавлева О. Б. Дистанционное обучение. Концепция, содержание, управление. Москва : Горячая линия – Телеком, 2007. 107 с.
3. Биков В. Ю. Дистанційна освіта : актуальність, особливості і принципи побудови, шляхи розвитку та сфера застосування. Інформаційне забезпечення навчально-виховного процесу: інноваційні засоби і технології. Київ : Атіка, 2005. С. 77–92.
4. Мандель Б. Р. Инновационные процессы в образовании и педагогическая инноватика. Москва, Берлин : Директ-Медиа, 2017. 340 с.
5. Kats Y. Learning Management System Technologies and Software Solutions for Online Teaching: Tools and Applications. Nashua : Rivier College, 2010. 486 с.
6. Moodle – Open-source learning platform. Moodle website. URL : https://moodle.org (дата звернення: 10.12.2019).
7. Claroline – Osez une pédagogie innovante avec notre LMS. Claroline website. URL: https://claroline.net
8. ATutor Features. ATutor website. URL : https://atutor.github.io/atutor/features.html (дата звернення: 10.12.2019).
9. Philip G. Python is Now the Most Popular Introductory Teaching Language at Top U.S. Universities. Communications of the ACM. URL :

http://cacm.acm.org/blogs/blog-cacm/176450-python-isnow-the-most-popular-introductory-teaching-language-at-top-us-universities (дата звереннння: 10.12.2019).

10. Руденко В. Д., Речич Н. В., Потієнко В. О. Інформатика (профільний рівень) : підручник для 10 класу закладів загальної середньої освіти. Харків : Ранок, 2018. 255 с.
11. Python tutorial. Codecademy website. URL : https://www.codecademy.com/courses/learn-python/lessons/python-syntax (дата звереннння: 10.12.2019).
12. Hello, world. Learn Python website. URL : https://www.learnpython.org/en/Hello%2C_World%21 (дата звереннння: 10.12.2019).
13. Python 3 для начинающих. Веб-сайт Pythonworld.Ru. URL : https://pythonworld.ru (дата звереннння: 10.12.2019).
14. Інформатика. Профільний рівень. Навчальна програма. Веб-сайт Міністерства освіти і науки України. URL : https://mon.gov.ua/storage/app/media/zagalna%20serednya/programy-10-11-klas/2018-2019/01/10-11-profilniy-riven.docx (дата звереннння: 10.12.2019).
15. NET | Free. Cross-platform. Open Source. Microsoft .Net website. URL : https://dotnet.microsoft.com (дата звереннння: 10.12.2019).
16. ASP.NET MVC Pattern. Microsoft .Net website. URL : https://dotnet.microsoft.com/apps/aspnet/mvc (дата звереннння: 10.12.2019).

ДОДАТОК А

Програмний код класу StudentController

```
using Microsoft.AspNet.Identity;
using ProgLessons.Models;
using ProgLessons.ViewModels;
using System;
using System.Collections.Generic;
using System.Data.Entity;
using System.Linq;
using System.Web;
using System.Web.Mvc;

namespace ProgLessons.Controllers
{
   [Authorize(Roles = "student")]
   public class StudentController : Controller
   {
      private ApplicationDbContext db = new ApplicationDbContext();

      [HttpGet]
      public ActionResult Index()
      {
         var data = (from l in db.Lessons
                     join c in db.TaskEstimations on l.Id equals c.Id into cJoin
                     from cj in cJoin.DefaultIfEmpty()
                     select new LessonViewModel()
                     {
                        Id = l.Id,
                        Name = l.Name,
```

```
                Mark = cj.Mark,
                Comment = cj.Comment
            }).ToList();
        return View(data);
    }

    [HttpGet]
    public ActionResult Lesson(Guid lessonId)
    {
        var lesson = db.Lessons.Where(l => l.Id == lessonId).Include(t => t.Tasks).FirstOrDefault();

        if(lesson == null)
            return HttpNotFound();

        return View(lesson);
    }

    [HttpPost]
    public ActionResult SubmitTask(string[] submissions)
    {
        string userId = User.Identity.GetUserId();
        List<TaskSubmission> taskSubmissions = new List<TaskSubmission>();
        for(int i = 0; i < submissions.Length; i = i+2)
        {
            taskSubmissions.Add(new TaskSubmission
            {
                Id = Guid.NewGuid(),
                TaskId = Guid.Parse(submissions[i]),
                Text = submissions[i+1],
                UserId = userId,
                CreationDate = DateTime.Now
            });
        }
```

```
            db.TaskSubmissions.AddRange(taskSubmissions);
            return View();
        }
    }
}
```