Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет ____Автоматики і комп'ютеризованих технологій_____
(повна назва)

Кафедра Комп'ютерно-інтегрованих технологій, автоматизації та мехатроніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти _____другий (магістрський)_____

«Розробка програмного засобу для розрахунку рейтингової оцінки студента»

_____
(тема)

Виконав: студент 6 курсу, гр. АУТПМ-21-1
Головко М. А
(прізвище, ініціали)
Спеціальність 151 Автоматизація та комп'ютерно-інтегровані технології
(код і повна назва спеціальності)
Тип програми Комп'ютерно-інтегровані технологічні процеси і виробництва
(код і повна назва напряму)
Тип програми освітньо-професійна
(повна назва освітньої програми)
Освітня програма Автоматизоване управління технологічними процесами
(повна назва освітньої програми)
Керівник доц. Бронніков А.І..
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Невлюдов І.Ш.
(підпис) (прізвище, ініціали)

2022 р.

Я, як студент ХНУРЕ, розумію і підтримую політику закладу із академічної доброчесності. Я не надавав і не одержував недозволену допомогу під час підготовки кваліфікаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

25.12. 2022 М. А. Головко

Харківський національний університет радіоелектроніки

Факультет АКТ

Кафедра КІТАМ

Рівень вищої освіти перший (бакалаврський)

Спеціальність 151 Автоматизація та комп'ютерно-інтегровані технології
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Автоматизоване управління технологічними процесами
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)
«_____»_____ 20 22 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Студентові Головко Максим Андрійович
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка програмного засобу для розрахунку рейтингової оцінки студента
затверджена наказом університету від 7 листопада 2022 р. №1463СТ
2. Термін подання студентом роботи до екзаменаційної комісії __ ___ 2022 р.
3. Вихідні дані до роботи
3.1 Призначення і цілі розробки: програмний засіб, призначений для розрахунку рейтингової оцінки студенту у Cloud- технологіях ;
3.2 Середовище та мова розробки: Angular JS;
3.3 Сумісність з ОС Windows XP, 7, 8, 10.
4. Перелік питань, що потрібно опрацювати в роботі
4.1 Вступ;
4.2 Аналіз існуючих автоматизованих систем;
4.3 Формалізація ключових аспектів автоматизованих систем;
4.4 Розробка програмного засобу для автоматизації розрахунку рейтингової оцінки студента;
4.5 Висновки.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) Демонстраційний матеріал представлений у форматі презентації PowerPoint (*.ppt) – 12 с. формату А4

6. Консультанти розділів роботи (п. 6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п. 1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1.	Аналіз літератури по темі роботи	2.11.2022	виконано
2.	Аналіз технічного завдання	12.11.2022	виконано
3.	Аналіз існуючих автоматизованих систем	14.11.2022	виконано
4.	Розробка структурної схеми програмного засобу розрахунку рейтингової оцінки студента	16.11.2022	виконано
5.	Розробка програмного забезпечення	25.11.2022	виконано
6.	Подання роботи на перевірку Інтернет-сервісом Unicheck	05.12.2022	виконано
7.	Оформлювання пояснювальної записки	06.12.2022	виконано
8.	Подання роботи на рецензію	08.12.2022	виконано
9.	Подання роботи на підпис зав. кафедри	21.11.2022	виконано
10.	Подання атестаційної роботи до ЕК	22.11.2022	виконано

Дата видачі завдання 11.05.2021 р.

Студент _____ Головко М.А.
(підпис) (прізвище, ініціали)

Керівник роботи _____ Бронніков А.І..
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 63 с., 2 табл., 15 рис., 1 дод., 17 джерел.

РОЗРОБКА, АВТОМАТИЗОВАНА СИСТЕМА РОЗРАХУНКУ, РЕЙТИНГОВА ОЦІНКА, РОЗРАХУНОК, CLOUD ТЕХНОЛОГІЇ

Мета кваліфікаційної роботи – автоматизація розрахунку рейтингової оцінки студента за допомогою Cloud-технологій.

Об’єкт розробки – процес розрахунку рейтингової оцінки студента.

Предмет розробки – рейтингові показники студентів.

Методи – теорія ймовірностей, розробка програмного забезпечення

У ході роботи було проаналізовано автоматизовані програмні засоби розрахунку у сфері освіти. Було виявлено недоліки існуючих засобів. Для вирішення недоліків існуючих засобів було розроблено програмний засіб розрахунку рейтингової оцінки студента та було проведено розрахунок та експериментальне дослідження надійності програмного засобу.

THE ABSTRACT

Explanatory note: 63 pp., 2 tables, 15 figs., 1 appendices, 17 sources.

DEVELOPMENT, AUTOMATED CALCULATION SYSTEM, RATING ASSESSMENT, CALCULATION, CLOUD

The purpose of the qualification work is to automate the calculation of the student's rating.

The object of development is the process of calculating a student's rating.

The subject of development is the rating indicators of students.

Methods – probability theory, software development.

In the course of work the automated software means of calculation in the field of education were considered. Weaknesses of existing means were revealed. To address the shortcomings of the existing tools, a software tool for calculating the student's rating was developed and a calculation and experimental study of the software's reliability was performed.

ПЕРЕЛІК СКОРОЧЕНЬ

АІС – автоматизована інформаційна система;

АС – автоматизована система;

АСР – автоматизована система розрахунків;

АСУ – автоматизована система управління;

АСУВ – автоматизована система управління виробництвом;

АСУП – автоматизована система управління підприємством;

БД – база даних;

ГАСУ – галузева автоматизована система управління;

ДСТУ – державні стандарти Україні;

ЕОМ – електронно-обчислювальна машина;

ІС – інформаційна система;

КСА – комплекс засобів автоматизації;

ПК – персональний комп’ютер;

ПЛІС – програмовані логічні інтегральні схеми;

ПЛК – програмований логічний контролер;

СДН – система дистанційного навчання;

СУБД – система управління базами даних;

ТП – технологічний процес.

ЗМІСТ

Перелік скорочень та термінів ……..... 6

Вступ …….……... 8

1 Аналіз існуючих автоматизованих систем……. …….……... 10

1.1 Аналіз існуючих програмних засобів у навчальній системі …… 10

1.2 Аналіз структури автоматизованих систем ……..…... 20

1.3 Аналіз бально-рейтингової системи ……... 25

2 Формалізація параметрів автоматизованої системи розрахунку рейтингової оцінки студента…… ……... 27

2.1 Формальний опис основних параметрів системи розрахунку рейтингової оцінки студента……. ……..…..... 27

2.2 Формалізація критеріїв оцінювання успішності студентів……..……..... 36

2.3 Аналіз та вибір середовища розробки автоматизованої системи розрахунку…….. ……...… 40

2.4 Аналіз та вибір бази даних автоматизованої системи розрахунку…….. ……...… 46

2.5 Аналіз та вибір середовища сумісної розробки автоматизованої системи розрахунку…….. …… 57

2.6 Використання теорії ймовірностей для передбачення майбутньої оцінки студента…….. …… 65

3 Розробка програмного засобу для розрахунку рейтингової оцінки студента ……. 66

3.1 Розробка алгоритму роботи програми……..……... 67

3.2 Розробка програмного забезпечення…….. ….……..... 75

3.3 Розробка людино-машинного інтерфейсу…. ……..... 80

3.4 Розрахунок надійності розробленого програмного засобу……..... 84

3.5 Забезпечення техніки безпеки при роботі з програмними засобами……... 85
Висновки ……..…….... 90
Перелік джерел посилання ..……..…. 91
Додаток А Демонстраційний матеріал ……...….. 93

ВСТУП

Одним із головних напрямків технічного прогресу є комплексна автоматизація розрахунків у всіх сферах діяльності – ряд технічних рішень, основне призначення яких полягає у впровадженні в систему програмного забезпечення, що мінімізує участь людини у процесах розрахунку [1].

Одним з основних напрямків, на які зараз звертає увагу наша країна, – це освітня система.

Для покращення якості освіти необхідно використовувати незалежну оцінку рейтингу студентів при його створенні а також у вимоги сучасного часу говорять про те, що для стабільного функціонування та забезпечення об’єктивності оцінювання будь-якого учбового закладу необхідно чітко контролювати роботу, вести звіти і управляти потоками інформації, а також прагнути до автоматизації. Тому автоматизація розрахунку рейтингової оцінки студента не є виключенням, і тема роботи є актуальною за наявністю великої кількості співробітників закладів освіти, які займаються розрахунком рейтингових оцінок студентів.

Мета кваліфікаційної роботи – автоматизація розрахунку рейтингової оцінки студента за допомогою Cloud-технологій.

Об’єкт розробки – процес розрахунку рейтингової оцінки студента.

Предмет розробки – рейтингові показники студентів.

Для досягнення поставленої мети необхідно вирішити такі завдання:

– аналіз сучасних засобів розрахунку в освіті;

– аналіз структури програмних засобів;

– розробка алгоритму роботи програми;

– проаналізувати та обрати середовище розробки автоматизованої системи;

– розробити людино-машинний інтерфейс;

– провести розрахунок надійності системи та експериментально перевірити правильність розрахунків;

– оформити пояснювальну записку згідно з рекомендаціями [2] та вимогами ДСТУ 3008:2015 [3].

1 АНАЛІЗ ІСНУЮЧИХ АВТОМАТИЗОВАНИХ СИСТЕМ

1.1 Аналіз існуючих програмних засобів у навчальній системі

1.1.1 Аналіз автоматизованого програмного засобу управління університетом.

Останніми роками реформування та модернізація освіти і науки в деяких країнах поглиблюється. Державна програма визначає стратегію, основні напрями, пріоритети, цілі та механізми реалізації державної політики у сфері освіти та є основою для зміни та доповнення структури та змісту законодавчої, адміністративно-фінансової, освітньої системи, кадрової та соціальної політики. План розвитку – необхідність фундаментальних реформ, спрямованих на підвищення якості освіти, стратегічний виклик для системи освіти в нових економічних і соціокультурних умовах. Одним із пріоритетів є автоматизація діяльності державного апарату, закладів освіти та культури.

Інформаційна система (ІС) – взаємопов’язаний набір інструментів, методів та персоналу, що використовуються для збору, зберігання, обробки та доставки інформації до завдання. ІС для нього необхідний у процесі прийняття рішень; вона допомагає аналізувати проблеми, прогнозувати та знаходити оптимальні рішення.

Автоматизована система (АС) – система, що складається з персоналу та комплексу засобів автоматизації, інформаційні технології реалізують виконання цих функцій.

Сучасні вищі навчальні заклади використовують у навчальному процесі десятки чи сотні ПК, або забезпечують їх впровадження. Інформаційні технології дозволяють значно підвищити ефективність використання комп’ютера. Вони дозволяють створювати ІС для ефективного управління організацією, дистанційного та автоматизованого навчання, зберігання, керування документами, обміну повідомленнями та спільної роботи над проектами. У цьому випадку принципово важливо постійно розширювати ІС і додавати нові компоненти або впроваджувати нові технології, які не вимагають коригування вже налагодженої

системи. Оскільки в процесі створення єдиного інформаційного простору ВНЗ існують різні способи отримання, передачі, зберігання та подання інформації, для реалізації цього різноманіття та забезпечення ефективної обробки інформації та її своєчасної доставки споживачам потрібна велика кількість систем.

Тому вибір моделі управління університетом має практичне значення, він дозволяє не тільки швидко побудувати просту та ефективну інформаційну систему, але й заощадити робочий час на першому етапі подальшого розширення та переходу до більш складної мережевої моделі [4].

Інтеграція ІС і АС створює інформаційну систему – людино-машинну систему, яка реалізує автоматичний збір і обробку інформації, необхідної для управління об’єктом прийняття рішень. АІС розроблено для оптимального контролю в будь-якій сфері. У цьому випадку об’єктами є університети, а сферою діяльності – освіта [1, 4].

АІС – система інформаційного забезпечення (звичайна АІС) із самостійним призначенням і сферою застосування. Прикладами таких систем можуть бути електронні бібліотечні каталоги, законодавство щодо АІС, системи електронного документообігу тощо;

Результат. Структура АІС складається з двох частин: функціональної та допоміжної системи. Функціональна частина являє собою групу підсистем, залежно від характеристик системи автоматичного керування. Ці підсистеми розділені деяким прапором (функціональним або структурним) і об'єднані відповідним набором завдань управління.

Допоміжною частиною є комплекс інформаційного, математичного, програмного, технічного, правового, організаційного, методичного, ергономічного програмного забезпечення.

Рисунок 1.1 показує структуру АІС, описує, які завдання втілювалися частинами структурної системи [4].

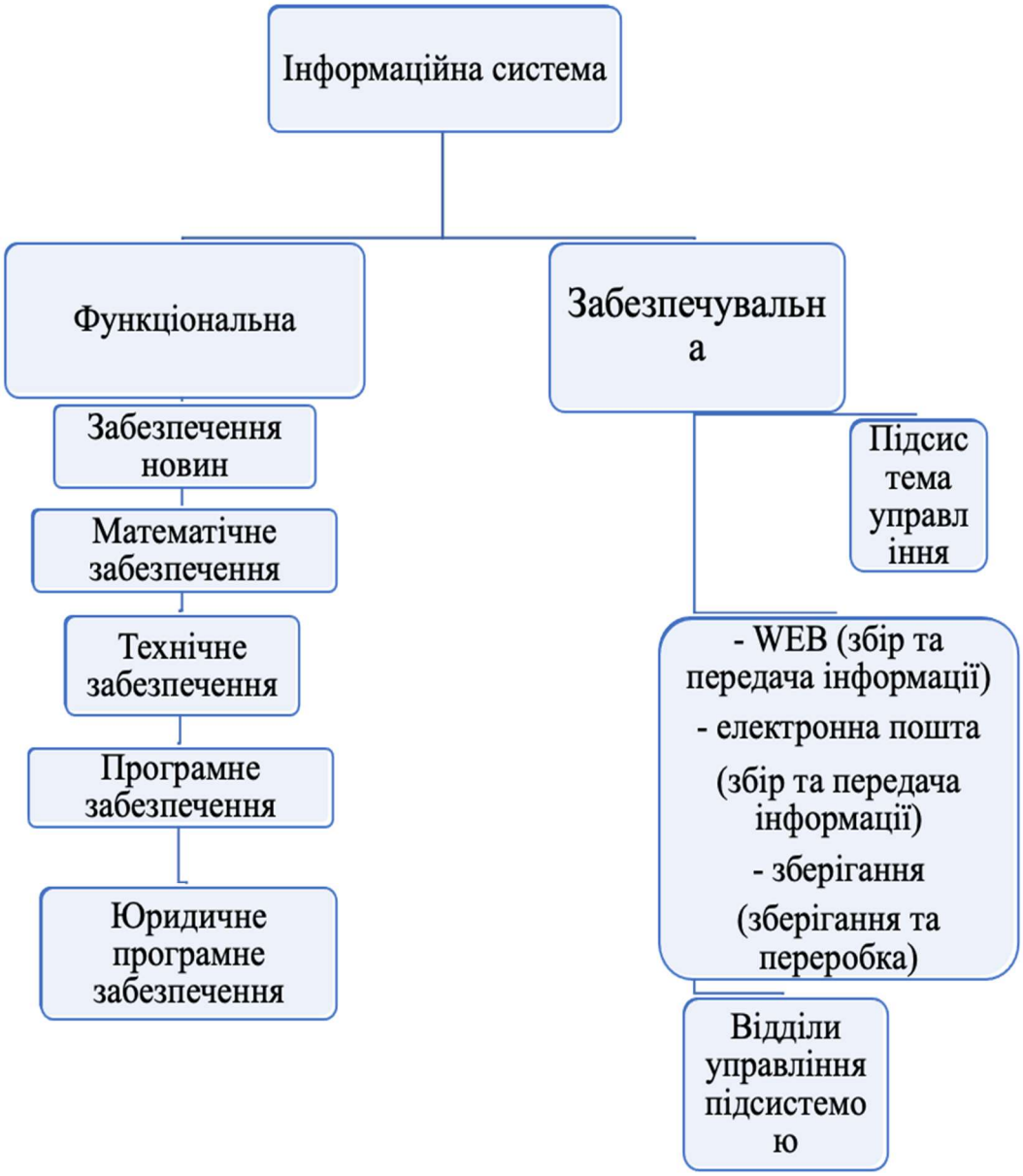

Рисунок 1.1 – Структура АІС

Тому розробка та впровадження будь-якої інформаційної системи хоча б у середній за розміром організації є надзвичайно складним завданням. Це пояснюється складною структурою і складними бізнес-процесами в організації.

Аналіз інформації отримано на базі університетської моделі технології Internet/Intranet.

Реалізація даної моделі може реалізувати вирішення комплексної проблеми автоматизації діяльності ВНЗ. Тобто роботу можна буде контролювати з єдиної точки, а єдиний файловий процес буде реалізовано на всіх кафедрах університету.

Система забезпечує централізоване зберігання даних, забезпечує універсальний доступ до баз даних за технологією Internet/Intranet, надсилає повідомлення, звіти, повідомлення електронною поштою.

Ефективність системи визначається за допомогою власних продуктів і технологій на базі найпопулярнішої програмно-технічної платформи (Microsoft), вартість впровадження невисока.

Однак необхідно підкреслити, що ІС – це інструмент для ефективного, якісного управління організацією, і насправді остаточні рішення приймаються лише відповідними підрозділами, а якість управління університетом залежить насамперед від якості професійних менеджерів та всієї команди.

Слід зазначити, що команда розробників ІС університету – інформаційний відділ. Вибір ІС базується на тому, що вони добре знають свої конкретні бізнес-процеси, собівартість розробок є досить невеликою, гарантовано високий рівень безпеки [4].

1.1.2 Аналіз засобу управління навчальним процесом Magellan.

Одномодульна система управління освітнім процесом Magellan включає електронний журнал успіхів, що дозволяє зберігати і узагальнювати дані про поточні успіхи і відвідуваність, а також формувати оцінки та навантаження вчителів у класах. Ця система призначена для використання у вищих навчальних закладах. Є лише дві рейтингові шкали, 5 та 10 балів. Його переваги включають інтеграцію з автоматизованими системами Moodle, гнучке налаштування функцій груп користувачів та створення звітів. Модулі представлені як окремими програмами, так і мобільною версією сайту. Цей продукт купується індивідуально для кожного покупця, тому комплектація та ціна системного комплекту можуть змінюватися.

Електронний журнал ElJour School розповсюджується безкоштовно, але доступний на запит. Існує функція виставлення оцінок, підтримуються різні системи оцінювання, є можливість коментувати роботи, оцінки, підручники, створювати різні звіти. Він має управління відвідуваністю та підтримує експорт

даних у Excel та PDF. У системі є реклама. Є мобільна версія програми, але немає десктопної.

Електронна університетська система має поточний модуль академічних досягнень. У його функції входить відвідування семінарів та практичних занять, робота в лабораторії, виконання курсових проектів, відправка домашніх завдань та тестів, а також підтримка прикордонного контролю та розвиток рейтингу. Модулі тісно пов'язані з іншими модулями у системі. Робота здійснюється завдяки веб-інтерфейсу, де всі зміни виконуються за допомогою миші.

Програмний комплекс ZT: Журнал хронографа відрізняє можливість автономної роботи з оцінками учнів за відсутності зв'язку із сервером збереження оцінок. Клієнти можуть встановлювати сервери самостійно, або використовувати готові централізовані сервери. Комплекс виконано у вигляді ще однієї програми для персональних комп'ютерів. Було розроблено веб-інтерфейс для виставлення оцінок учням. Серед інших особливостей варто відзначити лише дві системи оцінювання та три варіанти поділу навчального року на окремі періоди. Є демо-версія системи.

Багато коледжів розробили власні системи контролю за успішністю учнів. Це комп'ютерна програма, яка підтримує планування навчання, збереження та обробку оцінок, а також створення звітів. У дизайні дослідження лише зазначено, яка група якого предмета відповідає тій чи іншій формі контролю, і не передбачено велику кількість контрольних точок. Є управління груповими списками, видалення та додавання студентів. Підрахунок балів здійснюється у вигляді списку контрольних балів для кожного студента.

Національний дослідницький університет "Вища школа економіки" використовує унікальну систему електронних журналів LMS. Система досягла дуже високого рівня з погляду управління оцінками та розрахунку оцінок учнів. Отже, ви дізналися, як використовувати зважування для оцінки та завантаження всіх даних про учнів у Excel. Однак це не залежить від дати розміщення класу або налаштування. Незважаючи на наявність адаптивного веб-інтерфейсу, в мобільній версії відсутнє зручне введення оцінок.

Головною особливістю журналів вважатимуться наявність дедлайнів. Умови, за яких оцінка роботи не може бути максимальною. У цьому журналі контролю успішності є три різні терміни виконання лабораторних робіт, після кожного з яких оцінка знижується на задане значення. Єдиними недоліками є складність та тривалість процесу створення журналу для нового модуля, відсутність окремої версії для студентів та підрахунок результатів лише з одного предмета.

На закінчення можна сказати, що електронних журналів для шкіл та вишів досить багато. Для вирішення завдання розробки автоматизованої системи контролю успішності та підрахунку оцінок учнів важливо виділити необхідні особливості та відкинути ті, які явно не враховуються в проекті, що розробляється.

1.1.3. Аналіз програмного забезпечення для дистанційного навчання.

На сьогоднішній день існує безліч автоматизованих систем дистанційного навчання (DSL). До 2000 р. домінуючу роль відігравали відомі західні бренди, такі як Lotus Learning Space (IBM, США), WebCT (WebCT, США), Cisco та Oracle, але з 2001 р. їхнє місце зайняв iport SDN. зростати. Найбільш популярними системами дистанційного навчання є АСДН ДОЦЕНТ, Прометей, ОРОКС, Competentum Magister, eLearning 3000 та LMS Moodle. Розглянуто їх характеристики та проведено порівняльний аналіз можливостей систем дистанційного навчання для розробки комп'ютерних тестів. Автоматична система дистанційного навчання ДОЦЕНТ (центр дистанційного навчання) розроблена компанією ЮНІАР. Це комплекс високоефективних програмно-системних засобів дистанційного навчання, перепідготовки та тестування слухачів [6]. Основними перевагами систем дистанційного навчання є:

– можливість працювати з необмеженою кількістю користувачів на одному курсі одночасно;

– простота підключення нових навчальних курсів (або їх фрагментів), які створені в інших середовищах пристроїв;

– низькі вимоги до клієнтського комп'ютера та навігатора;

– індивідуальна генерація тестів;

– графічна оболонка для підготовки до тестів.

Наступний SDN, "ОРОКС", раніше був доступний під назвою "WEB-Tester". Це мережева оболонка для створення навчально-методичних модулів та організації навчальних процесів із використанням мережевих технологій.

Система призначена для підтримки сценаріїв навчальних процесів, віддаленого контролю знань, організації спільної роботи над предметом навчальних процесів, моніторингу процесів навчання.

Основними перевагами систем дистанційного навчання є:

– навчання на основі електронної навчальної програми;

– об'єднання в одну оболонку різноманітних інформаційних ресурсів, які забезпечують освітній процес;

– простота та оперативність створення навчальних модулів;

– управління навчальним процесом;

– управління регламентним процесом взаємодії викладачів та студентів за їх самостійної роботи;

– велика база даних [6] для зберігання результатів навчання модулів та контролю навчання.

SDN може створювати різні типи освітніх модулів, системи навчання та контролю, системи тестування та контролю.

SDN Prometheus є програмною оболонкою, здатною не тільки забезпечити дистанційне навчання та тестування студентів, але й управляти всією діяльністю у віртуальній установі, сприяючи швидкому впровадженню дистанційного навчання та його переходу в широке комерційне використання.

Інтерфейс перекладено кількома національними мовами, включаючи російську, українську, казахську, узбецьку (латиницю та кирилицю) та англійську.

Основними перевагами SDN є:

– організація курсової реєстрації на кшталт електронного магазину;

– використання календаря для вивчення курсу;

– гнучка підсистема обліку платежів (витрат);

– підсистема реєстрації/видачі сертифікатів;

– студенти можуть приєднатися до будь-якої кількості груп із одним логіном;

– можливість суміщення ролей;

– зберігання історії взаємодій із слухачами;

– створення навчальної програми, що об'єднує кілька курсів [7].

Програмний комплекс Фізикон - SDN Competentum призначений для автоматизації навчання та перевірки якості знань через локальні мережі або Інтернет. Магістр SDN встановлюється на сервері в локальній мережі навчального закладу та доступний для всіх студентів через Інтернет.

SDN Competentum.Magister має багато технічних характеристик, що забезпечують широкий діапазон універсальності. Система включає зручні інструменти для підготовки мультимедійних навчальних матеріалів, планування та контролю процесів навчання, розвинені механізми індикаторного аналізу.

Основною рушійною силою впровадження SDN eLearning 3000 стало переважання дуже дорогих зарубіжних SDN.

Основними перевагами систем дистанційного навчання є:

– створення мультимедійних навчальних курсів з графікою, відео та звуком;

– педагогічні технології: наука та практика, інтерактивні сцени з елементами анімації та підключення зовнішніх програм та об'єктів;

– формування системи інтерактивного тестування та вбудовані інтерактивні тести для дистанційної перевірки знань та оцінки успішності можуть бути завантажені безпосередньо з компакт-диска та доставлені через Інтернет із сервера освітнього центру, а результати тестування доставлені через Інтернет-канал направлений безпосередньо до Освітнього центру;

– наявність вбудованої пошукової системи. Ви можете шукати весь матеріал курсу та словник термінів;

– використання Інтернету для забезпечення зв'язку вчителів та учнів;

– підтримка спеціальних серверів у навчальних центрах, призначених для організації навчального процесу.

Наступна система, LMS Moodle (Learning Management System Modular Object-Oriented Dynamic Learning Environment), є найбільш перспективною платформою для дистанційного навчання. Кросплатформова система «Moodle» поширюється під Стандартною громадською ліцензією GNU, що гарантує свободу використання та розповсюдження. Розробку та підтримку системи забезпечують тисячі програмістів із багатьох країн світу. Загальна кількість користувачів перевищує 24 мільйони, а деякі сервери досягають 40 000 користувачів Moodle. Основними перевагами систем дистанційного навчання є:

– зосередження на методах спільного навчання;

– наявність широкого спектра можливостей для спілкування;

– можливість створювати та зберігати портфоліо для кожного студента;

– можливість використання будь-якої рейтингової системи, представлення матеріалів у будь-якому форматі (фото, відео, аудіо, текст);

– різні навчально-методичні матеріали (робочі зошити, лекції, вправи, уроки, тести);

– створення власних комплексних та інтегрованих курсів у вибраній вами області [6].

Система підтримує близько 20 елементів активності (форуми, глосарії, завдання, тести, Scrum-пакети, бази даних тощо) і кожен елемент можна використовувати в багатьох варіаціях.

Основна перевага діяльнісного підходу проявляється в об'єднанні елементів усередині послідовностей та груп, побудові освітньої траєкторії, за якою кожен діяльнісний елемент може враховувати попередні результати.

Дуже важливим елементом інтерактивної системи навчання Moodle є блок контролю знань. Ефективна реалізація тестової функції означає можливість формування різних контрольованих наборів даних (статистика результатів

тестування для конкретної групи учнів, статистика відсотка правильних відповідей для конкретної групи) на основі результатів тестів, які пройшли студенти. Це означає, що ви можете використовувати його для швидкого створення звітів. питання контролю його точності).

Важлива інформація, що реєструється під час тесту, – це запис часу, витраченого обмірковування кожного питання, кількість відповіді це питання і відсоток правильних відповідей нього. За підсумками цих параметрів здійснюється об'єктивна оцінка складності проблеми. Ця оцінка використовує випадкову вибірку питань із бази даних та може динамічно створювати тести однакової складності. За наявності достатньої кількості питань у навчальних матеріалах у базі даних з'являється можливість багаторазового тестування кожного студента та видачі тільки тих питань, на які ще не дано відповіді або на які дано неправильну відповідь.

Ефективність управління знаннями під час комп'ютерного тестування залежить від можливостей системи дистанційного навчання, на яку розроблявся тест. У зв'язку з цим особливе значення набуває питання вибору SDN розробки комп'ютерних тестів [6].

За результатами проведеного аналізу можна дійти невтішного висновку, що не розглянуті програми мають досить широким набором інструментів розробки та проведення комп'ютерних тестів. Тому найбільш підходящим середовищем для створення комп'ютерних тестів є програмна оболонка Moodle, що показав детальний аналіз представленого інструментального середовища та апробацію програм комп'ютерних тестів у найважливіших навчальних закладах. Він відповідає більшості стандартів, висунутих розробці комп'ютерних тестів. Оболонка Moodle може використовуватися для тестування як освітніми установами (ВНЗ, університетами, школами), так і іншими установами (наприклад, корпоративними відділами кадрів).

1.2 Структурний аналіз системи автоматизації

У процесі свого функціонування АС є сукупністю автоматизованих засобів (АСС), організаційно-методичних та технічних документів, а також фахівців, які використовують їх у своїй професійній діяльності.

У процесі проектування АС та її частин у цілому розробляють такі види забезпечення, як технічне, програмне, інформаційне, організаційно-методичне, метрологічне, юридичне, математичне, мовне, ергономічне [7].

Інформаційне забезпечення складається з єдиної системи класифікації та кодування інформації, єдиної системи документації, схеми інформаційного потоку, що циркулює в об'єктах автоматизації (підприємство, виробництво, комплексні об'єкти тощо) та методики побудови баз даних.

Призначенням підсистеми інформаційного забезпечення є своєчасне формування та видача достовірної інформації для ухвалення управлінських рішень на різних рівнях ієрархії.

Технічне забезпечення – комплекс технічних заходів, які забезпечують роботу автоматизованих систем, документування цих заходів та технічних процесів.

Комплекс технічних засобів складається з:

- комп'ютер (будь-яка платформа);
- пристрої для збирання, зберігання, обробки та виведення інформації;
- передавачі та приймачі даних та лінії зв'язку;
- оргтехніка та інше допоміжне обладнання;
- експлуатація та витратні матеріали тощо.

Централізована технічна підтримка ґрунтується на використанні великих комп'ютерів (мейнфреймів, суперкомп'ютерів) та обчислювальних центрів у складі автоматизованих систем.

Децентралізація технологічних засобів передбачає впровадження на робочих місцях складних функціональних підсистем операційної системи безпосередньо на робочих станціях, персональних комп'ютерах та промислових комп'ютерах.

Вочевидь, що перспективний підхід слід як частково децентралізований підхід. Це організація технічної підтримки, що розглядає розподілену мережу персональних комп'ютерів та великих комп'ютерів для зберігання загальних всім функціональних підсистем баз даних.

Централізована структура часто підходить для складних об'єктів або автоматизації процесів, що розташовані в обмеженому географічному просторі. Наприклад, для об'єктів, що знаходяться в різних будинках, на віддалених один від одного відгалуженних або розподілених на великій території, таких як насосні станції (газові, нафтові), електричні мережі тощо. Єдиною відповідною структурою є розподілена структура. Математичні та програмні – сукупність математичних методів, моделей, алгоритмів та програм для реалізації цілей та завдань автоматизованих систем, а також звичайного функціонування складних технічних засобів.

До засобів математичного забезпечення належать:

– інструменти моделювання систем та процесів управління;

– типові алгоритми управління обладнанням та технологічними процесами;

– такі методи, як математична теорія систем, системна інженерія, математична статистика, теорія масового обслуговування та математичне програмування.

Загальносистемне програмне забезпечення містить набір орієнтованих користувача програм, призначених на вирішення типових завдань обробки та управління інформацією. Вони розширюють можливості вашого комп'ютера та допомагають контролювати та керувати процесами обробки даних.

Спеціалізоване програмне забезпечення – це сукупність програм, розроблених під час створення тієї чи іншої системи автоматизації.

Сюди входять пакети прикладних програм, які у тому часі мають відповідні моделі розробки, відбивають функціональність реальних об'єктів.

Технічна документація програмного засобу має включати опис задачі, її алгоритмізацію, економічну та математичну моделі задачі, контрольні приклади. Організаційне забезпечення являє собою сукупність методів та засобів регулювання взаємодії працівників з технічними засобами та один з одним у процесі розробки та експлуатації АСУ. Організаційне забезпечення розробляється на основі результатів передпроектного вивчення організації (підприємства, виробництва) та реалізує такі функції:

– аналіз існуючих систем управління об'єктами, в яких використовується СКУД, та визначення завдань, які необхідно автоматизувати;

– підготовка машинорозв'язуваних завдань, у тому числі техніко-економічних обґрунтувань технічних завдань та ефективності на проектування АСУ;

– розробка управлінських рішень щодо організаційного складу та структури, методології вирішення завдань, спрямованих на підвищення ефективності.

Правове забезпечення являє собою сукупність правових норм, що визначають створення, правовий статус та функціонування підсистем управління інформацією та систем загалом, що регламентують порядок отримання, перетворення та використання інформації.

Основною метою правового захисту є забезпечення законності. Він ґрунтується на законах, указах, постановах органів державної влади, наказах, директивах та інших нормативних документах міністерств, відомств, організацій та органів місцевого самоврядування.

У правових нормах можна розрізняти загальну частину, регулюючу функціонування систем автоматичного управління, як інформаційних, і систем управління, і локальну частину, регулюючу функціонування конкретних підсистем.

Правове забезпечення включає в себе стан системи, правову та матеріальну відповідальність розробників (постачальників) та замовників, права, обов'язки та відповідальність персоналу, правове забезпечення окремих видів управлінських процесів, процедури створення та використання інформації та ін.

Програмно-технічне та інформаційне забезпечення проектних рішень - це ті частини, які реалізуються як вироби у вигляді набору взаємопов'язаних компонентів та комплексів, що входять до складу АС та містять іншу необхідну документацію. На рис. 1.2 представлена класична схема автоматизації [8].

Рисунок 1.2 – Класична піраміда автоматизації

Проектні рішення по іншим видам забезпечень входять до складу АС та їх частин як організаційно-методичних та експлуатаційних документів або реалізують в компонентах програмного, технічного або інформаційного забезпечень.

Внутрішня будова систем характеризують за допомогою структур, що описують стійкі зв'язки між їх елементами.

При описі АС використовують такі види структур, що відрізняються типами елементів і зв'язків між ними:

– функціональні (елементи – функції, завдання, процедури; зв'язку – інформаційні);

– технічні (елементи – пристрої, компоненти і комплекси; зв'язку – лінії і канали зв'язку);

– організаційні (елементи – колективи людей і окремі виконавці; зв'язку – інформаційні, супідрядності і взаємодії);

– документальні (елементи – неподільні складові частини і документи АС; зв'язку – взаємодії, входимо і підпорядкування);

– алгоритмічні (елементи – алгоритми; зв'язку – інформаційні);

– програмні (елементи – програмні модулі і вироби; зв'язку - керуючі);

– інформаційні (елементи – форми існування і подання інформації в системі, зв'язку – операції перетворення інформації в системі) [7].

На рисунку 1.3 зображена структурна схема програмного засобу [9].

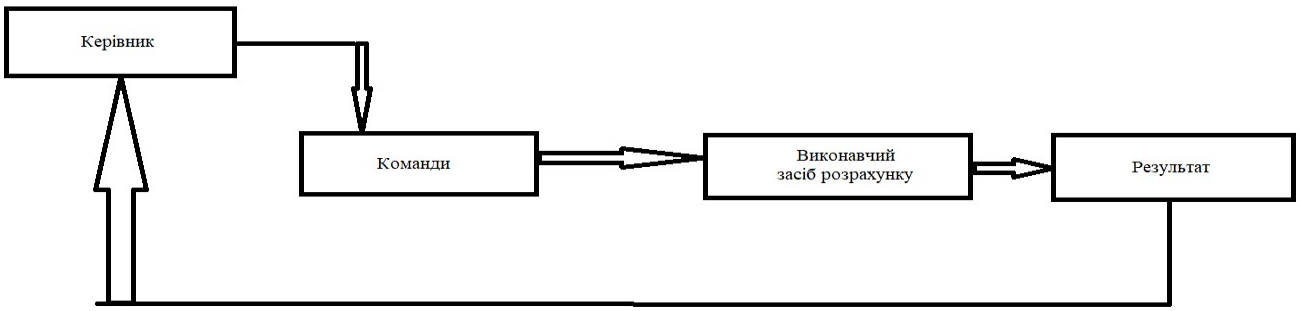

Рисунок 1.3 – Структурна схема програмного засобу

1.3 Аналіз бально-рейтингової системи

Управління знаннями, вміннями та навичками учнів є одним із найважливіших елементів освітнього процесу. Ефективність управління освітнім процесом великою мірою залежить з його правильної організації.

Контролі є набори дій, які визначають якісні та кількісні характеристики результатів навчання та дозволяють оцінити, наскільки учні засвоюють навчальні матеріали [12].

Контроль та оцінка мають бути спрямовані на виявлення рівнів знань учнів. Тільки в цьому випадку випускники можуть досягти успіху та набути певних особистісних та поведінкових навичок. Серед них відповідальність, здатність до альтернативного вибору, готовність до активної творчості.

Є три функції управління:

– діагностична – призначена для виявлення та оцінки цікавих для нас якостей учня, а також рівня знань, умінь та навичок учня;

– виховна – спрямована на оптимізацію самостійної роботи учня та активізацію пізнавальної діяльності при підготовці до поточного заняття та під час проведення заліків, рефератів та заліково-екзаменаційної підготовки;

– виховна функція полягає у формуванні у студентів відповідального та творчого ставлення до наукової сфери та через них любові до професії у розвитку кожного культу навчання [12].

Система управління включає різні форми, такі як іспити, тести, усні іспити, управлінські документи і звіти про виробничу практику. Вибір форми контролю залежить від мети, змісту, методу, часу та місця. Усне дослідження, наприклад, виявляє як знання, а й володіння усною мовою, що допомагає виправляти мовні помилки. Лист дає можливість оцінити глибину та силу навчання.

В даний час система управління знаннями в освітніх організаціях середньої професійної освіти суперечить сучасним вимогам щодо підготовки кваліфікованих фахівців. Головний його недолік очевидний – він не сприяє активній, ритмічній та

самостійній роботі студента. Можна сміливо сказати, що проблема застосування оцінювальної системи контролю знань учнів давно назріла.

Оцінювання учнів є як методом позиціонування учнів за вимірними освітніми результатами, і науково обґрунтованою формою, яка як управляє знаннями, так і організує весь освітній процес.

Система оцінювання являє собою сукупність правил, методичних вказівок та відповідного математичного апарату, реалізованих у програмному комплексі, що обробляє інформацію як за кількісними, так і за якісними показниками індивідуальної навчальної діяльності учня, гарантує та дозволяє виставляти індивідуальні оцінки (загальні оцінки, номери) для кожного студента за розділами всіх навчальних дисциплін, всіх видів професій, а також є узагальненою для багатьох дисциплін [12].

Оцінка – це сума балів, набраних учням за певний період часу, що розраховується за певною формулою, яка не змінюється протягом цього періоду [12].

Основними завданнями бальної рейтингової системи управління знаннями є:

– підвищити мотивацію студентів до якісного освоєння освітніх програм за рахунок більш диференційованої оцінки навчальної діяльності студентів;

– стимулювати студентів до регулярного виконання самостійної роботи шляхом запровадження диференційованого підходу до оцінки знань студентів та результатів навчальної діяльності;

– упорядкувати систему контролю знань учнів та розробити єдині вимоги до оцінки знань у межах окремих дисциплін та цілих освітніх програм;

– підвищити рівень пізнавальних процесів.

Загалом оцінки поділяються на різні види та диктують порядок вивчення дисципліни та її володіння, у тому числі:

– оцінка за місцем з урахуванням поточної роботи студента та результатів на іспитах;

– загальна оцінка, що відображає успіхи учня з усіх предметів, що вивчаються у конкретному семестрі;

– загальна оцінка за період навчання, що відображає загальний успіх учня за період навчання.

Основна проблема при впровадженні системи управління оцінюванням полягає в тому, що вона значно збільшує час, який витрачають вчителі на підготовку уроків та додаткових занять. Однак із досвідом гострота проблеми зменшується.

2 ФОРМАЛІЗАЦІЯ ПАРАМЕТРІВ АВТОМАТИЗОВАНОЇ СИСТЕМИ РОЗРАХУНКУ РЕЙТИНГОВОЇ ОЦІНКИ СТУДЕНТА

2.1 Формальний опис основних параметрів системи розрахунку рейтингової оцінки студента

Відповідно до Положення про організацію освітнього процесу в ХНУРЕ, форми контрольних заходів з навчальних дисциплін відображено в освітній програмі та навчальному плані. Інструментом контрольних заходів є рейтингове оцінювання успішності навчання здобувачів вищої освіти. Рейтинг є інтегральною оцінкою результатів усіх видів навчальної діяльності студента під час опанування ними освітньої програми підготовки.

Основним завданням оцінки є підвищення мотивації абітурієнтів до активного навчання, систематичної самостійної роботи протягом семестру та відповідальності за результати своєї освітньої діяльності, а також постійний зворотний зв'язок із кожним абітурієнтом та своєчасне коригування, його учнівська діяльність, об'єктивна оцінка рівня підготовки тощо.

Оцінка абітурієнтами вищої освіти за їх навчальними дисциплінами вимірюється зі 100 і далі конвертується в оцінки за національною шкалою та шкалою ECTS.

Результати іспитів та оцінок зазвичай виражаються в оцінках. Однак у Європі співіснує безліч різних систем оцінок. Крім того, питання перерахунку оцінок було однією з найважливіших проблем студентів – учасників ECTS:

– з одного боку, інтерпретація оцінок сильно відрізняється від країни до країни, від предмета до предмета та від навчального закладу до навчального закладу;

– з іншого боку, помилки у перерахунку оцінок можуть мати серйозні наслідки для учнів.

В результаті Єврокомісія скликала експертну групу для вирішення спірного питання. Для розробки запропонованої шкали ECTS було розглянуто інформацію, коментарі та статистичні дані, представлені 80 із 84 установ, що беруть участь у ECTS на той момент. Усі групи піддослідних були зіставлені за рейтинговою шкалою ECTS для підтвердження її ефективності.

Шкала оцінок ECTS була розроблена для того, щоб освітні установи могли переносити оцінки, виставлені місцевими установами. Він надає додаткову інформацію про роботу учнів та не замінює загальних оцінок. Вищі навчальні заклади ухвалюють власні рішення щодо використання рейтингових шкал у своїх системах.

Європейська система спрощених рейтингових шкал.

З початку дослідницької фази багатосторонні обговорення між п'ятьма групами показали, що спрощена система оцінювання може ефективно передавати оцінки і добре розуміється у Європі. Концепція спрощеної рейтингової шкали означала, що шкала була добре охарактеризована і тепер могла використовуватись будь-яким навчальним закладом для своїх предметів.

– таким чином, шкала ECTS доповнювала оцінку агентства, але не замінювала її;

– рейтингова шкала ECTS була зрозуміла іншим вузам, які присвоювали відповідні рейтинги студентам або випускникам, що надходять, за власною рейтинговою системою;

– оцінка за системою ECTS визначає досягнення кожного студента до та після періоду навчання, поряд із оцінкою, представленою навчальним закладом у предметному списку оцінки студента.

Інакше кажучи, спрощені рейтингові шкали забезпечують ясність, але з заважають нормальному процесу оцінювання у кожному установі.

Під час обговорення рейтингової шкали ECTS особливу увагу було приділено строгому числовому визначенню, заснованому на класифікації учнів відповідно до їх класних знань, та якіснішому визначенню, заснованому на загальному розумінні ключових слів. «добре», «відмінно» тощо. Жоден з підходів

не дав задовільних результатів. Крім того, суворий числовий підхід визначив би нелогічні межі з погляду міжнародної рейтингової системи. Тому спостерігалася різниця у мірі розуміння ключових слів залежно від установи.

Оцінка в балах	Оцінка за національною шкалою	Оцінка за шкалою ECTS	
		Оцінка	Пояснення
90-100	Відмінно	A	Відмінно (відмінне виконання лише з незначною кількістю помилок)
82-89	Добре	B	Дуже добре (вище середнього рівня з кількома помилками)
75-81		C	Добре (в цілому правильне виконання з певною кількістю суттєвих помилок)
67-74	Задовільно	D	Задовільно (непогано, але зі значною кількістю недоліків)
60-66		E	Достатньо (виконання задовольняє мінімальним критеріям)
35-59	Незадовільно	FX	Незадовільно (з можливістю повторного складання)
1-34		F	Незадовільно (з обов'язковим повторним курсом)

Рисунок 2.1 – Таблиця оцінок за шкалою ECTS

Кількість оцінок за рейтинговою шкалою ECTS є компромісною. Низький бал дає надто мало інформації. Високі бали мають на увазі конкретні пояснення, яких не існує, що збільшує механічну роботу з підрахунку балів. Було вибрано п'ять визначень прохідного балу, щоб максимізувати цінність оцінок A та E.

Подвійне використання терміну «відмінно» та статистичного виразу «10 відсотків кращих учнів» є двома підходами до загальної мети. Шкала накладає своє визначення рейтинг A за шкалою ECTS, а чи не нав'язує визначення переваги кожному установі. Є багато моментів у виборі числа 10%. У деяких навчальних

закладах важко дати суворіше визначення, але загальне визначення знижує стимули для більш здібних студентів.

Крім того, шкала оцінок ECTS заснована на визначенні переваги, а не на припущеннях щодо розподілу оцінок учнів. Визначення переваги та підсумкової оцінки в системі ECTS покликане полегшити повторне зарахування, але не замінювати та не плутати оцінки, виставлені в установі, в якій навчається студент.

Хоча акцент робиться на відмінних оцінках, виявлення більш низьких оцінок відіграє важливу роль для багатьох учнів, тому шкала ECTS повинна застосовуватися на всіх рівнях успішності.

Неможливо визначити єдиний стандарт для рейтингових систем у європейських країнах. У більшості країн існує універсальна рейтингова система, але вона ні в якому разі не є універсальною. Крім того, визначення прохідного бала за певною шкалою може змінюватись в залежності від навчального закладу, а порогові значення для застосування всіх доступних балів сильно різняться між установами, від року до року та від предмета до предмета.

Один із основних принципів рейтингової шкали ECTS полягає в тому, що вона має бути визначена досить добре, щоб установи могли самостійно приймати рішення щодо використання шкали.

Ось як співвідносити бали агентства з рейтинговою шкалою системи ECTS:

Навчальний заклад перевіряє розподіл балів, що присуджуються студентам. Для отримання 10-25-30-25-10 балів за моделлю кордону між оцінками відповідають 10%, 35%, 65% та 90% від загальної кількості успішних студентів.

Неможливо провести статистичні обмеження для оцінок, отриманих 10% найкращих учнів, та необхідно враховувати ключові слова, а також статистику. Заходи мають сильну статистичну підтримку, але статистика має поєднуватися з реалістичним описовим підходом. Наприклад, британський навчальний заклад, який нагороджує 8% своїх студентів першою оцінкою при присвоєнні ECTS оцінки A, може віддати перевагу збереженню того ж визначення вищої оцінки в рейтингу ECTC, італійські навчальні заклади присуджують 30. % до 14% учнів не мають засобів розрізнення цих учнів . З іншого боку, іспанські навчальні заклади, які

присуджують Matricula de Honor менш ніж 5% своїх студентів, були б надто вузьким визначенням «вищої оцінки» для цілей системи ECTS.

Невелика різниця в балах, виставлених Іспанією, Нідерландами і, можливо, Грецією, може ускладнити визначення меж системи ECTS. Яскравим прикладом є те, що 70% студентів групи отримали 7 балів в опублікованому рекорді датського навчального закладу, легко перекриваючи оцінки C та D за системою ECTS. Навіть якщо екзаменатор ставить більш високу оцінку, наприклад, 6,8 або 7,2, вам зазвичай потрібна 7. Таким чином, екзаменаційні бали в цих країнах можуть використовуватися для забезпечення реалістичного розподілу учнів за оцінками в системі ECTS.

Розподіл балів важливий при виставленні оцінок. Розподіл балів за курсом може змінюватися рік у рік, і можуть бути відмінності у кількісному та якісному обсязі занять. Чим ближче навчальний заклад знаходиться до єдиного критерію між своїми балами та системою оцінок ECTS, тим простіше буде процес виставлення оцінок, але ця простота дій є критичним та систематичним відходом від визначення шкали оцінок, відхилення мають бути врівноважені. Іншими словами, я проти несправедливості стосовно студентів.

Якщо кількість учнів, які відвідують клас, дуже мала, то чіткий розподіл цього невеликого числа за моделлю 10-25-30-25-10 не підходить. Однак досвід показує, що:

– Зазвичай розподіляються бали за кілька занять одного рівня.

- Розподіл балів за 5 років із більшою ймовірністю призведе до збалансованих результатів.

Інформація, яку надає рейтингова система ECTS, співвідносить роботу одного студента з роботою інших студентів у групі. Зрозуміло, що учні з високим рівнем групи з низьким рівнем отримують помітно вищі оцінки, ніж вони виконували спільну роботу. Так само, як студенти, які вивчають описові курси, перебувають у невигідному становищі у зарубіжних навчальних закладах, де наголошується на математичних навичках. Немає рейтингової шкали, яка вирішує цю проблему. Інформація, яка передається до рейтингового списку доменів,

повинна відображати те, що сталося насправді, а не те, що могло статися. Якщо оцінки відображаються у списку оцінок дисципліни, оцінки повинні бути індикатором кредиту для окремого класу.

Тому важливо не припускати, що розподіл середньорічних балів є адекватним визначення цих оцінок, оскільки розподіл середньорічних балів значно відрізняється від розподілу групових предметних балів, що становлять середнє значення. Наприклад, більше студентів отримують певний дуже високий бал за одну сесію, ніж у середньому за всі сесії протягом року. Це може вплинути на визначення А-балла системи ECTS і меншою мірою на В-балла.

Оцінки за системою ECTS від A до E надаються за умови проходження тесту, а оцінки від FX до F у разі його неуспішного проходження. Відмінності між FX та F допомагають визначити майбутні навчальні програми для менш успішних учнів. Установи, які можуть розрізняти рівні поганої роботи, ігнорують оцінки FX і застосовують лише оцінки F.

Оскільки місцеві та закордонні установи вирішують, як бали відповідають оцінкам у системі ECTS, оцінки перераховуються таким чином:

– Італійський студент склав іспит у французькому навчальному закладі та набрав 13 балів із 20. У цій установі 13 балів означають хорошу роботу та відповідають оцінці С за системою ECTS. Список оцінок з дисципліни виражає наявність 13 балів та оцінки «С» з ECTS. На підставі цієї інформації місцева влада Італії присвоює 27 балів із 3.

– Студент з Німеччини не показав успіхів при складанні іспиту з однієї з іспанських дисциплін, а у списку оцінок з дисципліни зазначено наявність 5 балів з 10 можливих, а за системою ECTS оцінка Є. Німецькі навчальні заклади визнають бали з оцінкою 4,0 за шкалою від 1,0 (дуже добре) до 5,0 (погано).

– Голландський португальський учень набрав 9 балів із 10 і був одним із найздібніших учнів у 10% найкращих учнів класу. Отримала відмінні оцінки, у списку оцінок на предмет зазначено кількість нарахованих балів та оцінка А за системою ECTC. Місцева влада Португалії використовує цю інформацію для присвоєння 19 балів із 20.

Як видно з вищевикладеного, вищі навчальні заклади є вільними у застосуванні рейтингової шкали ECTS найбільш відповідним чином. Тим не менш, шкала оцінок ECTS призначена для відображення різноманітності систем оцінок, що існують у країнах ЄС та ЄАВТ, і не може охоплювати всі можливі варіанти оцінок та розподілу окремо, що забезпечує певну гнучкість.

В основі системи оцінки успішності студентів ВНЗ лежать поточний контроль та семестровий контроль, системи, які накопичують оціночні бали студентів ВНЗ у процесі навчання. Поточний контроль здійснюється на різних видах навчальних занять та призначений для перевірки рівня знань студентів вузів у суміжних галузях [13].

Ведення поточного управління успішністю претендента визначається відповідною робочою програмою дисципліни. Система оцінювання успішності вступників включає систему контрольних заходів: індивідуальні семестрові завдання, контрольні роботи, звіти та лабораторні захисту, поточний контроль на практичних заняттях, комп'ютерні тести та ін. Ще одним засобом об'єктивної оцінки якості знань, компетенцій і навичок, набутих у процесі навчання , є організація самостійної роботи студентів вищих навчальних закладів Для управління самостійною роботою використовуються такі види оцінки: Контрольні завдання для практичних та лабораторних занять. контрольні роботи, тестування чи інше керування темами (модулями), поданими на самостійне дослідження; Відповіді питання, поточне управління засвоєнням матеріалів практичних занять з урахуванням звітів. Після побудови системи контрольних заходів визначаються максимальний та мінімальний бал за кожним контрольним показником з урахуванням того, що кандидат засвоїв певний рівень знань. У таблиці 2.1 наведено приклад кількісних показників дисципліни «Проектування систем автоматизації» (PSA).

Таблиця 2.2 – Кількісні критерії оцінювання [4]

Вид заняття / контрольний захід	Оцінка
Лб № 1, 2	(6…10)·2 = 12…20
Контрольна робота №1	12…30
Контрольна точка 1	24…50
Лб № 3, 4, 5	(6…10) 3 = 18…30
Контрольна робота №2	12…20
Контрольна точка 2	24…50
Всього за семестр	48…100

Виконання індивідуального навчального плану з кожної дисципліни відображається в електронному журналі (у відсотках) на визначену дату, як правило, один раз на семестр. Результати виконання навчального плану відображаються в індивідуальному навчальному плані здобувачів вищої освіти щосеместрово, а також у навчальній картці студента.

Загальна успішність студента визначається за оцінювальною шкалою ECTS, яка ранжує студентів на статистичній основі. Розподіл оцінок між студентами, які отримали за курс оцінку вище незадовільної, виглядає таким чином:

– A – кращі 10 %;

– B – наступні за ними 25 %;

– C – наступні за ними 30 %;

– D – наступні за ними 25 %;

– E – наступні за ними 10 %.

Для неуспішних студентів ставляться оцінки FX і F. Між ними існує різниця, яка полягає в тому, що FX означає – «не виконав якусь частину роботи, необхідну для здобуття оцінки вище незадовільної», а F – «не виконав всю необхідну роботу». Включення оцінок FX і F в розшифровку оцінок не є обов’язковим [13].

Облік загальної успішності студентів університету потребує правильної організації збереження та відтворення даних. Враховуючи великий об’єм

відомостей щодо окремих студентів, семестрів та дисциплін, що постійно оновлюється та може потребувати редагування, необхідним є створення бази даних з успішності студентів університету. При цьому треба передбачити, що дані можуть вноситися різними людьми, а потім об'єднуватися в одну загальну базу даних. Наприклад, таблицю про студентів заповнюють на кафедрі, на якій навчається студент, таблицю про викладача та предмети - на кафедрі цих викладачів, а оцінки ставить викладач. Побудувати базу даних, яка була б максимально гнучкою. У базі даних повинні міститися дані про студентів, оцінках, викладачів і предметах.

База даних вирішує наступні завдання:

– зберігання даних про студентів та їх успішності;

– додавання нових відомостей про успішність студентів, редагування відомостей про них;

– пошук за різними критеріями.

2.2 Формалізація критеріїв оцінювання успішності студентів

Якщо формою підсумкового контролю для дисципліни є комбінований іспит, то підсумкова оцінка $Р_п$ обчислюється за формулою [13]:

$$P_n = 0{,}6 \cdot O_{сем} + O_{icп}, \tag{2.1}$$

де $O_{сем}$ – оцінка за семестр у 100-бальній системі;

$О_{icп}$– оцінка за іспит у 100-бальній системі.

Білет для іспиту складається з теоретичного запитань та практичного завдання, яке необхідно виконувати на комп’ютери. Теоретичне запитання оцінюються в 30 балів, а практичне завдання – у 40 балів (в сумі – 100 балів).

Для дисциплін, формою підсумкового контролю яких є залік, використовується формула для обчислення підсумкової оцінки

$$P_n = 0{,}6 \cdot O_{сем} + 0{,}4 \cdot O_{зал}, \qquad (2.2)$$

де $O_{сем}$– оцінка за семестр у 100-бальній системі;

$O_{зал}$– оцінка за залік у 100-бальній системі. Запитання для заліку складаються з теоретичних запитань. Теоретичні запитання оцінюються в 50 балів (в сумі – 100 балів).

За результатами отриманих підсумкових оцінок успішність студента повинна бути також відображена за допомогою оцінювальної шкали ECTS, а також за національною шкалою. У таблиці 2.2 наведена відповідність між використовуваними шкалами оцінювання.

За результатами навчання в університеті студентам денної форми навчання може бути призначена академічна стипендія згідно з «Положенням про стипендіальне забезпечення у Харківському національному університеті радіоелектроніки».

Призначення академічної стипендії здійснюється за відповідними рейтингами сформованими за середнім балом підсумкового контролю за відповідний семестр зазначеного навчального року.

Таблиця 2.3 – Відповідність між використовуваними шкалами оцінювання [14]

<table>
<tr><td rowspan="2">Сума балів за всі види навчальної діяльності</td><td rowspan="2">Оцінка ECTS</td><td colspan="2">Оцінка за національною шкалою</td></tr>
<tr><td>для екзамену, курсового проекту (роботи), практики</td><td>для заліку</td></tr>
<tr><td>96-100</td><td>A</td><td rowspan="2">відмінно</td><td rowspan="4">зараховано</td></tr>
<tr><td>90-95</td><td>B</td></tr>
<tr><td>75-89</td><td>C</td><td>добре</td></tr>
<tr><td>66-74</td><td>D</td><td rowspan="2">задовільно</td></tr>
<tr><td>60-65</td><td>E</td></tr>
<tr><td>35-59</td><td>FX</td><td>незадовільно з можливістю повторного складання</td><td>не зараховано з можливістю повторного складання</td></tr>
<tr><td>1-34</td><td>F</td><td>незадовільно з обов’язковим повторним вивченням дисципліни</td><td>не зараховано з обов’язковим повторним вивченням дисципліни</td></tr>
</table>

Рейтинг призначення академічної стипендії розраховується за наступними формулами:

$$R = 0{,}9 \cdot S + 0{,}1 \cdot H, \tag{2.3}$$

де R – загальне значення Рейтингу студента $0 \leq R \leq 100$, розраховується до тисячних знаків (округлення результату – до третього знаку після коми);

S – складова усішності $0 \leq S \leq 100$;

H – складова участі у науковій, науково-технічній діяльності, громадському житті та спортивній діяльності (активність студента); 0 ≤ H ≤ 100.

Складова успішності визначається за формулою:

$$S = \frac{\sum_{i=1}^{n} a_i \cdot b_i}{\sum_{i=1}^{n} b_i}, \quad (2.4)$$

де a_i – підсумкова оцінка з і дисципліни (у 100-бальній шкалі),

b_i – кількість кредитів ECTS з і дисципліни;

n – кількість дисциплін у семестрі.

Участь у науковій, науково-технічній діяльності, громадському житті та спортивній діяльності визначається за формулою:

$$H = z \cdot F, \quad (2.5)$$

де *z* – коефіцієнт нормування для переведення результату показника активності студента у діапазоні 0 ≤ H ≤ 100; z = 2,5

F – відносний показник кількості балів активності студента (визначається у діапазоні 0 ≤ F ≤ 40), що розраховується за формулою:

$$F = (V - t), \quad (2.6)$$

де *t* – технічна константа, $t = 60$;

V – показник кількості балів активності студента (визначається у діапазоні 60 ≤ V ≤ 100), який розраховується за формулою:

$$V = (t + G), \quad (2.7)$$

де *t* – технічна константа, $t = 60$;

G – абсолютне значення сумарної кількості балів за наукову, науково-технічну діяльність, участь у громадському житті та спортивну діяльність студента за семестр (визначається у діапазоні $0 \leq G \leq 40$); показник G складається із суми балів, що нараховуються за науково-технічну діяльність, участь у громадському житті та спортивну діяльність студента за семестр [13].

До рейтингу призначення академічної стипендії не включаються особи, які:

– протягом навчального семестру до початку поточного семестрового контролю з будь-якого навчального предмета (дисципліни) набрали меншу кількість балів, ніж визначена в університеті. Рішенням ректора університету таким особам може встановлюватися строк, протягом якого вони можуть покращити результати навчання, але не більш як до дати початку наступного навчального семестру згідно з навчальним планом за відповідною спеціальністю (напрямом підготовки). У разі коли у визначений строк академічна заборгованість не ліквідована, здобувач вищої освіти підлягає відрахуванню з числа осіб, які навчаються за державним замовленням;

– станом на перше число місяця, що настає після закінчення семестрового контролю згідно з навчальним планом, мають академічну заборгованість;

– під час семестрового контролю здійснювали повторне складення контрольних заходів з метою покращення отриманих раніше оцінок;

– до дати завершення семестрового контролю, визначеного навчальним планом, не склали семестровий контроль з будь-якого навчального предмета (дисципліни).

2.3 Аналіз та вибір середовища розробки автоматизованої системи розрахунку

Visual Studio Code – це редактор вихідного коду, розроблений Microsoft для Windows, Linux та MacOS. Позиціонується як «легковажний» редактор коду для кросплатформної розробки веб-додатків та хмарних додатків. Включає відладчик, інструменти Git, підсвічування синтаксису, IntelliSense та інструменти

рефакторингу. Широкі можливості налаштування включають теми, сполучення клавіш, файли конфігурації та багато іншого. Він розробляється і розповсюджується безкоштовно як програмне забезпечення з відкритим вихідним кодом, але готові зборки поширюються під проприетарною ліцензією.

Код Visual Studio заснований на Electron та реалізований через веб-редактор Monaco, розроблений для Visual Studio Online [14].

Він підтримує десятки мов програмування, підсвічування синтаксису, IntelliSense, рефакторинг, налагодження, навігацію за кодом, підтримку Git та багато іншого. Багато функцій Visual Studio Code недоступні через графічний інтерфейс і найчастіше доступні через палітру команд або файли JSON (наприклад, налаштування користувача). Палітра команд є свого роду командний рядок, що викликається комбінаціями клавіш.

Visual Studio також може замінити кодову сторінку, символ перетворення рядка та мову програмування поточного документа під час збереження документа.

Станом на березень 2019 року ви можете завантажувати та встановлювати тисячі розширень тільки в категорії «Мови програмування», використовуючи вбудований інтерфейс користувача продукту.

Тепер подумайте про PhpStorm. Це середовище є інтелектуальним редактором для PHP, HTML та JavaScript з аналізом коду на льоту, запобіганням помилок у коді та інструментами автоматичного рефакторингу для PHP та JavaScript. Автозаповнення коду PhpStorm підтримує специфікації PHP 5.3, 5.4, 5.5, 5.6, 7.0, 7.1, 7.2, 7.4 та 8.0 [12]. Він має повноцінний редактор SQL з можливістю редагування результатів запиту.

PhpStorm розроблено на основі платформи IntelliJ IDEA, написаної на Java. Користувачі можуть розширити функціональність свого середовища розробки, встановивши плагіни, розроблені для платформи IntelliJ, або створивши власні плагіни.

Його переваги:

– підтримка SQL та бази даних (рефакторинг схеми бази даних, створення сценаріїв міграції схеми, експорт результатів виконання запитів у файл або буфер обміну, редагування збережених процедур тощо);

– віддалене розгортання програм та автоматична синхронізація з використанням FTP, SFTP, FTPS тощо протокол;

– інтеграція із системами контролю версій (Git - включаючи спеціальні можливості для роботи з GitHub, Subversion, Mercurial, Perforce, CVS, TFS). Це дозволяє виконувати безліч дій, таких як фіксація, злиття, порівняння і т. д., прямо з PhpStorm.

– локальна історія (Local History) (локально відстежує зміни коду);

– PHP UML (діаграма класів UML коду PHP з рефакторингом, що викликається безпосередньо з діаграми);

– підтримка Phing (забезпечує автодоповнення, перевірку стандартних тегів, властивостей, цільових імен, значень атрибутів шляху у файлах збирання);

– інтеграція з баг-трекерами;

– підтримка Docker, Vagrant, консолі SSH та віддалених інструментів.

– підтримка Google App Engine для PHP.

– використання різноманітних комбінацій клавіш для підвищення ефективності [16].

Тепер розглянемо високе текстове середовище. Sublime Text – текстовий редактор. Підтримує плагіни для програмування Python.

Розробники можуть ознайомитися з продуктом безкоштовно та без обмежень, але програма нагадує їм про необхідність придбання ліцензії.

Sublime Text підтримує безліч мов програмування, C, C++, C#, CSS, D, Dylan, Erlang, HTML, Groovy, Haskell, Java, JavaScript, LaTeX, Lisp, Lua, Markdown, MATLAB, OCaml, Perl, PHP, Python, R ., Рубі, SQL, TCL, XML.

На додаток до мов програмування, включених за замовчуванням, користувачі мають можливість завантажити плагіни для підтримки інших мов.

Sublime Text може бути оснащений менеджером пакетів, який дозволяє користувачам знаходити, встановлювати, оновлювати та видаляти пакети без

перезапуску програми. Менеджер підтримує встановлені пакети у актуальному стані, завантажуючи нові версії з репозиторію. Крім того, він надає команди для активації та деактивації встановлених пакетів[16].

Переваги піднесеного тексту.

По-перше, це інтерфейс. Редактор включає різні візуальні теми, а також можна завантажити додаткові теми.

Весь код користувач бачить у правій частині екрана у вигляді клікабельної мінікарти.

Існує кілька режимів екрана. Один із них містить від 1 до 4 панелей і може одночасно відображати до 4 файлів. У повному (вільному режимі) режимі відображається лише один файл без додаткових меню.

Другий – вибір стовпця та множинне редагування. Виділіть весь стовпець або помістіть кілька вказівників на текст для швидкого редагування.Індекси поводяться так, ніби кожен із них був одиницею в тексті. Такі команди, як перейти до символу, перейти до рядка, виділити текст, перейти до слова або його частини (верблюжий регістр, розділений дефісом або символом підкреслення), перейти до початку/кінця рядка і т.д. Впливають на них одночасно. Швидко керуйте складним текстом без використання макросів або регулярних виразів.

По-третє, автозаповнення. Коли ви вводите свій код, Sublime Text пропонує різні варіанти завершення введення в залежності від мови, яку ви використовуєте. Редактор також автоматично доповнює створені користувачем змінні.

По-четверте, підсвічування синтаксису та висока контрастність. Темний фон Sublime Text призначений підвищення контрастності тексту. Основні синтаксичні елементи виділені різними кольорами, які краще працюють із темним тлом, ніж зі світлим тлом.

По-п'яте, підтримка системи збирання. Sublime Text дозволяє користувачам компілювати та запускати програми, не перемикаючись на командний рядок. Користувачі також можуть налаштувати систему збирання, щоб увімкнути автоматичне збирання програми при кожному збереженні коду.

Шосте – заготівлі (фрагменти). Зберігайте часто використовувані фрагменти коду та ключові слова для їх запуску.

Сьоме – навігація між файлами. Інструмент навігації, який дозволяє користувачам переміщатися між файлами та всередині них за допомогою нечіткого пошуку.

Тепер давайте подивимося на NetBeans.

IDE NetBeans – це безкоштовне інтегроване середовище розробки додатків (IDE) для мов програмування, як Java, Python, PHP, JavaScript, C, C++.

Проект NetBeans IDE підтримується та спонсорується Oracle, але NetBeans розробляється незалежною спільнотою захоплених розробників (Спільнота NetBeans) та NetBeans Org.

Остання версія IDE NetBeans підтримує рефакторинг, профільування, виділення кольором синтаксичних конструкцій, автоматичне завершення типізованих конструкцій на льоту та безліч зумовлених шаблонів коду.

Для розробки програм у середовищі NetBeans, а також для успішного встановлення та запуску середовища NetBeans у вас повинна бути попередньо встановлена правильна версія Sun JDK або J2EE SDK. Середовище розробки NetBeans за промовчанням підтримує розробку для платформ J2SE та J2EE. Починаючи з версії 6.0, NetBeans підтримує мобільну платформу J2ME, C++ (тільки g++) та розробку PHP без встановлення будь-яких додаткових компонентів.

У середовищі IDE NetBeans версії 6.1 заявлено підтримку UML, SOA, мови програмування Ruby (включаючи підтримку Ruby on Rails) та інструментів для створення програм J2ME для мобільних телефонів. Підтримка мови PHP була додана у версії 6.5. Модуль підтримки Python [14] також надається для тестування.

Середовище IDE NetBeans підтримує модулі, тому розробники можуть розширювати функціональні можливості середовища. Одним із найпопулярніших плагінів є потужний конструктор звітів iReport [8] (на основі бібліотеки JasperReports).

Комерційні інтегровані середовища розробки Sun для Java – Sun Java Studio Creator, Sun Java Studio Enterprise та Oracle Solaris Studio (для розробки на C, C++

або Fortran) використовуються для розробки ідей, технологій та значною мірою засновані на . Вихідний код середовища IDE NetBeans. Нещодавно Sun почала безкоштовно пропонувати ці середовища розробки розробникам, зареєстрованим у Sun Developer Network (SDN). Реєстрація на самому сайті безкоштовна та не вимагає жодних попередніх умов, крім прийняття ліцензії CDDL.

Середовище IDE NetBeans доступне у вигляді готових дистрибутивів (скомпільованих двійкових файлів) для платформ Microsoft Windows, Linux, FreeBSD, Mac OS X, OpenSolaris та Solaris (як SPARC, так і x86 – Intel та AMD). Всі інші платформи можуть компілювати NetBeans незалежно від вихідний текст.

У випуск NetBeans IDE 6.7 додана інтеграція з Project Kenai [9], підтримка мови Groovy та веб-інфраструктура Grails. Версія 6.8 підтримує фреймворк Symfony PHP, а 6.9 - Zend Framework.

У версії 7 розробники відмовилися від підтримки мови Ruby та Ruby on Rails. Це пов'язано з великим обсягом роботи, виконаної для підтримки Java 7 і користувачів NetBeans (компоненти Ruby і Rails були передані спільноті і продовжуватимуть існувати [18]). Тому остання версія NetBeans «з коробки» підтримує лише Java (включаючи Java FX, Java ME та Java EE), C/C++, Groovy, PHP, HTML, JavaScript та CSS. Крім того, у версії 7 немає пропрієтарних або безкоштовних сторонніх компонентів для підтримки моделювання UML.

Переваги NetBeans:

– підсвічування синтаксису, автодоповнення коду, виділення входжень та помилок.

– налагодження коду xdebug;

– підтримка тестування за допомогою PHPUnit та Selenium.

– підтримка PHP-фреймворку Symfony (версія 6.8 та вище) та Zend Framework (версія 6.8 та вище). 9);

– підтримка PHP 5.3 (починаючи з версії 6.0.8);

– запуск підтримки Git (з версії 7.1).

У результаті аналізу чотирьох середовищ розробки було обрано середовище Visual Studio Code, тому що воно має інтуітивно зрозумілий інтерфейс, великі можливості додатків для поліпшення умов розробки програм та велику базу знань від користувачів.

2.4. Вибір БД

При виборі бази даних розглядалися чотири бази даних.

– База даних Oracle;

– MySQL;

– Microsoft Access;

– ФайрБейс.

Oracle Database – об'єктно-реляційна СУБД (система управління базами даних), створена Oracle. В даний час існує безліч різних версій та типів. Однак у цій статті обговорюватиметься не тип бази даних Oracle, а скоріше структура та основні поняття, що належать СУБД Oracle. Розуміння архітектури СУБД Oracle забезпечує необхідну основу розуміння інших функцій, наданих базою даних Oracle (і ці функції дуже великі).

СУБД Oracle Database містить фізичні та логічні компоненти. На окрему згадку заслуговує концепція екземплярів. Зверніть увагу, що я використовую терміни «база даних» та «примірник» як синоніми. Так, вони взаємопов'язані, але все ж таки різні. База даних у термінології Oracle – це фізичне місце зберігання інформації, а екземпляр – це програмне забезпечення, що працює на сервері, яке забезпечує доступ до інформації, що міститься в базі даних Oracle. Екземпляр запускається на певному сервері або комп'ютері, а база даних зберігається на диску, підключеному до цього сервера.

База даних Oracle – це фізичний об'єкт, що складається з файлів, що зберігаються на диску. У той же час екземпляр - це логічна сутність, що складається зі структур в оперативній пам'яті та процесів, запущених на сервері. Примірник може бути частиною лише однієї бази даних. Декілька екземплярів можуть бути

пов'язані з однією базою даних одночасно. Примірники мають обмежений термін служби, але бази даних можуть існувати відносно завжди.

Також зверніть увагу, що користувачі не мають прямого доступу до інформації, що зберігається в базах даних Oracle. Користувач повинен запросити цю інформацію у екземпляра Oracle.

Простіше кажучи, екземпляр – це міст до бази даних, а сама база даних – це острів. Під час роботи екземпляра працює міст, і дані можуть входити та виходити з бази даних Oracle. Якщо міст не працює (примірник не працює), користувачі не можуть отримати доступ до бази даних, навіть якщо база даних фізично не втрачена.

Структура бази даних Oracle

Бази даних Oracle включають:

– табличний простір.

– контрольні файли;

– журнал;

– архівні журнали;

– блокувати файли відстеження змін.

– ретроспектива;

– резервні файли (rman).

Табличний простір Оракула. Дані, що зберігаються в базі даних Oracle, повинні знаходитись у будь-якому табличному просторі. Табличне простір сприймається як логічна структура. Інакше кажучи, ви можете попросити ОС показати вам табличні простору Oracle.

У цьому кожен табличний простір містить фізичні структури, звані файлами даних. Один табличний простір Oracle може містити один або кілька файлів даних, але кожен файл даних може належати лише одному табличному простору. При створенні таблиці можна вказати табличний простір, в якому вона буде розміщена. Oracle знаходить таблицю в одному з файлів даних, що становлять вказаний табличний простір.

На наступній діаграмі показано зв'язок між файлами даних та табличними просторами у базі даних Oracle.

При створенні нових таблиць можна помістити їх у табличний простір DATA1 або DATA2. Таким чином, фізично таблиця є одним із файлів даних, що становлять зазначений табличний простір.

Файл бази даних Oracle. База даних Oracle може містити три основні типи фізичних файлів:

– контрольний файл – контрольний файл.

– файл даних – файл даних;

– файл журналу повторів – файл журналу або журнал.

Керуючі файли містять інформацію про розташування інших фізичних файлів, що становлять базу даних Oracle, таких як файли даних та файли журналів. Він також зберігає важливу інформацію про вміст та стан бази даних Oracle. Що це за інформація:

– ім'я бази даних Oracle.

– час створення бази даних;

– імена та розташування файлів журналів та даних.

– інформація про табличні простори.

– інформація про архівні журнали;

– історія журналу, порядковий номер журналу.

– інформація про файли даних в автономному режимі.

– інформація про резервні копії, контрольні точки та копії файлів даних.

У той же час, функція контрольного файлу не обмежується зберіганням критичної інформації, необхідної при запуску екземпляра. Вони також допомагають у процесі видалення бази даних Oracle.

Наприклад, починаючи з Oracle Database 10g, ви можете використовувати команду DROP DATABASE, щоб видалити всі файли, перелічені у файлі бази даних, включаючи сам управляючий файл.

На сьогоднішній день MySQL є однією з найпопулярніших систем керування базами даних в Інтернеті. Ця система використовується для обробки досить

великих обсягів інформації. Тим не менш, MySQL відмінно підходить як для невеликих, так і великих інтернет-проектів.Важливою особливістю системи є те, що вона безкоштовна.

Коли користувач намагається відкрити сторінку сайту (page.php), перед відображенням сайту на серверах хостинг-провайдера відбуваються такі речі:

Код PHP у файлі page.php виконується. Весь текстовий вміст сторінки зчитується з бази даних (database.sql). Стилі обчислюються з файлу стилів (style.css) (що, де, де, шрифт, розмір тощо).

Відображається сторінка, яку хотів побачити користувач.

Також слід розуміти, що користувачі можуть залишати коментарі та додавати пости (статті) на сторінки. До цього моменту всі зміни збережені в базі даних, і при наступному запиті сторінку сторінку вже буде оновлено (оскільки оновлена інформація зчитується з бази даних).

Основна перевага MySQL. Надійність, швидкість та гнучкість – основні характеристики MySQL. Робота з цією системою не викликає особливих проблем, а підтримка MySQL сервера автоматично включена в дистрибутив PHP. MySQL надається відповідно до умов громадської ліцензії GNU (GPL).

У минулому файли використовувалися для безпечного зберігання інформації. Деякі рядки були записані у файл та вилучені для подальшої роботи. Проблема довгострокового зберігання інформації є дуже актуальною для процесу програмування інтернет-додатків. Наприклад, якщо лічильник вважає відвідувачів вашого сайту, або ви зберігаєте повідомлення на форумі, або якщо ваш сайт вимагає віддаленого керування даними, ви змушені використовувати систему довгострокового зберігання інформації.

Однак професійні методи роботи з файлами дуже складні і вимагають багато часу, тому що вам потрібно бути дуже обережним при введенні інформації у файл, сортуванні даних та їх виведенні. Але при цьому слід пам'ятати, що ці дії виконуються на серверах хостинг-провайдера. Ймовірно, у вас встановлено один із варіантів Unix. У зв'язку з цим слід звернути увагу на безпеку доступу до файлів. Це значно збільшує обсяг коду і робить програму більш схильною до помилок.

Зазначене завдання успішно вирішується за рахунок використання бази даних, яка координує безпеку інформації та її сортування, а також пропонує можливість вилучення та впорядкування даних з використанням одного рядка. Код, який використовує базу даних, виглядає компактнішим і його набагато легше налагоджувати. Крім того, не можна забувати про індикатор швидкості. Вибір інформації з бази даних виконується швидше, ніж вибір файлу.

Microsoft Access - це система управління базами даних (СУБД) Microsoft, яка поєднує в собі механізм реляційної бази даних Microsoft Jet з графічним інтерфейсом користувача і інструментами розробки програмного забезпечення. Він є частиною пакету Microsoft Office і входить до складу Професійної та пізнішої версії або продається окремо. Він також є учасником Microsoft 365. Остання версія має найкращий захист. Доступна пробна версія програми.

Microsoft Access зберігає дані у форматі, що базується на ядрі СУБД Access Jet. Ви також можете безпосередньо імпортувати або зв'язувати дані, що зберігаються в інших програмах та базах даних.

Розробники програмного забезпечення, архітектори даних та досвідчені користувачі можуть використовувати Microsoft Access для розробки прикладного програмного забезпечення. Access, як і інші програми Microsoft Office, є Visual Basic для додатків, об'єктно-орієнтована мова програмування, яка може посилатися на безліч об'єктів, включаючи традиційні DAO (об'єкти доступу до даних), об'єктні дані ActiveX та багато інших компонентів ActiveX (VBA) . Візуальні об'єкти, що використовуються у формах та звітах, надають методи та властивості у середовищі програмування VBA. Модуль коду VBA може оголошувати та викликати операції операційної системи Windows. Нема веб-версії.

Крім використання власного сховища бази даних, ви також можете використовувати Microsoft Access як зовнішній інтерфейс програми. З іншого боку, інші продукти, такі як Microsoft SQL Server та інші продукти, такі як Oracle і Sybase, діють як «внутрішні» таблиці. Бази даних Microsoft Access Jet (формати ACCDB та MDB) можуть використовувати декілька внутрішніх джерел. Аналогічно, Visual Basic, ASP. NET чи Visual Studio. NET використовує формат

бази даних Microsoft Access для таблиць та запитів. Microsoft Access також може бути частиною складніших рішень, таких як Microsoft Excel, Microsoft Outlook, Microsoft Word, Microsoft PowerPoint та елементи керування ActiveX.

Таблиці мають доступ до різних типів полів, індексів та цілісності посилань. Це включає каскадні оновлення та видалення. Access також включає інтерфейс запитів, форми для перегляду і введення даних і звіти для друку. Базова база даних Jet, що містить ці об'єкти, є розрахованою на багато користувачів і обробляє блокування запису.

Завдання, що повторюються, можна автоматизувати за допомогою макросів з параметрами «вкажи і клацніть». Бази даних також можна легко об'єднувати через мережу, що дозволяє кільком користувачам обмінюватися даними та оновлювати їх, не перезаписуючи роботу один одного. Дані блокуються лише на рівні записи. Це відрізняється від Excel, який блокує всю електронну таблицю.

Ця програма має шаблон бази даних, який можна завантажити з веб-сайту Microsoft. Ці параметри запускаються під час запуску Access і дозволяють користувачам розширювати базу даних за рахунок визначених таблиць, запитів, форм, звітів та макросів. Шаблони баз даних є кодом бази даних VBA, а шаблони Microsoft не містять коду VBA.

Програмісти можуть використовувати Visual Basic 6.0 (VB6) для створення рішень за допомогою VBA, який використовується у всіх програмах Microsoft Office, таких як Excel, Word, Outlook та PowerPoint. Більшість коду VB6, що використовує Windows API, може бути оголошена у VBA. Досвідчені користувачі та розробники можуть розширити базові рішення для кінцевих користувачів до професійних рішень з розширеною автоматизацією, перевіркою даних, виловом помилок та розрахованою на багато користувачів підтримкою.

Кількість одночасних користувачів, що підтримуються, залежить від обсягу даних, завдань, рівнів і дизайну програми. Загальноприйнятим обмеженням є рішення 1 ГБ або менше даних (доступ підтримує до 2 ГБ) та добре працює при 100 або менш активних підключеннях (підтримка 255 одночасних користувачів). Ця функція найчастіше підходить для відомчих рішень. При використанні рішення

бази даних Access у розрахованому на багато користувачів сценарії вам необхідно «розділити» вашу програму. Це означає, що таблиці знаходяться в одному файлі, який називається бекенд (зазвичай зберігається в спільній папці мережі), а програми (звіти, запити, код, макроси, пов'язані таблиці) – в іншому файлі. називається інтерфейсом. Таблиця асоціацій інтерфейсу показує внутрішні файли. Потім усі користувачі Access одержують копію файлу зовнішнього інтерфейсу.

Користувачі можуть створювати таблиці, запити, форми та звіти та пов'язувати їх за допомогою макросів. Досвідчені користувачі можуть використовувати VBA для створення багатофункціональних рішень з розширеним керуванням даними та користувальницьким контролем. Access має можливості створення звітів, які можуть працювати з іншими даними, до яких Access має доступ.

Початкова концепція Access полягала в тому, щоб дозволити користувачам отримувати доступ до даних із будь-якого джерела. Інші функції включають імпорт та експорт даних у різних форматах, включаючи Excel, Outlook, ASCII, dBase, Paradox, FoxPro, SQL Server та Oracle. У вас є можливість підключатися до існуючих даних про випадки та використовувати їх для перегляду, запиту, редагування та створення звітів. Це дозволяє змінювати наявні дані, щоб Access використовував останні дані. Між наборами даних, що зберігаються на різних платформах, можуть виконуватись різні з'єднання. Доступ часто використовується користувачами, які завантажують дані з баз даних корпоративного рівня для локального управління, аналізу та складання звітів.

Існує також формат бази даних Jet (MDB або ACCDB у Access 2007), який дозволяє зберігати програми та дані в одному файлі. Це робить дуже зручним передачу всієї програми іншому користувачеві, який може запускати його в автономному середовищі.

Однією з переваг Access з погляду програміста є його відносна сумісність із SQL (мовою структурованих запитів). Ви можете переглянути запит графічно або відредагувати його як інструкцію SQL. Оператори SQL можна використовувати безпосередньо в макросах та модулях VBA для керування таблицями Access.

Користувачі можуть використовувати комбінацію VBA та макросів для програмування форм та логіки та забезпечення об'єктно-орієнтованої функціональності. Ви також можете включити VBA до своїх запитів.

Microsoft Access надає параметри запиту. Ці запити та таблиці Access доступні в VB6 та . NET можна використовувати з інших програм, таких як DAO та ADO. З Microsoft Access VBA ви можете посилатися на параметри процедури, що зберігається через ADO.

База даних Firebase Realtime – це хмарна база даних NoSQL Firebase. Дані зберігаються у форматі JSON та синхронізуються в режимі реального часу з кожним підключеним клієнтом. Підтримує інтеграцію з додатками для операційних систем Android та iOS, JavaScript, Java, Objective-C та Node. Реалізує API для js-додатків. Ви також можете безпосередньо працювати з базами даних у стилі REST із багатьох фреймворків JavaScript. AngularJS, React, Vue. js, Ембер. js, магістраль. js увімкнено. Для шифрування даних передбачено API. При створенні кросплатформових програм з використанням SDK для iOS, Android та JavaScript всі клієнти спільно використовують один екземпляр бази даних у режимі реального часу, яка автоматично оновлюється останніми даними.

Ключові особливості включають:

Робота у режимі реального часу. Замість звичайних запитів HTTP база даних Firebase Realtime використовує синхронізацію даних. Щоразу, коли дані змінюються, підключені пристрої отримують оновлення протягом мілісекунд. Це дозволяє вам забезпечити іммерсивне спільне використання, не замислюючись про мережний код працювати офлайн. Пакет Firebase Realtime Database SDK зберігає дані на диск, тому Firebase залишаються чуйними навіть в автономному режимі. Після відновлення з'єднання клієнтський пристрій отримує втрачені зміни шляхом синхронізації із поточним станом сервера.

Доступність із пристроїв клієнтів. Ви можете отримати доступ до бази даних Firebase Realtime безпосередньо зі свого мобільного пристрою або браузера. Сервер програм не потрібний. Безпека та перевірка даних доступні за допомогою

правил безпеки бази даних Firebase Realtime, які базуються на виразах правил, які запускаються під час читання або запису даних.

Масштабування за кількома базами даних. База даних Firebase Realtime дозволяє масштабувати потреби вашої програми даних, поділяючи ваші дані між кількома екземплярами бази даних в одному проекті Firebase. Ви можете використовувати аутентифікацію Firebase у своєму проекті, щоб оптимізувати аутентифікацію та перевіряти справжність користувача у всіх примірниках бази даних. Використовуйте правила користувача бази даних Firebase Realtime для кожного екземпляра бази даних, щоб контролювати доступ до даних у кожній базі даних.

База даних Firebase Realtime Database дозволяє створювати багатофункціональні програми для спільної роботи, забезпечуючи безпечний доступ до бази даних безпосередньо з коду на стороні клієнта.Дані зберігаються локально, а події в реальному часі продовжують відбуватися навіть в автономному режимі, надаючи користувачам повний зворотний зв'язок. Коли пристрій повторно підключається, база даних у реальному часі синхронізує локальні зміни даних з віддаленими оновленнями, які відбулися, коли клієнт був в автономному режимі, автоматично поєднуючи всі зміни.

База даних реального часу є базою даних NoSQL і тому має інші види оптимізації і функціональність в порівнянні з реляційною базою даних. API-інтерфейс бази даних реального часу дозволяє виконувати тільки операції, які можуть бути виконані швидко. Це дозволяє проводити обробку даних в реальному часі, обслуговуючи мільйони користувачів без шкоди для швидкості відгуку. У зв'язку з цим важливо продумати те, як користувачі повинні отримувати доступ до ваших даних, а потім відповідним чином структурувати їх.

Всі дані бази даних Firebase Realtime зберігаються як об'єкти JSON. Ви можете думати про базу даних як про дерево JSON, розміщене в хмарі. На відміну від бази даних SQL, тут немає таблиць чи записів. Коли ви додаєте дані до дерева JSON, воно стає вузлом у існуючій структурі JSON із пов'язаним ключем.

Хоча база даних використовує дерево JSON, дані, що зберігаються в базі даних, можуть бути представлені у вигляді певних власних типів, які відповідають доступним типам JSON, що забезпечує можливість написати код, що підтримується.

База даних Firebase Realtime дозволяє вкладати дані глибиною до 32 рівнів, проте не варто думати, що така структура виглядає за замовчуванням. Таким чином, коли ви вибираєте дані у будь-якому місці у вашій базі даних, ви також отримуєте всі його дочірні вузли. Крім того, коли ви надаєте комусь доступ для читання або запису у вузлі вашої бази даних, ви також надаєте їм доступ до всіх даних цього вузла. Тому на практиці краще зберігати структуру даних якомога плоскішою.

У результаті аналізу чотирьох баз даних була обрана база даних Firebase, тому що вона має інтуітивно зрозумілий інтерфейс, хмарну структуру, керується за допомогою .json файлів через онлайн кабінет на однойменному сервісі та є безкоштовною.

2.5. Вибір середовища сумісної розробки

При виборі сумісних середовищ розробки враховувалися чотири середовища.

– Bitbucket;

– SourceForge;

– RhodeCode;

– Git.

Bitbucket – сервіс розміщення систем контролю версій коду (систем контролю версій, VCS). За допомогою такої системи розробники відстежують зміни коду.

Bitbucket стає все більш популярним. Він, як і його конкуренти, має можливість повернутися до правильної версії коду та виправити будь-які помилки.

Bitbucket є унікальним тим, що його можуть безкоштовно використовувати до 5 розробників. У той же час безкоштовна версія включає доступ до необмеженої кількості приватних репозиторіїв (де дані зберігаються та підтримуються).

Оновлення вашої передплати на послугу збільшує кількість хвилин збору та об'єм сховища для великих файлів, включених до пакета. 3 долари на місяць за стандартний обліковий запис та 6 доларів за преміум-версію.

SourceForge – це інтернет-сервіс, який надає розробникам програмного забезпечення централізований онлайн-репозиторій для моніторингу та управління безкоштовними проектами з відкритим вихідним кодом. Надає репозиторій вихідного коду, відстеження помилок, віддзеркалення завантаження з балансуванням навантаження, вікі для документації, списки розсилки розробників та користувачів, підтримку користувачів форуму, огляди та рейтинги користувачів, інформаційні бюлетені, мікроблоги для оновлень проекту та інші функції.

SourceForge була однією з перших компаній, що запропонували цю послугу безкоштовно проектам із відкритим вихідним кодом. З 2012 року цей сайт працює на програмному забезпеченні Apache Allura. SourceForge надає безкоштовний доступ до хостингу та інструментів для розробників безкоштовного програмного забезпечення з відкритим кодом. Станом на вересень 2020 року в репозиторії SourceForge налічується понад 502 000 проектів та понад 3,7 мільйона зареєстрованих користувачів. SourceForge. com, до серпня 2009 р., sourceforge. net залучив щонайменше 33 мільйони відвідувачів.

SourceForge – це веб-репозиторій вихідного коду. Він є централізованим сховищем безкоштовного програмного забезпечення з відкритим вихідним кодом. Він був першим, хто запропонував цю послугу безкоштовно проектам із відкритим вихідним кодом. Найбільш відомий своїми системами контролю версій, такими як CVS, SVN, Bazaar, Git та Mercurial, він надає розробникам проектів доступ до централізованих репозиторій та інструментів управління проектами. Ключові функції (серед іншого) включають вікі проекту, метрики та аналітику, доступ до бази даних MySQL та унікальні субдомени URL (у формі http://project-name.sourceforge.net).

Величезна кількість користувачів Net (більше 3 мільйонів станом на 2013 рік) може представити чудові проекти різним розробникам та створити цикл позитивного зворотного зв'язку. У міру зростання активності проекту SourceForge. Внутрішня система ранжування мережі робить проект більш помітним для інших розробників через каталоги SourceForge та підприємства. Враховуючи, що багато проектів з відкритим вихідним кодом зазнають невдачі через відсутність підтримки розробників, взаємодія з такою великою спільнотою розробників – це постійний спосіб вдихнути нове життя у ваш проект.

RhodeCode – це корпоративна платформа керування вихідним кодом для репозиторіїв Mercurial, Git та SVN. Він також надає веб-інтерфейси та API для доступу до керування вихідним кодом, користувачам та перевірки коду. Платформа застосовує існуючі інструменти та інтеграцію у вашій кодовій базі за один прохід.

RodKo de написаний на Python з використанням фреймворку Pylons. Він працює як окремий розміщений додаток на виділеному сервері (або в приватній хмарі) і керує кількома репозиторіями у вашій організації. RhodeCode CE безкоштовний з необмеженою кількістю користувачів та репозиторіїв. RhodeCode EE є платним та забезпечує корпоративну інтеграцію поверх CE.

Основні можливості:

а) робота в команді:

- розширений аналіз коду;
- паралельна різниця;
- запит на виведення коштів;
- вбудований чат вихідного коду;
- повнотекстовий пошук за кодом та індексація вихідного коду;
- додайте, редагуйте та видаляйте файли через інтернет.
- система фрагментації коду (pastebin).

б) управління репозиторієм:

- інтегрована підтримка mercurial, git та subversion.

– детальне керування користувачами та інструменти контролю доступу.

– розширена система дозволів з обмеженнями ip.

в) безпека коду та автентифікація:

– система аутентифікації з використанням токенів та підтримка ldap, atlassian crowd, http-headers та pam.

– варіанти корпоративної аутентифікації: active directory, аутентифікація github/google/bitbucket, двофакторна аутентифікація.

– інтеграція зі сторонніми системами відстеження проблем та інструментами ci (jira, redmine, jenkins тощо).)

Платформа RhodeCode доступна у двох версіях:

– RhodeCode CE (Community Edition) є безкоштовним та відкритим вихідним кодом. Ліцензується на умовах ліцензії з відкритим кодом AGPLv3.

– RhodeCode EE (Enterprise Edition) ліцензується на користувача та додає до RhodeCode CE технічну підтримку та корпоративну сертифікацію.

Git – це розподілена система контролю версій. Цей проект був створений Лінусом Торвальдсом для управління розробкою ядра Linux і перша версія була випущена 7 квітня 2005 року. Досі Суміо Хаман підтримує.

Проекти, що використовують Git, включають ядро Linux, Swift, Android, Drupal, Cairo, GNU Core Utilities, Mesa, Wine, Chromium, Compiz Fusion, FlightGear, jQuery, PHP, NASM, MediaWiki, DokuWiki, Qt та багато інших дистрибутивів Linux.

Ця програма безкоштовна та випущена під ліцензією GNU GPL версії 2. За замовчуванням використовується TCP-порт 9418.

Система є набір програм, спеціально розроблених з урахуванням сценарію. Це спрощує створення власної спеціальної системи контролю версій та інтерфейсу користувача на основі Git. Наприклад, Cogito – це саме така оболонка для репозиторію Git, а StGit використовує Git для керування набором виправлень (патчів).

Git підтримує швидке управління версіями та злиття та включає інструменти для візуалізації та навігації з нелінійної історії розробки. Подібно до Darcs, BitKeeper, Mercurial, Bazaar та Monotone[en], Git надає кожному розробнику локальну копію всієї історії розробки, а зміни копіюються з одного репозиторію в інший.

Віддалений доступ до репозиторій Git надається демоном git, SSH або сервером HTTP. Служба git-daemon TCP включена до дистрибутиву Git і, поряд з SSH, є найбільш популярним і надійним методом доступу. Незважаючи на деякі обмеження, метод HTTP дуже популярний у контрольованих мережах, оскільки дозволяє використовувати існуючі конфігурації мережевих фільтрів.

За своєю суттю Git є набір утиліт командного рядка з параметрами. Усі налаштування зберігаються у текстовому конфігураційному файлі. Ця реалізація робить Git легко переносимим на будь-яку платформу і полегшує інтеграцію Git в інші системи (крім іншого, ви можете створювати графічні клієнти git з довільними інтерфейсами).

Репозиторій Git – це каталог файлової системи, що містить репозиторій, що містить файл конфігурації репозиторію, файли журналів, в яких записуються операції, що виконуються в репозиторії, індекс, що описує розташування файлів, і самі файли. Структура зберігання файлів не відображає реальної структури дерева файлів, що зберігається в репозиторії. Це для прискорення виконання операцій над репозиторієм. Коли ядро обробляє команду модифікації (чи локальна модифікація або патч, отриманий від іншого вузла), у сховищі створюється новий файл, що відповідає новому стану зміненого файлу. Важливо, щоб операція не змінила вміст файлів, які вже існують у репозиторії.

За замовчуванням репозиторії зберігаються у підкаталозі з ім'ям «.git» у кореневому каталозі робочої копії дерева файлів репозиторію. Будь-яке файлове дерево в системі можна перетворити на git-репозиторій, виконавши команду, що створює репозиторій із кореневої директорії цього дерева (або вказавши кореневу директорію у параметрах програми). Репозиторій можна імпортувати з іншого вузла, доступного через мережу. Імпорт нового репозиторію автоматично створить

робочу копію, що відповідає останньому зафіксованому стану імпортованого репозиторію (тобто будь-які зміни у робочій копії вихідного вузла, команда фіксації яких не була виконана на цьому вузлі, не будуть скопійовані).

Нижній рівень git – це так звана файлова система з адресацією вмісту. Інструмент командного рядка git включає ряд команд прямого низькорівневого управління цим репозиторієм. Ці команди не потрібні для нормальної роботи з використанням git як система контролю версій, але необхідні для складних операцій (таких як відновлення пошкодженого репозиторію) і створення власних програм на основі git-репозиторіїв, а також дає можливості.

Хеш SHA-1 обчислюється для кожного об'єкта в репозиторії, тобто файлу. Це буде ім'я файлу, що містить цей об'єкт, у каталозі git/objects. Для оптимізації роботи з файловими системами, які не використовують дерева каталогів, перший байт хешу - це ім'я підкаталогу, а інші байти – імена файлів усередині нього. такі застарілі фактори файлової системи.

Всі посилання на об'єкти репозиторію, включаючи посилання на об'єкти всередині іншого об'єкта, є хеші SHA-1.

Крім того, в репозиторіях є каталог refs, де ви можете вказати імена, що легко читаються, для деяких об'єктів Git. У команді Git два типи посилань (посилання, що посилаються, і низхідні посилання SHA-1) повністю взаємозамінні.

У традиційному поширеному сценарії репозиторії git містять три типи об'єктів: файли, дерева та комміти. Файл – це версія файлу користувача, дерево – це набір файлів з різних підкаталогів, а коміт – це дерево плюс додаткова інформація (наприклад, батьківські коміти та коментарі).

Репозиторій може запускати складання сміття. При цьому застарілі файли замінюються "дельтою" між ними та поточним файлом (тобто поточна версія файлу зберігається неінкрементно, інкременти використовуються тільки для повернення до попередніх версій). Збільшення). Ці дельти компілюються в один великий файл, який індексується. Це знижує потрібну ємність сховища.

Репозиторії Git можуть бути локальними чи віддаленими. Локальний репозиторій – це підкаталог .git, створений командою git init (порожній) і

створений командою git clone (не порожній, вміст батьківського віддаленого репозиторію відразу копіюється і зв'язується з батьківським).

Майже всі поширені операції контролю версій, такі як комміти та злиття, виконуються лише у вашому локальному репозиторії. Видалений репозиторій можна синхронізувати лише з локальним репозиторієм як вгору (push), і вниз (pull).

Git має багато переваг перед SVN, тому що весь репозиторій проекту є локальним для кожного розробника. Так, наприклад, усі операції, окрім push та pull, можна виконувати без підключення до інтернету.

Дуже потужна функція git - гілки, які реалізовані набагато повніше, ніж SVN. Гілка git – це просто іменоване посилання, що вказує на конкретну фіксацію в репозиторії (з використанням підкаталогу refs). Комміти, які не створюють нову гілку, просто переміщують це посилання, комміти, які створюють гілку, залишають старе посилання вдома, але створюють новий новий коміт і оголошують його поточним. Так само очевидним є заміна локальних файлів розробки набором файлів з іншої гілки і перехід на їх використання.

Також підтримуються підрепозиторії із синхронізацією їхньої поточної гілки.

Команда push переносить все нові дані (дані, яких ще немає у віддаленому репозиторії) з локального репозиторію до віддаленого репозиторію. Ця команда вимагає, щоб у віддаленому репозиторії не було нових комітів від інших клієнтів. В іншому випадку, push завершиться помилкою, і вам доведеться робити pull і merge.

Команда витягування протилежна команді проштовхування. Якщо одна й та сама гілка має незалежні історії в локальній та віддаленій копіях, витягування негайно переходить до злиття.

Злиття в різних файлах виконується автоматично (вся ця поведінка налаштовується) та в одному файлі за допомогою стандартного 3-панельного порівняння файлів. Після злиття конфлікти мають бути оголошені дозволеними.

Це призводить до нового стану об'єднаних локальних файлів розробника. Йому потрібно негайно зробити коміт, але цей об'єкт коміту в репозиторії містить

інформацію про те, що коміт є результатом злиття двох гілок і має два батьківські коміти.

Крім злиття, Git також підтримує операції перебазування. Операція полягає в тому, щоб взяти набір всіх змін у гілці А і потім перенести їх у гілку В. В результаті гілка В переходить у стан АВ. На відміну від злиття, проміжні комміти в гілці А не залишаються в історії гілки АВ (тільки історія гілки Б і запис самого ребазу. Це спрощує інтеграцію великих і дуже великих проектів).

Версія для Windows (офіційна версія для Windows називається mSysGit) використовує пакет mSys, порт POSIX-сумісного командного рядка Windows проекту MinGW. Всі бібліотеки та інструменти, необхідні Git, а також сам Git, були переміщені до mSys. При використанні віддаленого репозиторію SSL використовується сховище сертифікатів від mSys, а не Windows.

Існує безліч графічних оболонок Git для Windows, наприклад, TortoiseGit. Всі вони реалізуються викликами mSysGit і мають бути встановлені на вашому комп'ютері. Рішення Atlassian, SourceTree, не є винятком, але mSysGit має свої сильні та слабкі сторони (наприклад, встановлення в глибокі підкаталоги ускладнює додавання необхідних SSL-сертифікатів до mSys).

Оскільки в Windows використовується інший символ закінчення рядка, ніж у більшості Unix-подібних систем, існують варіанти (як на стороні клієнта, так і на рівні репозиторію), що забезпечують одноманітне представлення кінців рядків для команд, які використовують різні операційні системи.

Після аналізу чотирьох середовищ спільної розробки було обрано середовище Git. Це пов'язано з тим, що має велику інформаційну базу, безкоштовний доступ для двох користувачів та інтеграцію з хмарою Vercel.

2.6. Використання теорії ймовірностей для передбачення майбутньої оцінки студента

Для передбачення майбутньої оцінки була використана теорія ймовірностей.

Передбачення майбутніх оцінок студентів буде корисним для кураторів груп. Куратор зможе оцінити майбутню успішність студентів та звернути увагу на тих, хто може мати проблеми з успішністю у майбутньому.

Для цього було вирішено додати у застосунок компонент передбачення майбутньої рейтингової оцінки студента на основі оцінок минулих періодів.

Для передбачення рейтингової оцінки студентів було використано порівняння оцінок двох семестрів з урахуванням коефіцієнтів складності предметів, тобто якщо у одному семестрі була оцінка за предмет 70 з коефіцієнтом 0,6, а у другому 80 з коефіцієнтом 0,5, то остаточна оцінка за предмет у другому семестрі буде менше. Коефіцієнт додаткових балів сталий – 0,1. Якщо сума балів після застосування коефіцієнтів другого семестру меньша за таку ж сумму першого, то велика ймовірність що у наступному семестрі оцінка буде ще гірше. Але якщо коефіцієнт складності предметів у другому семестрі більше, ніж у першому, а середній бал менше, то велика вірогідність що у наступному семестрі середній бал буде більше.

3 РОЗРОБКА ПРОГРАМНОГО ЗАСОБУ ДЛЯ РОЗРАХУНКУ РЕЙТИНГОВОЇ ОЦІНКИ СТУДЕНТА

3.1 Розробка алгоритму роботи програми

Було створено алгоритм роботи програмного засобу розрахунку рейтингової оцінки студента. Алгоритм показано на рисунку 3.1.

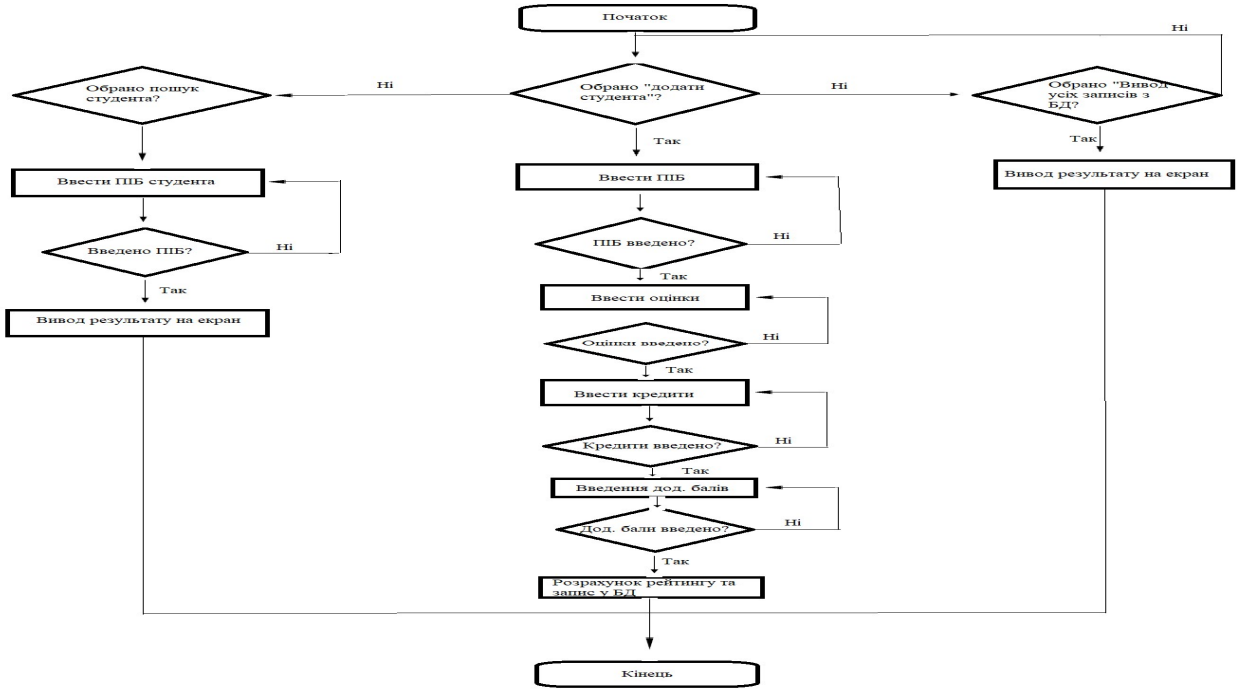

Рисунок 3.1 – Алгоритм роботи програмного засобу

Перший етап – вибір потрібної функції програмного засобу;

Другий етап – якщо вибрано пошук студента, то введення ПІБ студента; якщо вибрано додавання студента, то введення усіх даних послідовно; якщо вибрано виведення усіх записів з БД, то отримання результату на екран;

Третій етап – у випадку вибору функцій пошук студента або додавання студента отримання результату. Для додавання студента це розрахунок

рейтингової оцінки та створення запису у БД; для пошуку студента це виведення результату пошуку на екран користувача;

Четвертий етап – кінець роботи програмного засобу.

3.2 Розробка програмного забезпечення

Розроблена програма повинна бути досить простою у використанні користувачем різного рівня знань персонального комп'ютера (ПК).

Було створено таблицю для відображення стовпчиків даних студентів, таких як факультет, група, прізвище, ім'я студента, рейтингова оцінка за 1-2 семестри та прогнозована оцінка у третьому семестрі.

Таблиця була реалізована за допомогою тегів «table», «tr» та «th». Фрагмент коду наведено нижче.

```
<table class="table table-light table-sm table-striped">
                <thead>
                  <tr class="table-primary">
                    <th class="text-center" style="border-top-left-radius: 0.35em;">Фак-т</th>
                    <th class="text-center">Група</th>
                    <th class="text-center">ПІБ студента</th>
                    <th class="text-center">Сер. бал за I семестр</th>
                    <th class="text-center">Сер. бал за II семестр</th>
                    <th class="text-center" style="border-top-right-radius: 0.35em;">Прогнозована оцінка</th>
                  </tr>
                </thead>
                <tbody>
                  <tr *ngFor="let student of studentsList">
                    <td class="text-center">{{student.faculty}}</td>
```

```
                    <td class="text-center">{{student.group}}</td>
                    <td class="text-center">{{student.name}}</td>
                    <td class="text-center">{{student.term1.average}}</td>
                    <td class="text-center">{{student.term2.average}}</td>
                    <td class="text-center">{{student.projectedMark.toFixed(3)}}</td>
                </tr>
            </tbody>
        </table>
```

У ході реалізації сортування записів за факультетами було створено navbar з короткими назвами факультету, при натисканні на назву факультету вібувається сортування за параметром faculta у БД. Фрагмент коду наведено нижче:

```
<nav id="sidebar">

    <ul class="list-unstyled components border border-right-1">

        <li>
            <a href="#pageSubmenu" role="button" data-bs-toggle="collapse" aria-expanded="true"
                class="dropdown-toggle">Факультети</a>
                <ul class="collapse list-unstyled show" id="pageSubmenu">
                <div class="nav flex-column nav-pills" id="v-pills-tab" role="tablist" aria-orientation="vertical">
                    <a class="nav-link" role="button" id="v-pills-all-tab" data-bs-toggle="pill"
                        data-bs-target="#v-pills-all" role="tab" aria-controls="v-pills-all"
                        aria-selected="true">Усі Факультети</a>
```

```
                    <a class="nav-link" role="button" id="v-pills-akt-tab" data-bs-
toggle="pill"
                       data-bs-target="#v-pills-akt" role="tab" aria-controls="v-
pills-akt"
                       aria-selected="false">АКТ</a>
                    <a class="nav-link" role="button" id="v-pills-ЕЛІБІ-tab" data-
bs-toggle="pill"
                       data-bs-target="#v-pills-ЕЛІБІ" type="button" role="tab" aria-
controls="v-pills-ЕЛІБІ"
                       aria-selected="false">ЕЛІБІ</a>
                    <a class="nav-link" role="button" id="v-pills-ІК-tab" data-bs-
toggle="pill"
                       data-bs-target="#v-pills-ІК" type="button" role="tab" aria-
controls="v-pills-ІК"
                       aria-selected="false">ІК</a>
                    <a class="nav-link" role="button" id="v-pills-ІРТЗІ-tab" data-
bs-toggle="pill"
                       data-bs-target="#v-pills-ІРТЗІ" type="button" role="tab" aria-
controls="v-pills-ІРТЗІ"
                       aria-selected="false">ІРТЗІ</a>
                    <a class="nav-link" role="button" id="v-pills-ІТМ-tab" data-bs-
toggle="pill"
                       data-bs-target="#v-pills-ІТМ" type="button" role="tab" aria-
controls="v-pills-ІТМ"
                       aria-selected="false">ІТМ</a>
                    <a class="nav-link" role="button" id="v-pills-kn-tab" data-bs-
toggle="pill"
                       data-bs-target="#v-pills-kn" type="button" role="tab" aria-
controls="v-pills-kn"
                       aria-selected="false">КН</a>
```

```
                        <a class="nav-link" role="button" id="v-pills-КІУ-tab" data-bs-toggle="pill"
                           data-bs-target="#v-pills-КІУ" type="button" role="tab" aria-controls="v-pills-КІУ"
                           aria-selected="false">КІУ</a>
                    </div>
                </ul>
            </li>
        </ul>
    </nav>
```

Для функціонування відображення кнопки додавання студента тільки для адмінастратора були створені функції прослуховування відображення. Тобто коли користувач заходить у систему, система перевіряє чи є у користувача відповідна привілея на відображення та функціонування компоненту додавання студенту та вібдображує або скриває відповідну кнопку. Фрагмент коду наведено нижче:

```
export class NavbarComponent implements OnInit {

  public showButton: boolean = false;

  constructor(private _authService: AuthService) {

  }
```

При ініціювання компонента йде перевірка чи було вказано у токені користувача привілею на доступ до компоненту додавання студента.

Якщо так, то прослуховування дає респонс що треба показати кнопку додавання студента:

```
 ngOnInit(): void{
  this.showButton = localStorage.getItem('privileged') === '1'? true: false;
 }
 logout() {
  this._authService.signOut();
 }
```

При виході з облікового запису кнопка очищується та відправляється повторний запит на привілеї при повторному вході:

```
 ngOnDestroy() {
  this.showButton = false;
 }
}
```

Форма додавання студента була реалізована за допомогою тегу form. У формі присутні поля для введення даних, які були створені за допомогою тегу input name типу text та наприкінці форми присутня кнопка типу button для відправки даних на сервер, яка запускає запит типу async. Приклад наведено нижче.

```
<form>
    <div class="row gx-0 my-3 d-flex justify-content-center">
      <div class="col-4 m-2">
        <h5 class="text-center">Відомості про студента</h5>
        <div class="input-group mb-3">
          <input name="name" id="name" class="form-control" type="text"
            [(ngModel)]="student.name" placeholder="ПІБ студента"
required>
        </div>
        <div class="input-group mb-3">
```

```
                    <select name="faculty" class="form-select" aria-label="Назва факультета"
                        [(ngModel)]="student.faculty" required>
                        <option selected disabled value="" hidden>Вибрати факультет</option>
        <td>
                            <input name="t-1-subj1" class="form-control" type="text"
                                [(ngModel)]="student.term1.subj1"  placeholder="Назва" required>
                        </td>
                        <td>
```

Таким чином, також створені поля для додавання даних про оцінки студента, кількість додаткових балів та кількість кредитів відповідних дисциплін.

Для реалізації функції відправки даних про студента на сервер було створено запит типу async, який опрацьовує та записує дані про факультет у окремі строки, а дані про оцінки та кредити записує у відповідні масиви даних. Усі введені дані про оцінки та кредити мають тип string. Фрагмент коду наведено нижче.

```
async addStudent(student: unknown) {
    // const dbInstance = collection(this._db, 'students');
    await addDoc(this._studReference, student)
      .then(() => alert('Student added successfully'))
      .catch(err => alert(err.message));   }
```

Після запису даних у масив система розраховує рейтингову оцінку студента за критеріями, які вказані у положенні про нарахування стипендій ХНУРЕ, та створює у базі даних новий запис про факультет, групу, ім’я та рейтингову оцінку студента. Фрагмент коду наведено нижче.

Створюємо метод для розрахунку ймовірної оцінки:

calculateProjectedMark(term1: any, term2: any) :number{}

Створюємо змінну прогнозованої оцінки:

```
let projectedMark: number = 0;
```

Створюємо змінну для різниці між оцінками за 2 семестри перемноженими на їх коэфіцієнт:

```
let difference:number = 0;
```

Робиться перебір оцінок. Кожна з оцінок перемножується на коефіцієнт та порівнюється з оцінкою минулого семестра.

Для кореляції складності предмету та ризниці оцінок за 2 семестри, умовний бал 0.5 помножений на коефіцієнт складності додається на користь студента. Таким чином, навіть якщо середній бал студента у 2 семестрі гірший за попередній, але оцінки за складні предмети покращилися - прогнозована оцінка буде вищою

```
for(let i:number = 1; i <= 5; i++) {
  difference+= (term2[mark${i}]*term2[coef${i}]+0.5*term2[coef${i}]) -
    (term1[mark${i}]*term1[coef${i}]);
}
```

За статистикою різниця середньої оцінки за 2 семестри більш ніж у 10 балів є маловірогідною тож у випадку коли показники перевищують це число, результатами підрахунку ми нехтуємо:

```
if(difference <= 10) {
```

```
    projectedMark = term2.average + difference;
  } else {
    if(term1.average > term2.average) {
      projectedMark = term2.average - 10;
    } else {
      projectedMark = term2.average + 10;
    }
  }
  return projectedMark;
}
```

При виведені даних на екран користувача для кожного запису створюється новий динамічний елемент таблиці td. Для виведення також використовується запит типу async Приклад створення елементу наведено нижче.

```
async retrieveStudents(){
  this.afsData = [];
  // const studCol = collection(this._db, 'students')
  // console.log(prodsCol);

  await getDocs(this._studReference)
    .then((res:QuerySnapshot<DocumentData>) => {
      (res.docs.map(doc => {
        const newStud = ({...doc.data(), id: doc.id});
        // console.log(prod);
        this.afsData.push(newStud);
      }));
  });
}
```

Для можливості видалення даних з БД користувачем створено кнопку «Видалити» на сторінці студента, на яку можна перейти по кліку на ПІБ. При натисканні на кнопку «Видалити» відправляється запит на сервер про видалення відповідних цьому рядку даних з БД. Приклад запиту наведено нижче.

```
async deleteStudent(id: string) {
   await deleteDoc(doc(this._studReference, id ))
     .then(() => {
       alert('Student Successfully deleted')
     })
     .catch(e => console.log(e));
  }
```

3.3 Розробка людино-машинного інтерфейсу

Для взаємодії користувача з АС розроблений інтерфейс за допомогою якого реалізується функція введення вхідних даних для проведення розрахунку рейтингової оцінки студента.

Розглянемо на прикладі аккаунту адміністратору.

Після відкриття програми з’являється початкова сторінка (рис. 3.2), на якій розміщені кнопки для виконання таких функцій як додавання нового студента, виведення на екран усіх записів з БД.

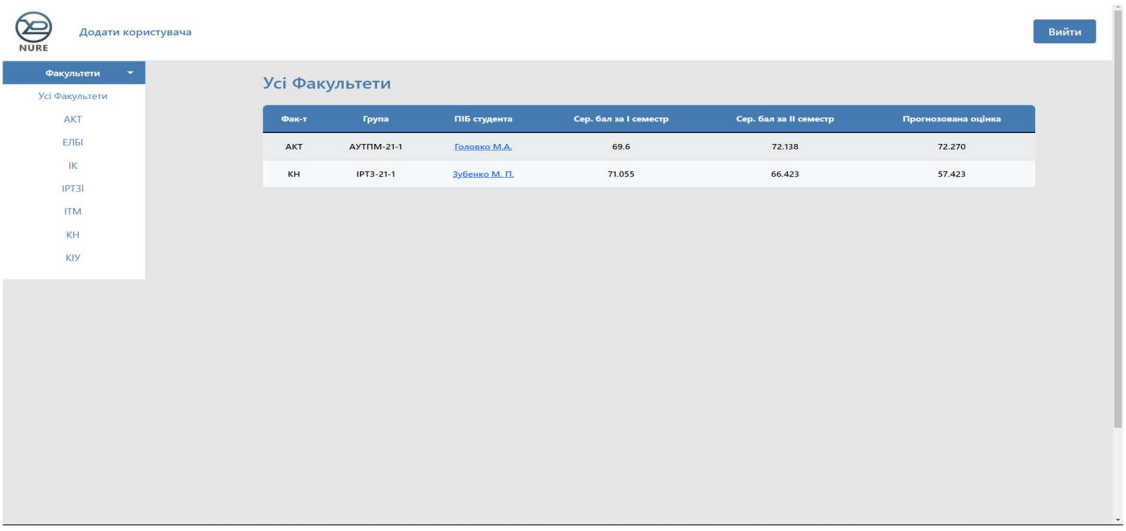

Рисунок 3.2 – Початкова сторінка

Якщо користувач вибрав кнопку Додати користувача (рис. 3.3), то з’являється діалогове вікно для додавання нового студента (рис. 3.4).

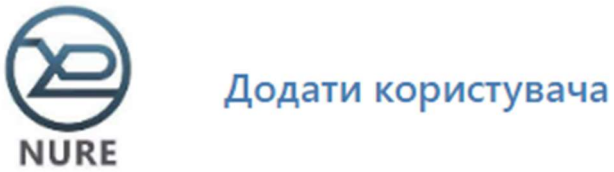

Рисунок 3.3 – Кнопка додавання студента

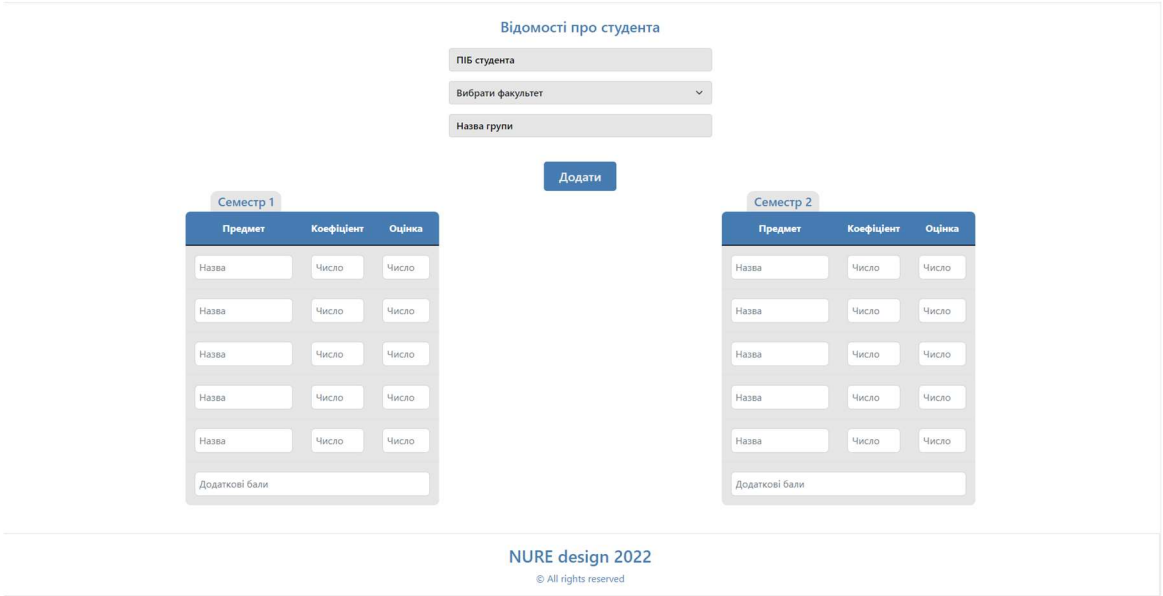

Рисунок 3.3 – Форма додавання студента

Далі користувач заповнює всі дані у формі та натискає кнопку Submit (рис. 3.4).

Рисунок 3.4 – Приклад заповнення форми додавання студента

Після натискання Submit відбувається відправка даних на сервер, розрахунок рейтингової оцінки та створення запису у БД. Приклад запису у БД наведено нижче на прикладі одного семестру.

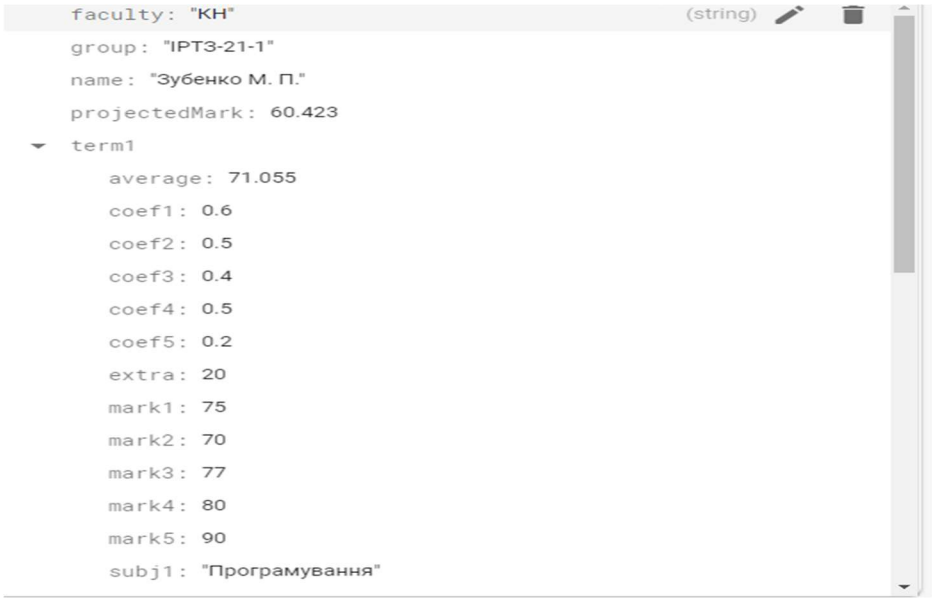

Рисунок 3.5 – Приклад запису у БД

На головній сторінці є можливість використати фільтри за факультетом (рис. 3.6).

Рисунок 3.6 – Вибір фільтра за факультетом АКТ

Також є можливість подивитись усі оцінки студента та видалити запис(для адміністратора). Для цього потрібно натиснути на ПІБ студента. Приклад виводу сторінки студента (рис. 3.7).

Рисунок 3.7 – Вивід сторінки студента

Для перевірки роботи програмного засобу було проведено контрольний експеримент по додаванню нового запису студента Шульженко К. І. з групи ІРТЗ-21-1.

Було додано дані за двома семестрами та проведено розрахунок рейтингових та передбачуваної оцінки. Дані наведено нижче:

faculty "КН"
group "ІРТЗ-21-1"
name "Шульженко К.І."
projectedMark 91.805 – передбачувана оцінка
term1
average 75.228 – середня оцінка за перший семестр
coef1 0.6
coef2 0.5
coef3 0.4
coef4 0.5
coef5 0.2
extra 20 – додаткові бали за перший семестр
mark1 80
mark2 90
mark3 85
mark4 80
mark5 60
subj1 "Програмування"
subj2 "Штучний інтеллект"
subj3 "Вища математика"
subj4 "Алгоритмізація"
subj5 "Українська мова"
term2
average 81.805 – середня оцінка за другий семестр

coef1 0.6

coef 2 0.5

coef 3 0.4

coef 4 0.5

coef 5 0.2

extra 15 – додаткові бали за другий семестр

mark1 90

mark2 95

mark3 92

mark4 90

mark5 65

subj1 "Програмування"

subj2 "Штучний інтеллект"

subj3 "Дискретна математика"

subj4 "Алгоритмізація"

subj5 "Правознавство"

У ході проведення експерименту визначено що застосунок працює вірно і розрахунки проводяться також вірно.

3.4 Розрахунок надійності розробленого засобу

Загальна модель надійності програмних засобів показана на рисунку 3.14.

Розглянемо дві концепції, які необхідно враховувати про розробці програмного забезпечення.

Рисунок 3.14 – Загальна модель надійності програмних засобів

У боротьбі зі складністю ПЗ використовуються дві концепції:

– ієрархічна структура. Ієрархія дозволяє розбити систему за рівнями розуміння (абстракції, управління). Концепція рівнів дозволяє аналізувати систему, приховуючи несуттєві для даного рівня деталі реалізації інших рівнів. Ієрархія дозволяє розуміти, проектувати і описувати складні системи;

– незалежність. Згідно з цією концепцією, для мінімізації складності, необхідно максимально посилити незалежність елементів системи.

Методи виявлення помилок, які базуються на введенні 3 систем різних видів надмірності:

– тимчасова надмірність. Використання продуктивності ЕОМ для контролю виконання та відновлення працездатності ПЗ після збою;

– інформаційна надмірність. Дублювання частини даних інформаційної системи для забезпечення надійності і контролю достовірності даних.

Програмна надмірність включає в себе: взаємна недовіра – компоненти системи проектуються, виходячи з припущення, що інші компоненти і вихідні дані містять помилки, і повинні намагатися їх виявити; негайне виявлення і реєстрація помилок; виконання однакових функцій різними модулями системи і зіставлення результатів обробки; контроль і відновлення даних з використанням інших видів надмірності.

Завдання забезпечення стійкості до помилок спрямовані на застосування методів мінімізації збитку, спричиненого появою помилок, і включають в себе:

– обробку збоїв апаратури;

– повторне виконання операцій;

– динамічна зміна конфігурації;

– скорочене обслуговування у випадку відмови окремих функцій системи;

– копіювання і відновлення даних;

– ізоляцію помилок.

Дається 4 групи принципів забезпечення надійності:

– попередження помилок;

– виявлення помилок;

– виправлення помилок;

– забезпечення стійкості до помилок.

Дії, спрямовані на мінімізацію помилок і збоїв:

– запобігання помилок за рахунок структурного програмування;

– приховування інформації або дозований доступ до даних з боку програмних засобів і об'єктів в об'єктно-орієнтованому програмуванні;

– налагодження;

– стійкість до збоїв;

– обробка виняткових ситуацій (перехоплення помилок, наприклад, ділення на нуль) і локалізація помилок і збоїв;

– відновлення після збою програми.

Ймовірність безвідмовної роботи P(t) – це ймовірність того, що в межах заданого напрацювання відмова системи не виникає

$$P(t) = 1 - Q / N, \tag{3.1}$$

де Q – кількість зареєстрованих відмов;

N – кількість експериментів.

Ймовірність відмови – ймовірність того, що в межах заданого напрацювання відмова системи виникає.

Це показник, зворотній попередньому

$$Q(t) = 1 - P(t), \tag{3.2}$$

де t – напрацювання, год;

$Q(t)$ – ймовірність відмови.

Інтенсивність відмов системи – це умовна щільність ймовірності виникнення відмови ПІ в певний момент часу за умови, що до цього часу відмова не виникла

$$\lambda(t) = f(t) / P(t), \tag{3.3}$$

де $f(t)$ – частота відмов;

$P(t)$ – ймовірність безвідмовної роботи.

Середнє напрацювання на відмову Ti – математичне очікування часу роботи до чергової відмови

$$Ti = \int_0^{\infty} t \cdot f(t) dt. \tag{3.4}$$

Використовуючи відому між $f(t)$, $Q(t)$ і $P(t)$, отримаємо

$$Ti = \int_0^{\infty} t \cdot Q(t)dt. \tag{3.5}$$

Основні показники надійності програмного забезпечення представлені в таблиці 3.1

Таблиця 3.1 – Розрахунки показників надійності

Назва	Значення
Кількість експериментів	50
Час на виконання одного експерименту	0,8 хв.
Кількість відмов	3
Ймовірність безвідмовної роботи P(t)	0,94
Імовірність відмови Q(t)	0,06
Інтенсивність відмов системи λ(t)	0,00041
Середнє напрацювання на відмову Ti	13,33 хв.

Провівши розрахунки надійності можна зробити висновки, що розрахунки надійності базуються на тестуванні роботи програми, час кожного тестового запуску був не більше 20 секунд, було розпізнано 50 текстових зображень, серед яких сталося 2 відмови.

Ймовірність безвідмовної роботи складає 96 % і це підтверджує той факт, що розроблене програмне забезпечення являється надійним та роботоздатним для наступного використання.

3.5. Забезпечення техніки безпеки при роботі з програмними засобами

3.5.1 Аналіз умов праці у виробничому приміщенні

Приміщення науково-дослідницької лабораторії, в якій виконувалася розробка програмного засобу, знаходиться на третьому поверсі чотирьох поверхового будинку. Розмір приміщення: ширина 4,5 м, довжина 5 м, висота 2,5 м. Площа і об'єм приміщення становить 22,5 м^2 та 56,25 м^3 відповідно.

Приміщення має одні двері шириною 0,9 м і висотою 2 м. В приміщенні знаходиться один проектувальник, який займається монтажем перетворювача напруги для системи контролю деталей. Додаткових евакуаційних виходів немає. Санітарні норми площі та об'єму приміщення для одного робочого становлять не менше 4,5 м^2 і не менше 15 м^3 відповідно. Будівля відноситься до II ступеня вогнестійкості згідно з ДБН В.1.1.7-2002 «Пожежна безпека об'єктів будівництва» [13].

3.5.2 Промислова безпека в лабораторії

По класу електробезпеки приміщення в якому виконувалася атестаційна робота згідно з правилами улаштування електроустановок (ПУЕ-2011) відноситься до нормального сухого класу приміщень без підвищеної небезпеки поразки людини електричним струмом.

Згідно з вимогами "Електробезпека. Захисне заземлення та занулення" все електрообладнання підлягає зануленню. Захисний ефект занулення полягає в зменшенні тривалості замикання на корпус, в скороченні часу впливу електричного струму на людину при нормованому значенні 0,2 с.

Для забезпечення безпеки необхідно щорічно проводити вимірювання опору ізоляції. Опір ізоляції згідно з ПУЕ-2011 має бути не менше 500 кОм.

Для забезпечення захисту від випадкового дотику до струмоведучих частин забезпечується: захисні оболонки, безпечне розташування струмоведучих частин, ізоляція струмоведучих частин, захисне відключення; також проводяться заходи щодо перевірки ізоляції струмоведучих частин. Випробування при капітальному

ремонті проводять не рідше одного разу на 3 роки. Випробування при поточних ремонтах – не рідше одного разу на рік.

Міжремонтні випробування проводяться один раз на півроку.

До організаційних заходів, згідно НПАОП 0.00-4.12-05 ведуть інструктаж (вступний, первинний, позаплановий, повторний і цільовий), контроль справності обладнання. Працівники повинні знати особливості обслуговуваних пристроїв. До обслуговування повинні допускатися працівники не молодше 18 років, що пройшли медичне обстеження, знають апаратуру і особливості її обслуговування. Працюючий персонал повинен знати і дотримуватися правил техніки безпеки, рівень яких визначається кваліфікаційною групою [14].

3.5.3 Виробнича санітарія в лабораторії

Робота розрахувальника відноситься до легкої фізичної роботи, відповідно до «Система стандартів безпеки праці. Загальні санітарно-гігієнічні вимоги до повітря робочої зони», відноситься до категорії Іа (легкої), так як ця робота виконується сидячи, і не вимагає систематичного фізичного навантаження і перенесення вантажів. Для таких робіт встановлені оптимальніпараметри мікроклімату.

Лабораторія, в якій використовують персональні комп'ютери повинна мати місцеву вентиляцію. У даній лабораторії вона відсутня, в результаті чого ми знайшли шкідливі фактори. Необхідно розрахувати параметри місцевої вентиляції. При нашому обсязі повітря на робітника (більше 20 м^3) кількість припливного повітря повинна бути не менше Gi = 20 м^3 / год на робочий.

Кількість припливного повітря з урахуванням кількості людей в приміщенні розраховується за формулою

$$G = G_i \cdot n = 20 \cdot 1 = 20\ м^3/год. \quad (3.6)$$

Визначимо кількість повітря L для місцевої вентиляції за кількістю шкідливих випарів з розрахунку розчину їх в максимально допустимих концентраціях

$$L = 1000\frac{n \cdot g}{g - g_1}, \quad (3.7)$$

де n – коефіцієнт, що враховує частку шкідливих речовин, що надходять вробочу зону;

g – максимально допустима концентрація, мг/м^3; g = 0,01 г/м^3;

g_1 – кількість шкідливих газів в припливно повітрі; g_1 = 0,004 мг/м^3;

$$L = \frac{2 \cdot 10^{-4}}{10^{-5} - 4 \cdot 10^{-6}} = 166\ . \quad (3.8)$$

Опір тертя R в круглих сталевих повітроводах, який повинен подолативентилятор, визначається за формулою:

$$R = \frac{a \cdot l \cdot p}{d} \cdot \frac{v^2}{2g}, \quad (3.9)$$

де a – коефіцієнт тертя (=0,02);

l – довжина воздуховоду, м;

d – діаметр воздуховоду;

p – питома маса повітря, кг/м^3;

v – середня швидкість повітря в повітроводі, м/сек;

g – прискорена сила тяжіння, м/с^2.

При цьому середня швидкість повітря в повітроводі

$$V = \frac{L}{3600 \cdot F}, \quad (3.10)$$

де L – годинна витрата повітря, м^3/год;

F – площа перетину каналу, м^2 (приймаємо F = 0,03 м^2).

Підставив відповідні значення, отримаємо:

$$VV = \frac{166}{3600 \cdot 0,03} = 1,54 \text{ м/с}, \quad (3.11)$$

тоді

$$RR = \frac{0,02 \cdot 200 \cdot 1,8 \cdot 1,54^2}{0,2 \cdot 2 \cdot 9,81} = 4,35 \text{ кг/м}^3, \quad (3.12)$$

Оскільки опір тертя повітряного потоку в трубі складає 30 %, то

$$N = \frac{L_1 \cdot H}{n \cdot 3600 \cdot 102}, \quad (3.13)$$

Для подальших розрахунків приймаємо $R_{общ}$ = 15 кг / м^2. Потужність вентилятора визначається за формулою

$$N = \frac{L_1 \cdot H}{n \cdot 3600 \cdot 102}, \quad (3.14)$$

де L_1 – продуктивність вентилятора, м^3/год;

H – повний тиск повітря, створений вентилятором, Па;

n – коефіцієнт корисної дії (обирається в межах 0,5 – 0,55).

При цьому:

$$H = \frac{p \cdot r \cdot v^2}{r} + R_{\text{общ}} = \frac{1,8 \cdot 1 \cdot 1,54}{1} + 15 = 19,27 \text{ кг/м}^3. \quad (3.15)$$

Тоді

$$NN = \frac{200 \cdot 19,27}{0,5 \cdot 3600 \cdot 102} = 0,021 \text{ кВт}. \quad (3.16)$$

Оскільки двигун повинен забезпечувати десятикратний запас потужності, то вибираємо двигун з потужністю не менше 210 Вт. Згідно з отриманими результатами, вибираємо вентилятор типу Ц4-70, (n в = 1400 об / хв).

ВИСНОВКИ

В ході виконання кваліфікаційної роботи було проаналізовано автоматизовані програмні засоби у сфері освіти, виявлено їх недоліки та переваги. Було проаналізовано та використано Cloud-технології, а також теорію ймовірностей.

В результаті проведеного аналізу виявлено, що програмний засіб розрахунку рейтингової оцінки студента має велику актуальність серед працівників закладів навчання, які зможуть використовувати її у своїй роботі та вдосконалити розрахунок та зберігання рейтингової оцінки студента, а також простоту та надійність програмного засобу.

У ході роботи було вирішено такі завдання:

– проаналізовано сучасні засоби розрахунку в освіті. В результаті виявлено особливості таких програмних саме в галузі освіти (проаналізовано людино-машинний інтерфейс таких програмних засобів);

– проаналізовано структуру програмних засобів (виявлено основні структурні елементи програмних засобів);

– розроблено алгоритм роботи програми, який детально описує процес роботи розробленого програмного засобу;

– проаналізовано та обрано середовище розробки програмного засобу (в результаті в роботі обгрунтовано середовище розробки VisualStudio Code та мову програмування Angular JS);

– розроблено людино-машинний інтерфейс програмного засобу (принцип роботи якого детально описано в роботі);

– проведено розрахунок надійності системи (в результаті проведених розрахунків виявлено, що система налаштована на 94%);

– проведено експериментальну перевірку розрахунку надійності розробленої програми.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Головко, М. А. Анализ современных систем автоматизации расчетов [Текст] / М. А. Головко // Конференция «Автоматизированные системы и компьютеризованные технологии радиоэлектронного приборостроения». – 2021. – С. 65-66.

2. ДСТУ 3008-15. Документація. звіти у сфері науки і техніки. структура і правила оформлення [Текст] – Введ. 2015-06-22. – К. Держстандарт України, 2017. – 29 с.

3. Освітньо – професійна програма «Автоматизоване управління технологічними процесами» другого (магістерського) рівня вищої освіти за спеціальністю 151 – «Автоматизація та комп'ютерно – інтегровані технології» галузь знань 15 «Автоматизація та приладобудування» (Вчена рада ХНУРЕ протокол № 1 від 28.01.2021 р.) [Електронний ресурс]. – Режим доступу: https://tapr.nure.ua/wp-content/uploads/2021/03/opp_autp_2021_compressed.pdf
ДСТУ 3008-15. Документація. Звіти у сфері науки та техніки. структура та правила оформлення. – Введ. 2015-06-22. – К. Держстандарт України, 2017 – 29 с.

4. Abishov, N. Development of an automated information system university management [Текст] / N. Abishov // Procedia-Social and Behavioral Sciences. – 2018. – Т. 143. – С. 550-554.

5. Delgado, J. M. D. Robotics and automated systems in construction: Understanding industry-specific challenges for adoption / J. M. D. Delgado // Journal of Building Engineering. – 2019. – Т. 26. – С. 100868.

6. Тайзетди-Тайзетдинова А. Г. Анализ автоматизированных систем дистанционного обучения / А. Г. Тайзетди-Тайзетдинова // Инновационное развитие профессионального образования. – 2016. – №. 2 (10).

7. Основы АСУ [Електронний ресурс]. – Режим доступу: https://studfile.net/preview/3828347/page:40/.

8. An Overview on Industrial Automation – Need, Structure, Types & Technologies [Електронний ресурс]. – Режим доступу: https://www.elprocus.com/an-overview-on-industrial-automation/.

9. [Електронний ресурс]. – режим доступу: https://www.researchgate.net/profile/Isabel-Demongodin/publication/3421718/figure/fig1/AS:394703557152775@1471116069563/Structure-of-a-typical-industrial-automation-system.png.

10. Автоматизированные системы управления структура [Електронний ресурс]. – режим доступу: https://chem21.info/info/793159/.

11. Автоматизированные системы управления технологическими процессами [Електронний ресурс]. – режим доступу: http://ktk-kuban.ru/wp-content/uploads/Metod/PopovaEP/metod_pop2.pdf.

12. Надежность и диагностика ТКС [Електронний ресурс]. – режим доступу: https://studfile.net/preview/3025468/page:4/.

13. Положення про стипендіальне забезпечення ХНУРЕ [Електронний ресурс]. – режим доступу: https://nure.ua/wp-content/uploads/Main_Docs_NURE/85-vid-02.03.2021-pro-stipendialne_zabezpechennja.pdf

14. Офіційний сайт VisualStudioCode [Електронний ресурс]. – режим доступу: https://code.visualstudio.com/

15. Офіційний сайт JetBrains [Електронний ресурс]. – режим доступу: https://www.jetbrains.com/ru-ru/phpstorm/

16. Офіційний сайт SublimeText [Електронний ресурс]. – режим доступу: https://www.sublimetext.com/

17. Офіційний сайт NetBeans [Електронний ресурс]. – режим доступу: https://netbeans.apache.org/