Детектор Плагиата v. 2867 - Отчёт оригинальности: 15.06.2025 12:24:19

Проанализированный документ: Пояснювальна записка.docx Лицензия: ВОЛОДИМИР МАТІЄВСЬКИЙ

Тип поиска: Поиск переписанного Язык: Uk

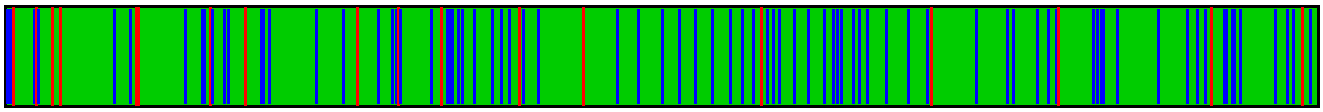
Тип проверки: Интернет

TEE и кодировка: DocX n/a

Детальный анализ тела документа:

Диаграмма соотношения частей:

Граф распределения зон:



Источники плагиата: 20

4%	905	1.	https://ela.kpi.ua/bitstreams/1e5f796e-5bbd-42d6-a46e-c520e0ae8e3d/download
3%	549	2.	https://www.mil.gov.ua/content/tenders/P_zmin_kos_01122016_.pdf
1%	232	3.	https://elar.khmnu.edu.ua/bitstreams/9c1e1462-40d7-4db1-bb0b-937fc7c39072/download

Детали обработанных ресурсов: 151 - OK / 6 - Ошибок

Важные замечания:

Википедия:	Google Книги:	Сервисы платных работ:	Античит:
[не обнаружено]	[не обнаружено]	[не обнаружено]	**Обнаружено сокрытие!**

Античит-отчет UACE:

1. Статус: Анализатор **Включен** Нормализатор **Включен** сходство символов установлено на **100%**
2. Обнаруженный процент загрязнения UniCode: **10,3%** с лимитом: 4%
3. Процент нераспознанных символов после нормализации: **6,4%**
4. Все подозрительные символы будут отмечены фиолетовым цветом: Abcd...
5. Найдены невидимые символы: 0

Рекомендации по оценке:

Особое внимание следует уделить анализу этого отчета! Предполагается, что этот документ содержит значительное количество символов, чуждых языку документа. Это прямое указание на то, что автор документа использовал специальное программное обеспечение\онлайн-веб-сервис, чтобы эффективно

скрыть текст в попытке избежать обнаружения потенциального плагиата. Настоятельно рекомендуется передать это дело на более высокий уровень! В случае сомнений обращайтесь: в службу поддержки Детектора плагиата!

Алфавитная статистика и анализ символов:

Активные ссылки (URL-адреса, извлеченные из документа):

URL не найдены

Исключённые ресурсы:

URL не найдены

Включённые ресурсы:

URL не найдены

Детальный анализ документа:

Міністерство освіти і науки України ДЗ

Цитирования: **0,04%** id: **1**

«луганський Національний університет імені тараса шевченка»

Навчально-науковий інститут математики та інформаційних технологій (назва факультету, інституту) Інформаційних технологій та систем (назва кафедри) Пояснювальна записка до дипломного проєкту (роботи) БАКАЛАВРА (освітньо-кваліфікаційний рівень) на тему: РОЗРОБКА КЛІЄНТ—СЕРВЕРНОЇ СИСТЕМИ

Цитирования: **0,01%** id: **2**

«ЕЛЕКТРОННИЙ ДЕКАНАТ»

ЗАСОБАМИ JAVA Виконав: студент 4 курсу, групи___ напряму підготовки (спеціальності) 123

Цитирования: **0,01%** id: **3**

«Комп’ютерна інженерія»

(шифр і назва напряму підготовки, спеціальності) Курячий О.В. (

Обнаружен Плагиат: **0,07%** https://elartu.tntu.edu.ua/bitstream/li... id: **4**

прізвище та ініціали) Керівник Переяславська С.О. (прізвище та ініціали) Рецензент

_____Козуб Ю. Г. (прізвище та ініціали) Полтава – 2025 року Зміст ВСТУПЗ РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ТЕХНОЛОГІЙ5 1.1. Аналіз процесу управління обліку студентів5 1.2. Вимоги до електронного деканату8 1.3. Вибір технологій для розробки9 1.4. Огляд клієнт-серверної архітектури11 РОЗДІЛ 2. ПРОЕКТУВАННЯ СИСТЕМИ17 2.1. Архітектура клієнт-серверної системи17 2.2. Модель бази даних та її структура19 2.3. Проектування серверної частини23 2.4. Проектування клієнтської частини24 2.5. Опис взаємодії між клієнтом і сервером25 РОЗДІЛ 3. РЕАЛІЗАЦІЯ СИСТЕМИ28 3.1. Опис серверної частини28 3.2. Опис клієнтської частини58 ВИСНОВКИ90 СПИСКИ ВИКОРИСТАНИХ ДЖЕРЕЛ92 ДОДАТКИ95 Змн. Арк. № докум. Підпис Дата Арк. 2 ІТС.4КІ.0124.03-ПЗ Розроб. Курячий О.В. Керівник Переяславська С.О. Реценз. Козуб Ю.Г. Н. Контр. Зав. каф. Семенов М.А. ЗМІСТ Літ. Акрушів 1 ЛНУ Кафедра ІТС, Гр.4КІ . ВСТУП В сучасних умовах цифровізації суспільства та активного розвитку інформаційних технологій особливу увагу приділяють автоматизації управління та обробки інформації в різних сферах діяльності. Однією з таких сфер є освітні заклади, де управління студентськими даними, ведення академічної документації та комунікація між викладачами й студентами потребують ефективного програмного забезпечення. Традиційні методи обліку та адміністрування освітніх процесів часто передбачають використання паперових або застарілих електронних систем, які не відповідають сучасним вимогам ефективності, безпеки та зручності. Внаслідок цього виникає потреба у впровадженні інноваційних рішень для автоматизації цих процесів. Одним із таких рішень є електронний деканат — система, що дозволяє керувати студентськими даними, розкладом занять, успішністю та іншими адміністративними аспектами навчального процесу. Об’єктом дослідження є процес управління студентськими даними в освітніх установах. Предметом дослідження є клієнт-серверна система

Цитирования: **0,01%** id: **5**

«Електронний деканат»,

Обнаружен Плагиат: **0,09%** https://dglibtest.nubip.edu.ua/bitstrea... id: **6**

розроблена з використанням мови програмування Java, технологій MySQL та TCP-з’єднання між клієнтом і сервером.

Метою роботи є розробка клієнт-серверної системи

Цитирования: **0,01%** id: **7**

«Електронний деканат»,

яка забезпечує зручний і безпечний доступ до даних, їх обробку та управління освітньою документацією. Для досягнення цієї мети передбачено вирішення таких основних завдань: аналіз предметної області та існуючих аналогів систем електронного деканату; Змн. Арк. № докум. Підпис Дата Арк. 3 ІТС.4КІ.0124.03-ПЗ Розроб. Курячий О.В. Керівник Переяславська С.О. Реценз. Козуб Ю.Г. Н. Контр. Зав. каф. Семенов М.А. ВСТУП Літ. Акрушів 2 ЛНУ Кафедра ІТС, Гр.4КІ . визначення вимог до програмного забезпечення та вибір

оптимальних технологій для розробки; проєктування архітектури клієнт-серверної системи та моделі бази даних; реалізація серверної частини системи з урахуванням вимог до безпеки та продуктивності; розробка клієнтської частини з графічним інтерфейсом для взаємодії з користувачем; тестування та оптимізація роботи системи для забезпечення її стабільної роботи. Запропоноване програмне рішення дозволить суттєво спростити адміністрування навчального процесу, підвищити швидкість обробки інформації та мінімізувати ймовірність помилок, пов’язаних із людським фактором. Використання клієнт-серверної архітектури забезпечить ефективний розподіл навантаження та зручність доступу до даних для користувачів системи. Таким чином, розробка даної системи є актуальним завданням у сфері автоматизації управління освітніми установами. Змн.

Обнаружен Плагиат: **0,13%** https://elar.khmnu.edu.ua/bitstreams... id: **8**

Арк. № докум. Підпис Дата Арк. 4 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

ТА ТЕХНОЛОГІЙ 1.1. Аналіз процесу управління обліку студентів В сучасних навчальних закладах управління обліком студентів є важливою частиною організації навчального процесу. Системи управління обліком студентів повинні забезпечити ефективне ведення та обробку даних, що стосуються академічної діяльності студентів, а також допомогти адміністрації університету, деканатам та викладачам у здійсненні поточного контролю успішності та академічних досягнень студентів [22]. Процес управління обліком студентів Процес управління обліком студентів у загальному випадку можна поділити на кілька основних етапів: Змн.

Обнаружен Плагиат: **0,23%** https://elar.khmnu.edu.ua/bitstreams... id: **9**

Арк. № докум. Підпис Дата Арк. 5 ІТС.4КІ.0124.03-ПЗ Розроб. Курячий О.В. Керівник Переяславська С.О. Реценз. Козуб Ю.Г. Н. Контр. Зав. каф. Семенов М.А. РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

ТА ТЕХНОЛОГІЙ Літ. Акрушів 12 ЛНУ Кафедра ІТС, Гр.4КІ . Реєстрація студентів у системі: Перший етап включає введення базових даних про студента в систему, що може включати особисті дані (Прізвище та ім’я, дата народження, контактні дані), академічні досягнення, інформацію про факультет та спеціальність. Цей етап є критично важливим, оскільки правильність і повнота внесених даних безпосередньо впливає на подальшу роботу з системою. Реєстрація на курси та дисципліни: На цьому етапі система дозволяє студенту обирати курси та дисципліни, на які він буде записаний протягом семестру. Процес автоматизує контроль за кількістю місць на курсах, дозволяє студентам бачити доступні курси та реєструватися на них, а також надає адміністрації можливість контролювати процес вибору дисциплін, реєстрацію на спеціалізації тощо. Моніторинг успішності: Після реєстрації на курси система слідкує за результатами студентів протягом семестру. Вона зберігає дані про оцінки за різноманітні завдання, тестування, контрольні роботи, заліки та екзамени. Це дає змогу викладачам та деканатам оперативно отримувати актуальну інформацію про успішність кожного студента, а також сприяє зручності для студентів, які мають можливість постійно перевіряти свої досягнення. Змн. Арк. № докум. Підпис Дата Арк. 6 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Ведення академічної історії: Протягом всього періоду навчання система автоматично формує академічну історію студента, що включає інформацію про курси, оцінки, академічні досягнення, а також історію переведення на інші факультети чи спеціальності, відпустки, відрахування або поновлення. Це дає змогу створювати повну картину успішності та кар’єри студента в межах навчального закладу. Управління випуском студентів: Наприкінці навчання студенти отримують дипломи та сертифікати, що підтверджують їхні досягнення. Система управління обліком студентів повинна автоматизувати процес видачі документів, а також надавати необхідну інформацію для формування державних атестатів, дипломів про вищу освіту, сертифікатів про курси тощо. Проблеми та обмеження існуючих процесів Хоча процеси управління обліком студентів є стандартними в більшості навчальних закладів, існують низка проблем, що супроводжують ці процеси: Відсутність інтеграції між системами: Багато університетів використовують окремі системи для різних етапів обліку студентів (реєстрація, оцінки, академічні досягнення). Це призводить до проблем з інтеграцією та обміном даними, що затрудняє адміністрування та роботу з інформацією. Неефективність ручного вводу даних: На багатьох етапах процесів обліку студентів все ще існують мануальні операції введення даних. Це може спричиняти помилки при обробці інформації та уповільнювати роботу з даними. Недостатня автоматизація аналізу

успішності: У багатьох системах управління обліком студентів не вистачає інструментів для глибокого аналізу успішності студентів, наприклад, для прогнозування потенційних проблем з успішністю або для генерації детальних звітів для викладачів та адміністрації. Змн. Арк. № докум. Підпис Дата Арк. 7 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Захист даних: Оскільки системи містять конфіденційну інформацію про студентів, забезпечення належного рівня безпеки та захисту даних є важливою проблемою. Багато існуючих систем мають вразливості щодо захисту особистих даних студентів, що ставить під загрозу їх конфіденційність [2]. Напрями вдосконалення процесів Інтеграція систем: Для покращення обробки даних та уникнення дублювання важливо інтегрувати різні системи, що використовуються в навчальному процесі, такі як системи реєстрації студентів, оцінювання, бібліотечні системи та інші, в єдину централізовану платформу [26]. Автоматизація збору та обробки даних: Замість ручного введення даних, можна використовувати автоматизовані системи збору інформації, що забезпечують точність та швидкість обробки даних, а також знижують ймовірність помилок. Аналіз успішності з використанням ІТ-рішень: Використання штучного інтелекту та алгоритмів машинного навчання для прогнозування успішності студентів може значно підвищити ефективність системи, а також допомогти виявляти потенційні проблеми ще до того, як вони стануть критичними. Посилення захисту даних: Використання сучасних технологій шифрування та захисту даних, а також постійне оновлення систем безпеки допоможуть зберегти конфіденційність даних студентів. [2] Змн. Арк. № докум. Підпис Дата Арк. 8 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ 1.2. Вимоги до електронного деканату Розробка клієнт-серверної системи

Цитирования: **0,01%** id: **10**

«Електронний деканат»

передбачає створення інформаційної системи для автоматизації обліку студентів, викладачів, курсів та оцінок. Дана, система має відповідати наступним функціональним та нефункціональним вимогам. [18, 16, 30] Функціональні вимоги Облік студентів: Додавання та видалення студентів. Призначення студентів у навчальні групи. Збереження академічної історії студента. Облік викладачів: Додавання та видалення викладачів. Призначення викладачів на конкретні дисципліни. Адміністрування навчальних дисциплін: Додавання та видалення курсів. Призначення конкретних дисциплін для студентів та викладачів. Облік оцінок: Внесення оцінок студентів у систему. Збереження історії оцінювання. Можливість перегляду оцінок студентами та викладачами. Реєстрація та автентифікація: Реєстрація користувачів. Авторизація користувачів. Захист від несанкціонованого доступу. Шифрування паролів. Клієнт-серверна взаємодія: Використання сервера бази даних. Використання окремого серверного додатка для обробки запитів. Клієнтський додаток для взаємодії користувачів із серверною частиною. Нефункціональні вимоги Безпека: Шифрування даних користувачів. [2] Зручність використання: Інтуїтивно зрозумілий графічний інтерфейс для роботи користувачів. Дружній дизайн з урахуванням принципів UX/UI. 1.3. Вибір технологій для розробки В процесі розробки клієнт-серверної системи

Цитирования: **0,01%** id: **11**

«Електронний деканат»

важливо обрати ефективні технології, які забезпечать стабільність, продуктивність та зручність у використанні. У даному підрозділі розглянуто вибір основних технологій: Java для програмної логіки, MySQL для управління базами даних, Maven для управління залежностями та збірки проєкту, а також засоби створення графічного інтерфейсу користувача (GUI). Огляд технологій Java Java —

Обнаружен Плагиат: **0,06%** https://metod.vntu.edu.ua/getfile.php... id: **12**

це об’єктно-орієнтована мова програмування, що широко використовується для розробки

корпоративних застосунків, зокрема клієнт-серверних систем. Вона забезпечує платформонезалежність завдяки використанню віртуальної машини Java (JVM), дозволяє запускати застосунок на різних операційних системах без змін у коді [5, 28].

Обнаружен Плагиат: **0,38%** https://ela.kpi.ua/bitstreams/1e5f796... + 8 ресурсов! id: **13**

Змн. Арк. № докум. Підпис Дата Арк. 9 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ MySQL MySQL — це реляційна система управління базами даних (СУБД), яка

відзначається високою продуктивністю, надійністю та масштабованістю. Вона підтримує мову SQL для виконання запитів і добре інтегрується з Java через драйвер JDBC [6, 7]. Змн. Арк. № докум. Підпис Дата Арк.

10 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Maven Maven — це система управління збіркою та залежностями, яка спрощує процес компіляції, тестування та розгортання Java-додатків. Вона автоматизує завантаження необхідних бібліотек та структуризацію проєкту [21]. Графічний інтерфейс користувача Для створення зручного інтерфейсу користувача використовуються бібліотеки JavaFX або Swing. В подальшому буде використовуватися Swing [15]. Переваги вибору технологій Java Портативність: програма може працювати на будь-якій ОС з підтримкою JVM. Висока продуктивність завдяки JIT-компіляції. Велика спільнота та підтримка. Інтеграція з різними технологіями для створення масштабованих додатків. MySQL Оптимізована для роботи з великими обсягами даних. Підтримка складних SQL-запитів. Висока швидкість виконання операцій читання та запису. Легка інтеграція з Java через JDBC. Maven Автоматизація процесу збірки. Управління залежностями без потреби в ручному завантаженні бібліотек. Стандартна структура проєкту, що полегшує роботу команди розробників. 1.4. Огляд клієнт-серверної архітектури Клієнт-серверна архітектура є одним із найпоширеніших підходів до побудови інформаційних систем. Вона базується на розподілі функціональності між двома основними компонентами: клієнтом, який запитує ресурси або послуги, і сервером, який обробляє ці запити та повертає результати. Така модель дозволяє створювати масштабовані, гнучкі та ефективні системи, які відповідають сучасним вимогам до продуктивності та безпеки [20]. Визначення клієнт-серверної архітектури Клієнт-серверна архітектура передбачає, що клієнти взаємодіють із сервером через мережу. Сервер виконує обчислення, зберігає дані та надає необхідні ресурси клієнтам. Основні принципи цієї моделі включають: Розподіл навантаження: клієнти виконують лише частину обчислень, а основна робота зосереджена на сервері. Централізоване управління даними: всі дані зберігаються на сервері, що забезпечує їхню цілісність і безпеку. Мережева взаємодія: клієнти та сервери можуть бути розміщені на різних пристроях та з'єднуватися через локальну або глобальну мережу. Типи клієнт-серверної архітектури Існує кілька типів клієнт-серверної архітектури залежно від рівня розподілу функцій між клієнтом і сервером: Одноріvнева архітектура (Thick Client) В такій архітектурі клієнт виконує більшість операцій, включаючи Змн. Арк. № докум. Підпис Дата Арк. 11 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ обробку даних, а сервер використовується переважно для збереження інф- ормації. Цей підхід менш залежний від мережі, але важко масштабується. Рис. 1.1. Одноріvнева архітектура клієнт-сервер Дворівнева архітектура (Two-Tier) Цей підхід передбачає чіткий розподіл між клієнтом та сервером. Клієнтська частина взаємодіє із сервером бази даних через SQL-запити. Вона часто використовується у корпоративних системах завдяки простоті реалізації. Рис. 1.2. Дворівнева архітектура клієнт-сервер Трирівнева архітектура (Three-Tier) Змн. Арк. № докум. Підпис Дата Арк. 12 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Тут додається ще один рівень — програмний сервер (Application Ser-ver), який виконує бізнес-логіку та обробку запитів. Це підвищує безпеку, гнучкість і продуктивність системи [14, 17]. Рис. 1.3. Трирівнева архітектура клієнт-сервер Багаторівнева архітектура (Multi-Tier) Ця модель включає кілька серверів, кожен з яких відповідає за конкретний функціонал (база даних, бізнес-логіка, API, веб-інтерфейс). Вона використовується у великих розподілених системах, що працюють із високими навантаженнями. Рис. 1.4. Багаторівнева архітектура клієнт-сервер Опис клієнт-серверної моделі в

Цитирования: **0,01%** id: **14**

«Електронному деканаті»

Система

Цитирования: **0,01%** id: **15**

«Електронний деканат»

використовує трирівневу клієнт-серверну архітектуру, у якій клієнтська частина взаємодіє із програмним сервером, який у свою чергу обробляє запити та передає їх серверу бази даних MySQL. Змн. Арк. № докум. Підпис Дата Арк. 13 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Основні компоненти системи: Клієнт (графічний інтерфейс, реалізований за допомогою Swing) дозволяє студентам та викладачам виконувати запити щодо інформа- ції про курси, розклад та оцінки. Програмний сервер (реалізований на Java)

є проміжним рівнем між клієнтом та сервером бази даних, виконує бізнес-логіку та обробляє запити. Сервер бази даних (MySQL) зберігає інформацію про студентів, дисципліни, оцінки, викладачів та інші адміністративні дані. Обробка запитів відбувається через програмний сервер, який взаємодіє із сервером бази даних за допомогою JDBC з'єднання. Переваги та недоліки клієнт-серверної архітектури Переваги: Централізоване управління даними — всі дані зберігаються на сервері, що спрощує їхнє адміністрування та захист. Розширюваність — можна додавати нові функції без значного перероблення клієнта чи сервера. Безпека — централізоване зберігання даних дозволяє контролювати доступ до інформації та реалізовувати резервне копіювання. Ефективне використання ресурсів — клієнти виконують лише необхідний мінімум завдань, основні обчислення здійснюються сервером. Недоліки: Високе навантаження на сервер — при великій кількості клієнтів сервер може стати

Цитирования: **0,01%** id: **16**

«вузьким місцем»

системи. Залежність від мережі — у разі проблем із підключенням система може працювати нестабільно. Складність розгортання — налаштування сервера та клієнтів потребує часу та технічних знань. Клієнт-серверна архітектура є оптимальним рішенням для реалізації сис- теми

Цитирования: **0,01%** id: **17**

«Електронний деканат»,

оскільки вона забезпечує ефективний роз- поділ ресурсів, централізоване управління даними та гнучкість у масшт- абуванні. Використання трирівневої моделі дозволяє забезпечити зручну взаємодію між клієнтською програмою, програмним сервером та сервером бази даних, що сприяє підвищенню продуктивності та безпеки системи.

Обнаружен Плагиат: **0,15%** https://ela.kpi.ua/bitstreams/1e5f796… + 3 ресурсов! id: **18**

Змн. Арк. № докум. Підпис Дата Арк. 14 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Змн. Арк. № докум. Підпис Дата Арк.

15 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Висновки до розділу 1 В результаті аналізу предметної області та технологій для розробки клієнт-серверної системи

Цитирования: **0,01%** id: **19**

«Електронний деканат»

було зроблено низку важливих висновків. По-перше, системи управління обліком здобувачів освіти є невід’ємною частиною сучасного навчального процесу. Ефективність таких систем значною мірою залежить від автоматизації процесів, інтеграції між різними підсистемами та зручності взаємодії для кінцевих користувачів —здобувачів, викладачів та адміністрації закладу. По-друге, розробка інформаційної системи вимагає чіткого визначення функціональних та нефункціональних вимог, які охоплюють облік здобувачів, викладачів, дисциплін, балів, а також безпечну автентифікацію користувачів і швидку взаємодію з базою даних. По-третє, обґрунтовано вибір технологій для реалізації системи, зокрема використання Java для логіки додатку, MySQL для зберігання даних та Swing для створення інтерфейсу. Застосування клієнт-серверної архітектури дозволяє забезпечити централізоване управління інформацією, гнучке масштабування системи та високу продуктивність. Таким чином, врахування вищезазначених аспектів є запорукою створення ефективної та надійної системи

Цитирования: **0,01%** id: **20**

«Електронний деканат»,

яка відповідатиме сучасним вимогам інформаційного забезпечення навчальних закладів. Змн. Арк. № докум. Підпис Дата Арк. 16 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ РОЗДІЛ 2. ПРОЕКТУВАННЯ СИСТЕМИ 2.1. Архітектура клієнт-серверної системи Загальний опис архітектури Розроблювана система

Цитирования: **0,01%** id: **21**

«Електронний деканат»

використовує клієнт-серверну архітектуру, яка забезпечує розподіл обчислювальних

навантажень між клієнтською та серверною частинами. Це дозволяє централізовано зберігати дані, забезпечувати безпеку та підтримувати масштабованість системи [3, 29]. Обґрунтування вибору клієнт-серверної моделі Вибір клієнт-серверної архітектури зумовлений потребою в централізованому управлінні даними та безпеці доступу до них. Основні переваги цієї архітектури: Централізоване управління даними — усі важливі академічні дані зберігаються на сервері бази даних, що запобігає втраті чи дублюванню інформації. Розподіл навантаження — сервер обробляє запити від клієнтів, мінімізуючи вимоги до ресурсів клієнтських пристроїв. Безпека — клієнти взаємодіють із сервером через авторизацію та автентифікацію, що запобігає несанкціонованому доступу. Основні компоненти системи Система складається з трьох основних компонентів: Клієнтська частина — графічний інтерфейс користувача, через який здобувачі вищої освіти та співробітники навчального закладу взаємодіють із системою. Вона реалізована мовою Java з використанням технологій GUI (Swing). Змн. Арк. № докум. Підпис Дата Арк. 17 ІТС.4КІ.0124.03-ПЗ Розроб. Курячий О.В. Керівник Переяславська С.О. Реценз. Козуб Ю.Г. Н. Контр. Зав. каф. Семенов М.А. РОЗДІЛ 2. ПРОЕКТУВАННЯ СИСТЕМИ Літ. Акрушів 11 ЛНУ Кафедра ІТС, Гр.4КІ . Програмний сервер — проміжний рівень, який обробляє запити клієнтів,

Обнаружен Плагиат: **0,15%** https://duikt.edu.ua/repozitorii/ai/202… id: **22**

взаємодіє з базою даних та здійснює бізнес-логіку системи. Реалізований за допомогою Java (Swing для графічного інтерфейсу). Сервер бази даних — зберігає інформацію про

здобувачів освіти, курси, викладачів та інші сутності системи. Використовується MySQL для забезпечення швидкого доступу до даних і їхньої цілісності. ServerSocket — це спеціальний клас, об'єкти якого виконують роль сервера. Тобто можуть обслуговувати запити, які прийшли на певний сокет [1]. Клас Socket — це фактично Socket-клієнт. За допомогою нього ми можемо надіслати повідомлення певному сокету та отримати відповідь [1]. Взаємодія між компонентами Клієнт під’єднується та надсилає запит до програмного сервера через Socket. Сервер обробляє запит, виконує необхідні операції (наприклад, перевіряє облікові дані користувача) та звертається до бази даних. Сервер бази даних виконує відповідний SQL-запит та повертає результат програмному серверу. Програмний сервер обробляє отримані дані та надсилає відповідь клієнтові у відповідному форматі (JSON). Клієнтська частина відображає отриману інформацію користувачеві. Використовувані технології Клієнт-серверна взаємодія — ServerSocket, Socket. База даних — MySQL. Серверна частина — Java, Swing. Клієнтська частина — Swing для графічного інтерфейсу. Клієнт-серверна архітектура забезпечує ефективний розподіл навантаження, централізоване збереження даних і підвищений рівень безпеки. Це дозволяє створити масштабовану, безпечну та продуктивну си- Змн. Арк. № докум. Підпис Дата Арк. 18 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ стему

Цитирования: **0,01%** id: **23**

«Електронний деканат»,

що відповідатиме вимогам навчального закладу. 2.2. Модель бази даних та її структура Загальний опис моделі бази даних Для реалізації системи

Цитирования: **0,01%** id: **24**

«Електронний деканат»

використовується реляційна база даних, створена на основі MySQL. Вона містить структуровану інформацію про студентів, викладачів, дисципліни, успішність та інші ключові аспекти навчального процесу. Використання MySQL дозволяє забезпечити ефективне збереження, швидкий доступ і підтримку цілісності даних. В даній роботі використовується дві бази даних це ElectronicDean та mysite. БД ElectronicDean — це сама база про зберігання даних

Цитирования: **0,01%** id: **25**

«Електронного деканату».

База даних mysite — це база яка відповідає за автентифікацію та авторизацію даних в системі. Для повного розуміння структури та взаємозв’язків у базі даних використовуються інфологічна та даталогічна моделі. Інфологічна модель відображає концептуальні сутності та їхні зв’язки у вигляді ER-діаграми, незалежно від конкретної СУБД. Даталогічна модель, у свою чергу, деталізує структуру таблиць, зв’язки між ними та визначає основні атрибути,

враховуючи особливості реляційного підходу до збереження інформації. Інфологічна (концептуальна) модель БД ElectronicDean та даталогічна (фізична) модель бази даних ElectronicDean (див. ДОДАТОК В рис. В.1., рис.В.2.) [6, 12, 25,7]. Оскільки база даних mysite містить одну таблицю, то інфологічна та даталогічна модель цієї БД однакова (див. ДОДАТОК В рис.В.3., рис.В.4.). Змн. Арк. № докум. Підпис Дата Арк. 19 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Основні таблиці баз даних Таблиця 2.1. Таблиця users даталогічної моделі бази даних Найменування Тип Розмір Обмеження Коментарі id INTEGER — AUTO_INCREMENT PRIMARY KEY Ідентифікатор користувача name VARCHAR 50 NOT NULL Ім’я користувача surname VARCHAR 50 NOT NULL Прізвище користувача email VARCHAR 100 NOT NULL UNIQUE Електронна пошта користувача password VARCHAR 255 NOT NULL Пароль phone VARCHAR 15 — Номер мобільного телефону користувача Таблиця 2.2. Таблиця Students даталогічної моделі бази даних Найменування Тип Розмір Обмеження Коментарі StudentID INTEGER — AUTO_INCREMENT PRIMARY KEY Ідентифікатор здобувача освіти FirstName VARCHAR 50 NOT NULL Ім’я здобувача освіти LastName VARCHAR 50 NOT NULL Прізвище здобувача освіти Email VARCHAR 100 NOT NULL UNIQUE Електронна пошта здобувача освіти Змн. Арк. № докум. Підпис Дата Арк. 20 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Продовження таблиці 2.2. Phone VARCHAR 15 — Номер мобільного телефону здобувача освіти BirthDate DATE — — Дата народження здобувача освіти EnrollmentYear YEAR — NOT NULL Рік народження здобувача освіти Таблиця 2.3. Таблиця Teachers даталогічної моделі бази даних Найменування Тип Розмір Обмеження Коментарі TeacherID INTEGER — AUTO_INCREMENT PRIMARY KEY Ідентифікатор викладача FirstName VARCHAR 50 NOT NULL Ім’я викладача LastName VARCHAR 50 NOT NULL Прізвище викладача Email VARCHAR 100 NOT NULL UNIQUE Електронна пошта викладача Phone VARCHAR 15 — Номер мобільного телефону викладача Department VARCHAR 100 NOT NULL Підрозділ в якому працює викладач Таблиця 2.4. Змн. Арк. № докум. Підпис Дата Арк. 21 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Таблиця Courses даталогічної моделі бази даних Найменування Тип Розмір Обмеження Коментарі CourseID INTEGER — AUTO_INCREMENT PRIMARY KEY Ідентифікатор дисципліни CourseName VARCHAR 100 NOT NULL Назва дисципліни Description TEXT — — Опис дисципліни Credits INTEGER — NOT NULL Кількість кредитів TeacherID INTEGER — NOT NULL Ідентифікатор викладача Таблиця 2.5. Змн. Арк. № докум. Підпис Дата Арк. 22 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Таблиця Groupe даталогічної моделі бази даних Найменування Тип Розмір Обмеження Коментарі GroupID INTEGER — AUTO_INCREMENT PRIMARY KEY Ідентифікатор групи GroupName VARCHAR 50 NOT NULL UNIQUE Назва групи Specialty VARCHAR 100 NOT NULL Спеціальність EnrollmentYear YEAR — NOT NULL Рік вступу Таблиця 2.6. Таблиця StudentGroups даталогічної моделі бази даних Найменування Тип Розмір Обмеження Коментарі StudentID INTEGER — NOT NULL Ідентифікатор здобувача освіти GroupID INTEGER — NOT NULL Ідентифікатор групи Таблиця 2.7. Таблиця Grades даталогічної моделі бази даних Найменування Тип Розмір Обмеження Коментарі GradeID INTEGER — AUTO_INCREMENT PRIMARY KEY Ідентифікатор успішності Продоаження таблиці 2.7. StudentID INTEGER — NOT NULL Ідентифікатор здобувача освіти CourseID INTEGER — NOT NULL Ідентифікатор дисципліни Grade TINY INTEGER Від 0 до 100 NOT NULL CHECK Підсумок оцінки від 0 до 100 балів Змн. Арк. № докум. Підпис Дата Арк. 23 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ 2.3. Проектування серверної частини Загальний опис серверної частини Серверна частина клієнт-серверної системи

Цитирования: **0,01%** id: **26**

«Електронний деканат»

виконує роль посередника між клієнтським додатком і сервером бази даних. Взаємодія між клієнтом і сервером реалізована за допомогою TCP-з'єднання, що забезпечує стабільний та швидкий обмін даними. Основна функція сервера — приймати запити від клієнтів, обробляти їх відповідно до бізнес-логіки та взаємодіяти з сервером бази даних для отримання або збереження інформації. Завдяки такому підходу клієнтська частина мінімально навантажується, оскільки всі обчислювальні процеси, перевірка даних та взаємодія з базою даних відбуваються на сервері. Це підвищує продуктивність системи та забезпечує централізований контроль за доступом і безпекою інформації. Архітектура серверної частини Серверна частина складається з кількох основних модулів, кожен з яких реалізований у вигляді окремих Java-класів. Серед основних компонентів можна виділити такі: ServerWithGUI — головний клас, який запускає сервер та забезпечує графічний інтерфейс адміністратора. Він керує прийомом з’єднань від . клієнтів, створенням сесій, а

також забезпечує візуалізацію поточних Змн. Арк. № докум. Підпис Дата Арк. 24 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ процесів. ClientHandler — окремий потік для кожного клієнта. Цей клас відповідає за прийом та обробку запитів клієнта, їх розшифровку, виконання необхідної дії та повернення відповіді. db_dekanat, db_log — класи доступу до бази даних, які реалізують з'єднання, підготовку та виконання SQL-запитів. Вони інкапсулюють логіку роботи з таблицями БД та забезпечують надійний обмін інформацією між сервером і СУБД. TeachersManager, StudentGroupsManager, GradesManager, CoursesManager, StGrManager — управління відповідними сутностями (курси, оцінки, викладачі, студентські групи). Кожен менеджер містить методи для додавання, оновлення, видалення та отримання записів з БД. Табличні класи: CoursesTable, GradesTable, groupTable, StudentGroupsTable, TeachersTable, StudentTable — відповідають за відображення та структуризацію даних. Допоміжні класи: StudentAdder, GroupAdder, TeachersAdder, Courses, CoursesDeleter, GradesDeleter, GroupDeleter, Groups, StudentDeleter, Students, Teacher, TeachersDeleter , StGrDeleter — реалізують конкретні дії над даними, що викликаються через запити клієнта. Діаграма серверної частини (див. ДОДАТОК А рис.А.1.). 2.4. Проектування клієнтської частини Загальний опис клієнтської частини Клієнтська частина системи

Цитирования: **0,01%** id: **27**

«Електронний деканат»

реалізована у вигляді десктопного застосунку з графічним інтерфейсом, що взаємодіє із сервером через TCP-з'єднання. Вона написана мовою Java з використанням відповідних бібліотек для створення GUI. Клієнтський додаток забезпечує автентифікацію користувачів, формування та надсилання запитів на сервер, а також обробку отриманих відповідей. Інтерфейс користувача розроблений таким чином, щоб забезпечити зручний доступ до функціоналу системи, включаючи перегляд та управління навчальним процесом [15, 31]. Основні функції клієнтського додатку включають: автентифікацію користувача (реалізовано у класі RegistrationForm); формування та передавання запитів на сервер; обробку відповідей від серверної частини; представлення навчальної інформації у структурованому вигляді. Архітектура клієнтської частини Клієнтський застосунок структурований за модульним принципом. Кожен функціональний компонент представлений окремим класом: Діаграма клієнтської частини (див. ДОДАТОК А рис.А.2.). Змн. Арк. № докум. Підпис Дата Арк. 25 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ 2.5. Опис взаємодії

Обнаружен Плагиат: **0,07%** https://www.xn--b1aqmii2j.com/shcho... id: **28**

між клієнтом і сервером Комунікація між клієнтською та серверною частинами системи

Цитирования: **0,01%** id: **29**

«Електронний деканат»

Обнаружен Плагиат: **0,21%** https://www.xn--b1aqmii2j.com/shcho... id: **30**

реалізована за допомогою протоколу TCP. В основі взаємодії лежить архітектура клієнт-сервер із чітким розподілом обов'язків. Клієнт формує запити, а сервер їх обробляє та повертає відповідь у відповідному форматі. Клієнт ініціює з'єднання

з сервером за допомогою класу ServerConnection, який забезпечує: встановлення TCP-з'єднання; надсилання запитів у вигляді текстових команд; отримання та передавання відповідей для відображення у графічному інтерфейсі. На сервері взаємодію обслуговує головний клас ServerWithGUI, який містить графічний інтерфейс та логіку обробки клієнтських запитів. Безпосередню обробку запитів виконує клас ClientHandler, який приймає дані з сокета, аналізує тип запиту та викликає відповідні сервіси для роботи з базою даних [31]. Приклад типової взаємодії: Клієнт надсилає команду, наприклад: GET_COURSES. Програмний сервер приймає запит через клас ClientHandler, розпізнає команду. Змн. Арк. № докум. Підпис Дата Арк. 26 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ На основі команди сервер викликає відповідний метод, наприклад, метод з класу CoursesManager. CoursesManager передає запит до сервера бази даних, де дані витягуються безпосередньо з таблиці курсів (через об'єкти CoursesTable або db_dekanat, які виконують SQL-запити). Отримані з бази дані повертаються до програмного сервера у вигляді рядка або структури. Програмний сервер формує відповідь і надсилає її назад клієнту. Клієнтська частина обробляє відповідь і відображає інформацію у

відповідному вікні інтерфейсу (наприклад, CoursesForm). Таким чином, реалізована трикомпонентна архітектура: клієнт – програмний сервер – сервер бази даних, що забезпечує чітке розділення функцій, масштабованість і зручність підтримки системи. Висновок до розділу 2 У другому розділі було здійснено детальне проектування клієнт-серверної системи

Цитирования: **0,01%** id: **31**

«Електронний деканат»,

що дозволило визначити її архітектурні особливості, структуру бази даних, а також механізми взаємодії між основними компонентами системи. По-перше, обґрунтовано доцільність застосування клієнт-серверної архітектури, яка забезпечує централізоване управління даними, підвищену безпеку, ефективний розподіл навантаження між клієнтською та серверною частинами та гнучкість у масштабуванні. По-друге, розроблено структуровану модель бази даних, яка охоплює всі ключові елементи навчального процесу: здобувачів освіти, викладачів, дисципліни, оцінки, групи. Використання реляційної СУБД MySQL забезпечує збереження цілісності та швидкий доступ до інформації, а також підтримку складних міжтабличних зв’язків. Спроєктовані інфологічні, даталогічні моделі баз даних, а також діаграми класів серверної та клієнтської частин. По-третє, детально спроектовано серверну та клієнтську частини системи, де кожна реалізована у вигляді модулів із чітким функціональним призначенням. Сервер виконує основну бізнес-логіку та обробку запитів, а клієнт забезпечує зручний інтерфейс для користувача. Реалізована TCP-взаємодія між клієнтом і сервером гарантує стабільний обмін даними та підтримку ефективної комунікації в межах системи. Таким чином, спроєктована система

Цитирования: **0,01%** id: **32**

«Електронний деканат»

відповідає сучасним вимогам інформаційного забезпечення навчальних закладів, демонструючи високу продуктивність, безпеку та зручність у використанні. Змн. Арк. № докум. Підпис Дата Арк. 27 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ РОЗДІЛ 3. РЕАЛІЗАЦІЯ СИСТЕМИ 3.1. Опис серверної частини Серверна частина системи

Цитирования: **0,01%** id: **33**

«Електронний деканат»

Обнаружен Плагиат: **0,13%** https://duikt.edu.ua/repozitorii/ai/202… id: **34**

реалізована на мові програмування Java. Вона виконує роль центрального компонента, що забезпечує обробку запитів клієнта, взаємодію з базою даних

і повернення відповідей. Таблиця класи серверної частини системи

Цитирования: **0,01%** id: **35**

«Електронний деканат»

(див. ДОДАТОК Б табл.Б.1.). Опис класу ServerWithGUI Загальний опис Клас ServerWithGUI (див. ДОДАТОК Г Код класу ServerWithGUI) відповідає за реалізацію серверної частини клієнт-серверної системи

Цитирования: **0,01%** id: **36**

«Електронний деканат».

Він забезпечує запуск сервера, управління підключеннями клієнтів та взаємодію з ними через TCP-з’єднання. Додатково, клас містить графічний інтерфейс адміністратора для запуску та зупинки сервера, а також для перегляду підключених клієнтів. Основні компоненти класу ServerSocket serverSocket — серверний сокет, що приймає вхідні з'єднання від клієнтів. boolean isRunning — змінна для контролю стану сервера. Thread serverThread — потік для роботи сервера, що дозволяє обробляти клієнтські підключення у фоновому режимі. SessionManager — внутрішній клас, який керує активними сесіями користувачів. Змн. Арк. № докум. Підпис Дата Арк. 28 ІТС.4КІ.0124.03-ПЗ Розроб. Курячий О.В. Керівник Переяславська С.О. Реценз. Козуб Ю.Г. Н. Контр. Зав. каф. Семенов М.А. РОЗДІЛ 3. РЕАЛІЗАЦІЯ СИСТЕМИ Літ. Акрушів 62 ЛНУ Кафедра ІТС, Гр.4КІ . Графічний інтерфейс (GUI) — реалізований на основі JFrame, JLabel, JButton, JTextArea, забезпечує керування сервером та відображення підключених клієнтів. Внутрішній клас SessionManager Цей клас використовується для управління активними сесіями

користувачів. ConcurrentHashMap String, Socket activeSessions — асоціативний масив, що зберігає підключені сесії (email → socket). isUserLoggedIn(String email) — перевіряє, чи є користувач з таким email в активних сесіях. loginUser(String email, Socket socket) — додає нового користувача в список активних. logoutUser(String email) — видаляє користувача із сесії. Графічний інтерфейс сервера Створення вікна (JFrame frame) — головне вікно сервера. Кнопки керування сервером (JButton startButton, JButton stopButton) — запуск та зупинка сервера. Текстова область (JTextArea connectedClientsArea) — відображає інформацію про підключених клієнтів. Механізм керування подіями — обробники кнопок для запуску та зупинки сервера. Методи класу startServer(JLabel statusLabel, JTextArea connectedClientsArea): Запускає сервер у новому потоці, створюючи ServerSocket на порту 12345. Очікує підключень клієнтів у циклі. Змн. Арк. № докум. Підпис Дата Арк. 29 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Приймає вхідні з'єднання (Socket clientSocket). Запускає окремий потік ClientHandler для кожного підключеного клієнта. Виводить інформацію про підключених клієнтів у текстовому полі GUI. stopServer() — Зупиняє сервер шляхом закриття ServerSocket і завершення потоків. Опис класу ClientHandler Загальний опис Клас ClientHandler (див. ДОДАТОК Г Код класу ClientHandler) є складовою частиною серверної частини клієнт-серверної системи

Цитирования: **0,01%** id: **37**

«Електронний деканат».

Його головне призначення — забезпечення паралельної обробки запитів кожного клієнта, підключеного до сервера через TCP-з’єднання. Кожен екземпляр цього класу створюється при підключенні нового клієнта та виконується в окремому потоці. У рамках своєї роботи клас ClientHandler виконує обробку широкого спектру клієнтських запитів, які охоплюють основні операції з базою даних, авторизацію, реєстрацію, а також інформаційні запити. Нижче представлено структурований опис підтримуваних запитів, розподілений за категоріями для зручності використання та супроводу системи. Таблиця Додавання даних (ADD) (див. ДОДАТОК А табл.Б.2.). Таблиця Отримання даних (GET) (див. ДОДАТОК А табл.Б.3.). Таблиця 3.1. Таблиця: Видалення даних (DELETE) Команда Формат запиту Опис дії DELETE_STUDENT ELETE_STUDENT;Email Видалення інформації відповідного здобувача освіти з бази та системи

Обнаружен Плагиат: **0,18%** https://ela.kpi.ua/bitstreams/1e5f796... + 3 ресурсов! id: **38**

Змн. Арк. № докум. Підпис Дата Арк. 30 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Продовження таблиці 3.1. Змн. Арк. № докум. Підпис Дата Арк.

31 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ DELETE_COURSE DELETE_COURSE;Назва_курсу Видалення відповідної дисципліни DELETE_GROUP DELETE_GROUP;Назва_групи Видалення відповідної академічної групи DELETE_TEACHERS DELETE_TEACHERS;Email Видалення інформації про відповідного викладача Таблиця 3.2. Таблиця: JSON-запити (спеціальні) JSON Команда Формат JSON-запиту Опис дії DELETE_STUDENT_FROM_GROUP {

Цитирования: **0,01%** id: **39**

"command":

Цитирования: **0,01%** id: **40**

"DELETE_STUD ENT_FROM_GROUP",

Цитирования: **0,01%** id: **41**

"studentID":

ID,

Цитирования: **0,01%** id: **42**

"groupID":

ID} Видаляє інформацію про відоповідного здобувача освіти з академічної групи DELETE_GRADES {"grades":" DELETE_GRADES

Цитирования: **0%** id: **43**

","

studentID

Цитирования: **0,01%** id: **44**

":ID,"

courseID":ID} Видаляє підсумкові бали відоповідного здобувача освіти з відповідної дисципліни Опис класу Courses Загальний опис Клас Courses є частиною модельного рівня клієнт-серверної системи

Цитирования: **0,01%** id: **45**

«Електронний деканат»

і представляє сутність

Цитирования: **0,01%** id: **46**

«Дисциплін».

Цей клас використовується для зберігання базової інформації про дисципліни, такої як його унікальний ідентифікатор та назва. Клас є простим контейнером даних (POJO), який забезпечує зручний доступ до інформації про дисципліни в межах системи. Таблиця 3.3. Основні компоненти класу Поле Тип Опис courseID int Унікальний ідентифікатор дисципліни courseName String Назва курсу (наприклад,

Цитирования: **0,01%** id: **47**

«Програмування»,

Цитирования: **0,01%** id: **48**

«Математика»

) Таблиця 3.4. Методи класу Метод Тип повернення Опис getCourseID() int Повертає ідентифікатор дисципліни getCourseName() String Повертає назву дисципліни toString() String Повертає текстове представлення дисциплінт — лише її назву. Застосовується при виведенні об’єкта у GUI або логах. Опис класу CoursesDeleter Загальний опис Клас CoursesDeleter є службовим компонентом системи

Цитирования: **0,01%** id: **49**

«Електронний деканат»

і використовується для видалення дисиплін із бази даних. Його основна функція — реалізація логіки видалення запису дисципліни за назвою (CourseName) з таблиці Courses. Клас працює з базою даних через JDBC-підключення. Змн. Арк. № докум. Підпис Дата Арк. 32 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Таблиця 3.5. Основні компоненти класу Компонент Тип Опис deleteCourse(String CourseName) boolean Метод, який виконує SQL-запит на видалення дисципліни за назвою. Повертає true, якщо курс видалено успішно, або false, якщо курс не знайдено чи сталася помилка. Алгоритм роботи методу deleteCourse(String CourseName) Встановлюється з’єднання з базою даних за допомогою db_dekanat.getConnection(). Формується SQL-запит: DELETE FROM Courses WHERE CourseName = ? Використовується PreparedStatement для безпечної передачі параметру. Виконується команда executeUpdate() і перевіряється кількість змінених рядків (Якщо rowsAffected 0 — дисципліну успішно видалено, Якщо rowsAffected == 0 — дисципліну не знайдено). У разі помилки виводиться повідомлення SQLException. Опис класу CoursesManager Загальний опис Клас CoursesManager є частиною логіки взаємодії з базою даних у системі

Цитирования: **0,01%** id: **50**

«Електронний деканат».

Він виконує функції управління дисциплінами: додавання нових дисциплін до таблиці Courses, отримання списку викладачів та пошук імені викладача за його ID. Клас використовує JDBC-з’єднання з базою даних для виконання SQL-запитів. Таблиця 3.6. Основні компоненти класу Метод Тип повернення Опис getTeachers() List Teacher Повертає список усіх викладачів з таблиці Teachers addCourse(String courseName, String description, int credits, int teacherID) boolean Додає нову дисципліну до бази даних. Повертає true у разі успіху getTeacherNameById(int teacherID) String Повертає повне ім’я викладача (Ім’я + Прізвище) за його ID Опис методу getTeachers() Змн. Арк. № докум. Підпис Дата Арк. 33 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Формує SQL-запит SELECT TeacherID, FirstName, LastName FROM Teachers. Повертає список об’єктів Teacher зі всіма викладачами. Може використовуватися для заповнення випадаючих списків або

зв’язування курсів з викладачами. Опис методу addCourse(...) Приймає параметри курсу: назва, опис, кредити, ID викладача. Перевіряє наявність викладача в базі (через getTeacherNameById). Виконує INSERT INTO Courses(...) для додавання нового запису. Повертає true, якщо запис додано успішно. Опис методу getTeacherNameById(int teacherID) Пошуковий метод, який виконує SQL-запит до таблиці Teachers за TeacherID. Формує повне ім’я викладача у форматі Ім’я Прізвище. Опис класу CoursesTable Загальний опис Клас CoursesTable реалізує функціональність отримання інформації про всі курси з бази даних системи

Цитирования: **0,01%** id: **51**

«Електронний деканат».

Він виконує запит до таблиці Courses та формує масив об’єктів Object[][], який може бути використаний для подальшого відображення дисциплін у графічному інтерфейсі або для передавання на клієнтську частину. Алгоритм роботи методу coursesTab() Встановлюється з’єднання з базою даних за допомогою db_dekanat.getConnection(). Формується SQL-запит: SELECT * FROM Courses. Змн. Арк. № докум. Підпис Дата Арк. 34 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Виконується запит через Statement та зчитується ResultSet. Для кожного запису зчитуються дані: CourseName, Description, Credits, TeacherID. Записи зберігаються у двовимірному масиві Object[][] з максимальною довжиною 100 рядків (обмеження умовне). Порожні значення (null) замінюються на

Цитирования: **0,01%** id: **52**

«NULL»

для уникнення помилок при подальшій обробці. Таблиця 3.7. Основні компоненти класу Метод Тип повернення Опис coursesTab() Object[][] Повертає таблицю курсів у вигляді масиву об’єктів: courseName, description, credits, teacherID Таблиця 3.8. Структура повертаємого масиву Індекс стовпця Значення Тип 0 Назва дисципліни (courseName) String 1 Опис дисципліни (description) String 2 Кількість кредитів (credits) int 3 Ідентифікатор викладача (teacherID) int Опис класу db_dekanat Змн. Арк. № докум. Підпис Дата Арк. 35 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Загальний опис Клас db_dekanat (див. ДОДАТОК Г Код класу db_dekanat) реалізує функціонал підключення до бази даних системи

Цитирования: **0,01%** id: **53**

«Електронний деканат»

за допомогою JDBC. Він відповідає за встановлення та закриття з’єднання з СУБД MySQL. Клас є утилітним (stat ic utility class) та надає зручний інтер- фейс для отримання Connection в інших класах системи. Алгоритм роботи методу getConnection() Використовує DriverManager.getConnection(...) для встановлення з’єднання з MySQL-сервером. Параметри (URL, USER, PASSWORD) жорстко задані в класі. У разі виникнення SQLException помилка передається викликаючому коду. Алгоритм роботи методу closeConnection(Connection connection) Перевіряє, чи з’єднання не є null. Викликає connection.close() для його завершення. У разі помилки при закритті — виводить повідомлення в консоль. Таблиця 3.9. Основні компоненти класу Компонент Тип Опис URL String Шлях до бази даних (MySQL) USER String

Обнаружен Плагиат: **0,21%** https://hostenko.com/uk/wpcafe/навч... + 3 ресурсов! id: **54**

Ім’я користувача бази даних PASSWORD String Пароль для підключення до БД getConnection() Connection Метод для встановлення з’єднання з базою даних closeConnection(Connection connection) void Метод для безпечного закриття з’єднання з базою даних

Опис класу db_log Загальний опис Змн. Арк. № докум. Підпис Дата Арк. 36 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Клас db_log (див. ДОДАТОК Г Код класу db_log) реалізує функціональність підключення до окремої бази даних mysite системи

Цитирования: **0,01%** id: **55**

«Електронний деканат»,

яка використовується для автентифікації користувачів, збереження логів або інших даних, пов’язаних з логуванням. Клас є утилітним (static utility class) і забезпечує централізований доступ доступ до БД через JDBC. Алгоритм роботи Ініціалізація драйвера MySQL: Через

Class.forName(...) у статичному блоці. Виконується при першому використанні класу. У разі помилки — повідомлення виводиться в консоль. Метод getConnection() Використовує DriverManager.getConnection(URL, USER, PASSWORD) для встановлення з’єднання. Метод closeConnection(Connection connection) Закриває активне з’єднання, якщо воно існує. Обробляє можливі виключення (SQLException). Таблиця 3.10. Основні компоненти класу Компонент Тип Опис URL String Шлях до бази даних mysite USER String Ім’я користувача для підключення PASSWORD String Пароль для підключення getConnection() Connection Метод для створення з’єднання з БД closeConnection(Connection connection) void Метод для завершення з’єднання з БД static блок ініціалізації — Завантаження драйвера MySQL com.mysql.cj.jdbc.Driver Опис класу GradesDeleter Загальний опис Клас GradesDeleter є частиною функціональності керування оцінками в системі

Цитирования: **0,01%** id: **56**

«Електронний деканат».

Його основне завдання — реалізація логіки видалення оцінок студентів з таблиці Grades у базі даних. Змн. Арк. № докум. Підпис Дата Арк. 37 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Перед видаленням оцінки клас перевіряє наявність відповідного студента та курсу. Алгоритм роботи методів isStudentExists(studentID) Виконує SQL-запит SELECT COUNT(*) FROM Students WHERE StudentID = ?. Повертає true, якщо студент існує, інакше — false. isCourseExists(courseID) Аналогічно перевіряє існування дисципліни в таблиці Courses. deleteStudentByID(studentID, courseID) Спочатку перевіряє існування студента та курсу. Якщо обидва існують — виконує SQL-запит: DELETE FROM Grades WHERE StudentID = ? AND CourseID = ? Повертає true, якщо запис видалено успішно. Таблиця 3.11. Основні компоненти класу Метод Тип повернення Опис isStudentExists(int studentID) boolean Перевіряє, чи існує здобувач освіти із заданим ID у таблиці Students isCourseExists(int courseID) boolean Перевіряє, чи існує дисципліна із заданим ID у таблиці Courses deleteStudentByID(int studentID, int courseID) boolean Видаляє запис оцінки здобувача з таблиці Grades за відповідними ID Опис класу GradesManager Загальний опис Клас GradesManager відповідає за взаємодію з таблицями здобувачів освіти та дисциплін в базі даних, а також за додавання здобувачів доакадемічних дисциплін та ведення записів про їх бали. Змн. Арк. № докум. Підпис Дата Арк. 38 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Клас також перевіряє наявність освітян та дисципліни за їхніми ідентифікаторами або іменами перед виконанням операцій. Алгоритм роботи методів getStudents() Виконує SQL-запит для отримання всіх здобувачів з таблиці Students. Повертає список здобувачів, де кожен освітянин є об'єктом класу Students. getCourses() Виконує SQL-запит для отримання всіх дисциплін з таблиці Courses. Повертає список дисциплін, де кожена дисципліна є об'єктом класу Courses. addStcu(studentID, courseID) Перевіряє існування здобувача та дисціипліни за допомогою методів getStudentNameById() та getCourseNameById(). Якщо освітянин тадисципліна існують, виконується SQL-запит для додавання запису про здобувача в таблицю Grades. Повертає true, якщо запис додано успішно, інакше — false. addGrade(studentID, courseID, grade) Перевіряє існування здобувача та дисціипліни за допомогою методів getStudentNameById() та getCourseNameById(). Якщо освітянин тадисципліна існують, виконується SQL-запит для додавання балів в таблицю Grades. Повертає true, якщо оцінка додана успішно, інакше — false. getCourseNameById(courseID) Виконує SQL-запит для отримання назви дисципліни за її ID. Змн. Арк. № докум. Підпис Дата Арк. 39 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Повертає назву дисципліни або порожній рядок, якщо дисципліна не знайдена. getStudentNameById(studentID) Виконує SQL-запит для отримання повного імені здобувача освіти за його ID. Повертає повне ім'я освітянина або порожній рядок, якщо здобувача не знайдено. getStudentIdByName(studentName) Виконує SQL-запит для отримання ID здобувача за його повним ім'ям. Повертає ID здобувача або -1, якщо здобувач не знайдений. getCourseIdByName(courseName) Виконує SQL-запит для отримання ID дисципліни за її назвою. Повертає ID дисципліни або -1, якщо дисципліна не знайдена. Таблиця 3.12. Основні компоненти класу Метод Тип повернення Опис getStudents() List Students Повертає список всіх здобувачів освіти з таблиці Students. getCourses() List Courses Повертає список всіх дисциплін з таблиці Courses. addStcu(int studentID, int courseID) boolean Додає здобувача до дисципліни за його ID. addGrade(int studentID, int courseID, int grade) boolean Додає бали освітянину з відповідної дисципліни. getCourseNameById(int courseID) String Повертає назву дисціипліни за її ID. etStudentNameById(int studentID) String Повертає повне ім'я та прізвище здобувача освіти за його ID

Обнаружен Плагиат: **0,18%** https://ela.kpi.ua/bitstreams/1e5f796... + 3 ресурсов! id: **57**

Змн. Арк. № докум. Підпис Дата Арк. 40 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Продовження таблиці 3.12. Змн. Арк. № докум. Підпис Дата Арк.

41 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ getStudentIdByName(String studentName) int Повертає ID освітянина за його повним ім'ям. getCourseIdByName(String courseName) int Повертає ID дисципліни за її назвою. Опис класу GradesTable Загальний опис Клас GradesTable відповідає за отримання даних балів здобувачів з таблиці Grades бази даних. Він виконує SQL-запит для вибірки всіх записів із цієї таблиці, що містять інформацію про здобувачів, дисципліни та їхні бали, і формує двовимірний масив для подальшого використання. Таблиця 3.13. Основні компоненти класу Метод Тип повернення Опис gradesTab() Object[][] Виконує SQL-запит до таблиці Grades, отримує всі записи та формує двовимірний масив з результатами. Алгоритм роботи методу gradesTab() Виконується підключення до бази даних за допомогою методу getConnection(). Виконується SQL-запит: SELECT * FROM Grades. Для кожного результату запиту отримуються значення колонок StudentID, CourseID, та Grade, дані додаються до масиву gradesData. Повертається масив gradesData, що містить всі отримані записи. Опис класу GroupAdder Загальний опис Клас GroupAdder відповідає за додавання нових груп до таблиці Groupe в базі даних. Він виконує SQL-запит для вставки нових записів, що містять інформацію про назву групи, спеціальність та рік зарахування, в базу даних. Змн. Арк. № докум. Підпис Дата Арк. 42 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Таблиця 3.14. Основні компоненти класу Метод Тип повернення Опис addGroup() boolean Додає нову групу до таблиці Groupe за допомогою SQL-запиту. Повертає true, якщо групу успішно додано, інакше — false. Алгоритм роботи методу addGroup() Виконується підготовлений SQL-запит для додавання нового запису до таблиці Groupe. Параметри запиту заповнюються переданими значеннями (groupName — назва групи, speciality — спеціальність, enrollmentYear — рік зарахування). Якщо вставка запису в таблицю успішна (кількість вставлених рядків більше 0), метод повертає true, інакше — false. Опис класу GroupDeleter Загальний опис Клас GroupDeleter відповідає за видалення груп із таблиці Groupe в базі даних. Він виконує SQL-запит для видалення групи за її назвою, перевіряючи наявність такої групи перед виконанням операції. Алгоритм роботи методу deleteGR() Виконується підготовлений SQL-запит для видалення групи за назвою з таблиці Groupe. Параметр запиту заповнюється значенням GroupName, яке передається методу. Якщо видалення групи з таблиці успішне (кількість змін у таблиці більше 0), метод повертає true. Якщо ж групи з такою назвою не знайдено, виводиться повідомлення про помилку, а метод повертає false. Таблиця 3.15. Основні компоненти класу Метод Тип повернення Опис deleteGR() boolean Видаляє групу з таблиці Groupe за її назвою. Повертає true, якщо групу успішно видалено, інакше — false. Опис класу Groups Загальний опис Клас Groups використовується для представлення інформації про групу в системі

Цитирования: **0,01%** id: **58**

«Електронний деканат».

Він містить дані про ідентифікатор групи та її назву. Цей клас дозволяє зберігати та отримувати інформацію про групи освітян, що є частиною системи керування навчальним процесом. Таблиця 3.16. Основні компоненти класу Метод Тип повернення Опис getGroupID() int Повертає ідентифікатор групи. getGroupName() String Повертає назву групи. toString() String Повертає назву групи у вигляді рядка (перевизначення методу toString() для зручності). Алгоритм роботи методів Конструктор класу приймає два параметри: groupID та groupName, і ініціалізує відповідні поля. Змн. Арк. № докум. Підпис Дата Арк. 43 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Метод getGroupName() повертає значення назви групи. Метод toString() перевизначено для того, щоб при виклику цього методу об'єкт класу відображався як назва групи. Опис класу groupTable Загальний опис Клас groupTable використовується для отримання даних про групи з бази даних та представлення їх у вигляді двовимірного масиву, що підходить для подальшого використання в таблицях або інтерфейсах користувача. Він здійснює з'єднання з базою даних, виконує SQL-запит для отримання інформації про групи та повертає результат у форматі, зручному для подальшої обробки. Таблиця 3.17. Основні компоненти класу Метод Тип повернення Опис groupTab() Object[][] Повертає двовимірний масив, який містить дані про групи. Для кожної групи зберігається її назва, спеціальність та рік набору. Алгоритм роботи методу groupTab() Метод створює двовимірний масив groupData розміром 100x3, що

містить інформацію про групи: назву групи, спеціальність і рік набору. Виконується SQL-запит для вибірки всіх груп з таблиці Groupe.Для кожного рядка результату запиту (отримуються значення для GroupName, Specialty та enrollmentYear; якщо значення не є NULL або порожнім, воно додається в масив; якщо значення є NULL, то додається рядок

Цитирования: **0,01%** id: **59**

«NULL»

). Змн. Арк. № докум. Підпис Дата Арк. 44 ITC.4KI.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Після отримання всіх даних результат виводиться в консоль (загальний підсумок по кількості груп). Метод повертає двовимірний масив groupData, який можна використовувати для відображення даних на інтерфейсі користувача або іншій обробці. Опис класу StGrDeleter Загальний опис Клас StGrDeleter використовується для перевірки існування здобувачів та груп у базі даних, а також для видалення зв’язку між здобувачем та групою в таблиці StudentGroups. Він забезпечує правильну логіку видалення освітян із відповідних груп, гарантуючи цілісність даних та мінімізуючи помилки. Перед видаленням виконується перевірка наявності здобувача та групи в базі даних за їхніми ідентифікаторами. Таблиця 3.18. Основні компоненти класу Метод Тип повернення Опис isStudentExists() boolean Перевіряє, чи існує здобувач в таблиці Students за StudentID. isGroupExists() boolean Перевіряє, чи існує група в таблиці Groupe за GroupID. deleteStudentByID() boolean Видаляє запис про належність здобувача до групи у таблиці StudentGroups за StudentID та GroupID. Повертає true, якщо видалення було успішним. Алгоритм роботи методу deleteStudentByID() Виконується перевірка, чи існує здобувач з вказаним StudentID у таблиці Students. Якщо здобувач існує, перевіряється наявність групи з вказаним GroupID у таблиці Groupe. Змн. Арк. № докум. Підпис Дата Арк. 45 ITC.4KI.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Якщо обидві перевірки успішні, виконується SQL-запит на видалення запису з таблиці StudentGroups, де StudentID і GroupID відповідають заданим. У разі успішного видалення виводиться повідомлення про успішне завершення операції. Метод повертає логічне значення true або false залежно від результату операції. Опис класу StGrManager Загальний опис Клас StGrManager реалізує функціонал керування здобувачами освіти та академічними групами в інформаційній системі

Цитирования: **0,01%** id: **60**

«Електронний деканат».

Основні задачі класу — отримання списків здобувачів освіти та груп, додавання здобувача освіти до групи, а також пошук імен або ідентифікаторів за відповідними даними. Клас забезпечує зв’язок між таблицями Students (здобувачі освіти), Groupe (академічні групи) та StudentGroups у базі даних, дозволяючи інтегрувати ці дані для подальшого використання в інтерфейсі користувача або інших модулях системи. Алгоритм роботи: З'єднання з базою даних через клас db_dekanat. Отримання даних про здобувачів освіти або групи за допомогою SQL-запитів. Перевірка наявності записів перед виконанням дій (наприклад, перед додаванням у групу). Змн. Арк. № докум. Підпис Дата Арк. 46 ITC.4KI.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Додавання зв’язку між здобувачем освіти та групою в таблиці StudentGroups. Таблиця 3.19. Основні компоненти класу Метод Тип повернення Опис getStudents() List Students Повертає список здобувачів освіти з бази даних. getGroups() List Groups Повертає список академічних груп з бази даних. addStGr(studentID, groupID) boolean Додає здобувача освіти до групи за відповідними ідентифікаторами. getStudentNameById(id) String Повертає прізвище та ім’я здобувача освіти за його ID. getGroupNameById(id) String Повертає назву групи за її ID. getStudentIdByName(name) int Повертає ID здобувача освіти за його прізвищем та ім’ям. getGroupIdByName(name) int Повертає ID групи за її назвою. Змн. Арк. № докум. Підпис Дата Арк. 47 ITC.4KI.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Опис класу StudentAdder Загальний опис Клас StudentAdder призначений для додавання нових здобувачів освіти до бази даних інформаційної системи

Цитирования: **0,01%** id: **61**

«Електронний деканат».

Він реалізує функціонал вставки нового запису до таблиці Students, зберігаючи основну інформацію про здобувача освіти: ім’я, прізвище, електронну пошту, номер телефону, дату народження та рік вступу. Алгоритм роботи методу addStudent() Формується SQL-запит для

вставки даних у таблицю Students. Використовується підготовлений запит (PreparedStatement), щоб запобігти SQL-ін’єкціям. Параметри заповнюються значеннями, переданими до методу. Виконується вставка нового запису в базу даних. У випадку успішної вставки виводиться повідомлення в консоль. Якщо виникає помилка — відображається відповідне повідомлення про помилку SQL. Метод повертає true, якщо запис було додано, або false — якщо виникла помилка чи запис не було вставлено. Таблиця 3.20. Основні компоненти класу Метод Тип повернення Опис addStudent(firstName, lastName, email, phone, birthDate, enrollmentYear) boolean Додає нового здобувача освіти до таблиці Students. Повертає true, якщо додавання успішне, і false — у разі помилки або невдалої спроби вставки. Опис класу StudentDeleter Загальний опис Клас StudentDeleter призначений для реалізації функціоналу видалення здобувачів освіти з бази даних у межах інформаційної системи

Цитирования: **0,01%** id: **62**

«Електронний деканат».

Він дозволяє здійснювати вилучення записів з таблиці Students за вказаними критеріями — електронною адресою або унікальним ідентифікатором (ID) здобувача освіти. Алгоритм роботи методів: Підключення до бази даних через db_dekanat.getConnection(). Формується SQL-запит на видалення запису з таблиці Students відповідно до параметра (за електронною адресою (email) — метод deleteStudent; за унікальним ідентифікатором (ID) — метод deleteStudentById). Змн. Арк. № докум. Підпис Дата Арк. 48 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Підготовка запиту (PreparedStatement) для запобігання SQL-ін’єкціям. Виконується запит і перевіряється кількість змінених рядків (rowsAffected). Якщо хоча б один запис було видалено, система виводить повідомлення про успішну операцію. Якщо записів не знайдено — повідомлення про відсутність відповідного здобувача освіти. Метод повертає true, якщо видалення відбулося, або false — у разі невдачі чи помилки. Таблиця 3.21. Основні компоненти класу Метод Тип повернення Опис deleteStudent(email) boolean Видаляє здобувача освіти з бази даних за вказаною електронною поштою. Повертає true, якщо запис успішно видалено. deleteStudentById(studentId) boolean Видаляє здобувача освіти з бази даних за його ID. Повертає true, якщо запис успішно видалено. Опис класу StudentGroupsManager Загальний опис Клас StudentGroupsManager призначений для керування зв’язками між здобувачами освіти та академічними групами в інформаційній системі

Цитирования: **0,01%** id: **63**

«Електронний деканат».

Він реалізує функціонал отримання списків здобувачів та груп, а також додавання записів до таблиці StudentGroups, що відображає належність здобувачів до певних академічних груп. Змн. Арк. № докум. Підпис Дата Арк. 49 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Алгоритм роботи основних методів Метод addStGroup1(studentID): Отримує прізвище та ім’я здобувача освіти за переданим ID. Формує SQL-запит для вставки StudentID та GroupID у таблицю StudentGroups. Використовує PreparedStatement для захисту від SQL-ін’єкцій. Виконує запит. Виводить повідомлення про успіх або помилку. Повертає true, якщо вставку здійснено успішно. Метод addStGroup2(groupID) — Аналогічний до addStGroup1, але вставляє лише GroupID. Таблиця 3.22. Основні компоненти класу Метод Тип повернення Опис getStudents() List Students Повертає список усіх здобувачів освіти з таблиці Students. getStudentsNameById(studentID) String Повертає ім’я та прізвище здобувача освіти за його ідентифікатором. addStGroup1(studentID) boolean Додає запис до таблиці StudentGroups, пов’язуючи здобувача з групою. Повертає true, якщо додавання успішне addStGroup2(groupID) boolean Додає запис до таблиці StudentGroups, додаючи групу (без конкретного здобувача). Повертає true, якщо додавання успішне. getGroupsNameById(groupID) String Повертає назву групи за її ідентифікатором. Grid.getGroups() List Groups Повертає список усіх академічних груп з таблиці Groupe. Опис класу StudentGroupsTable Змн. Арк. № докум. Підпис Дата Арк. 50 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Загальний опис Клас StudentGroupsTable призначений для отримання інформації про здобувачів освіти та їх групи з таблиці StudentGroups бази даних інформаційної системи

Цитирования: **0,01%** id: **64**

«Електронний деканат».

Клас надає метод для отримання всіх записів здобувачів освіти та їхніх груп і зберігає ці дані у двовимірному масиві, що дозволяє зручно працювати з інформацією. Алгоритм роботи методу stgroupsTab() Формується SQL-запит для отримання всіх записів з таблиці StudentGroups. Використовується підготовлений запит (Statement) для виконання запиту. Для кожного запису з результату запиту, значення StudentID та GroupID зберігаються в масиві stgrData. Після обробки всіх записів масив повертається для подальшої роботи. Таблиця 3.23. Основні компоненти класу Метод Тип повернення Опис stgroupsTab() Object[] [] Метод отримує всі записи зі зв'язку між здобувачами освіти та їх групами з таблиці StudentGroups і повертає їх у вигляді двовимірного масиву. Опис класу Students Змн. Арк. № докум. Підпис Дата Арк. 51 ITC.4KI.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Загальний опис Клас Students представляє модель для збереження та обробки даних про здобувачів освіти в інформаційній системі

Цитирования: **0,01%** id: **65**

«Електронний деканат».

Клас включає основні атрибути, такі як ID здобувача освіти, та ім’я. Цей клас використовується для зберігання та передачі даних про здобувачів освіти, а також для їх подальшої обробки в різних частинах системи. Алгоритм роботи методу toString() Метод toString() повертає повне ім'я здобувача освіти у вигляді рядка, складаючи його з імені та прізвища. Таблиця 3.24. Основні компоненти класу Метод Тип повернення Опис getStudentID() int Метод для отримання ID здобувача освіти. getFirstName() String Метод для отримання імені здобувача освіти. getLastName() String Метод для отримання прізвища здобувача освіти. toString() String Метод для отримання повного імені здобувача освіти (ім'я + прізвище) у вигляді рядка. Опис класу StudentTable Загальний опис Клас StudentTable призначений для отримання інформації про здобувачів освіти з таблиці Students бази даних інформаційної системи

Цитирования: **0,01%** id: **66**

«Електронний деканат».

Він надає метод для отримання всіх записів про здобувачів освіти і зберігає їх у двовимірному масиві, що дозволяє зручно працювати з даними. Клас також обробляє можливі значення NULL для кожного з атрибутів здобувачів освіти. Алгоритм роботи методу studtab() Підключення до бази даних за допомогою методу db_dekanat.getConnection(). Формується SQL-запит для отримання всіх записів з таблиці Students. Використовується Statement для виконання запиту та обробки результатів. Перевірка на NULL для кожного значення, що дозволяє уникнути помилок при роботі з даними. Результати запиту додаються до двовимірного масиву studentsData.

Обнаружен Плагиат: **0,21%** https://ela.kpi.ua/bitstreams/1e5f796... + 3 ресурсов! id: **67**

Змн. Арк. № докум. Підпис Дата Арк. 52 ITC.4KI.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Після виконання запиту виводиться кількість отриманих записів у консоль. Змн. Арк. № докум. Підпис Дата Арк.

53 ITC.4KI.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Таблиця 3.25. Основні компоненти класу Метод Тип повернення Опис studtab() Object[][] Метод для отримання всіх записів про здобувачів освіти з таблиці Students і збереження їх у двовимірному масиві. Опис класу Teacher Загальний опис Клас Teacher призначений для зберігання інформації про викладачів у системі

Цитирования: **0,01%** id: **68**

«Електронний деканат».

Він містить основні атрибути викладача: ID, ім'я та прізвище, а також відповідні методи для доступу до цих даних. Клас реалізує метод toString(), що дозволяє представляти викладача у вигляді рядка, що містить його ім'я та прізвище. Алгоритм роботи класу У конструкторі класу ініціалізуються поля: teacherID, firstName та lastName. Метод getTeacherID() повертає ID викладача. Метод getFirstName() повертає ім'я викладача. Метод getLastName() повертає прізвище викладача Перевизначений метод toString() формує рядоку форматі

Цитирования: **0,01%** id: **69**

«ім'я прізвище»,

що використовується, наприклад, у компонентах GUI, таких як JcomboBox. Таблиця 3.26.

Основні компоненти класу Метод Тип повернення Опис getTeacherID() int Метод для отримання ID викладача. Продовження таблиці 3.26. getFirstName() String Метод для отримання імені викладача. getLastName() String Метод для отримання прізвища викладача. toString() String Перевизначений метод toString(), що повертає ім'я та прізвище викладача у вигляді рядка. Опис класу TeachersAdder Загальний опис Клас TeachersAdder призначений для додавання нових викладачів до бази даних інформаційної системи

Цитирования: **0,01%** id: **70**

«Електронний деканат».

Клас містить метод для вставки нового запису в таблицю Teachers, зберігаючи основну інформацію про викладача, таку як ім'я, прізвище, електронну пошту, номер телефону та відділ. Таблиця 3.27. Основні компоненти класу Метод Тип повернення Опис addTeacher(firstName, lastName, email, phone, department) boolean Метод для додавання нового викладача до таблиці Teachers. Повертає true, якщо додавання успішне, і false — у разі помилки або невдалої спроби вставки. Алгоритм роботи методу addTeacher() Формується SQL-запит для вставки даних у таблицю Teachers. Використовується підготовлений запит (PreparedStatement), щоб уникнути SQL-ін’єкцій. Змн. Арк. № докум. Підпис Дата Арк. 54 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Параметри заповнюються значеннями, переданими до методу. Виконується вставка нового запису в базу даних. Якщо кількість вставлених рядків більше 0, виводиться повідомлення про успішне додавання викладача. В іншому випадку — повідомлення про невдачу. Метод повертає true, якщо запис було додано, або false, якщо виникла помилка. Опис класу TeachersDeleter Загальний опис Клас TeachersDeleter призначений для видалення викладачів з бази даних інформаційної системи

Цитирования: **0,01%** id: **71**

«Електронний деканат».

Клас містить метод для видалення викладача з таблиці Teachers за його електронною поштою. Таблиця 3.28. Основні компоненти класу Метод Тип повернення Опис deleteTeacher(email) boolean Метод для видалення викладача з таблиці Teachers за електронною поштою. Повертає true, якщо викладач був успішно видалений, і false — у разі помилки або якщо викладач не знайдений. Алгоритм роботи методу deleteTeacher() Формується SQL-запит для видалення викладача за допомогою його електронної пошти. Змн. Арк. № докум. Підпис Дата Арк. 55 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Використовується підготовлений запит (PreparedStatement), щоб уникнути SQL-ін’єкцій. Параметр запиту заповнюється значенням електронної пошти викладача. Виконується операція видалення в базі даних. Якщо кількість вплинутих рядків більше 0, виводиться повідомлення про успішне видалення викладача. Якщо викладач не знайдений, виводиться відповідне повідомлення. Метод повертає true, якщо викладач був видалений, або false, якщо виникла помилка чи викладач не був знайдений. Опис класу TeachersManager Загальний опис Клас TeachersManager призначений для управління даними викладачів в інформаційній системі

Цитирования: **0,01%** id: **72**

«Електронний деканат».

Клас надає методи для отримання імені та прізвища викладача за його ID, а також для отримання ID викладача за його повним ім'ям. Алгоритм роботи методів getTeacherNameById(teacherId) Формується SQL-запит для отримання імені та прізвища викладача за його ID. Використовується підготовлений запит (PreparedStatement), щоб уникнути SQL-ін’єкцій. Параметр запиту заповнюється ID викладача. Виконується запит, і якщо викладач знайдений, формується та повертається його повне ім'я. Якщо викладача не знайдено, виводиться повідомлення про помилку. getTeacherIdByName(teacherFullName) Змн. Арк. № докум. Підпис Дата Арк. 56 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Формується SQL-запит для отримання ID викладача за його повним ім'ям. Встановлюється параметр запиту як повне ім'я викладача. Виконується запит, і якщо викладач знайдений, повертається його ID. Якщо викладач не знайдений, повертається -1. Таблиця 3.29. Основні компоненти класу Метод Тип повернення Опис getTeacherNameById(teacherId) String Отримує ім'я та прізвище викладача за його ID. Повертає повне ім'я викладача або повідомлення, що викладач не знайдений. getTeacherIdByName(teacherFullName) int Отримує ID викладача за його повним ім'ям.

Повертає ID викладача або -1, якщо викладач не знайдений. Опис класу TeachersTable Загальний опис Клас TeachersTable призначений для отримання інформації про викладачів з таблиці Teachers бази даних інформаційної системи

Цитирования: **0,01%** id: **73**

«Електронний деканат».

Клас надає метод для отримання всіх записів викладачів та зберігає їх у двовимірному масиві, що дозволяє зручно працювати з даними. Алгоритм роботи методу teachtab() Формується SQL-запит для отримання всіх записів з таблиці Teachers. Використовується підготовлений запит (PreparedStatement) для виконання SQL-запиту, щоб уникнути SQL-ін'єкцій. Змн. Арк. № докум. Підпис Дата Арк. 57 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Виконується запит, і результати обробляються за допомогою ResultSet. Для кожного викладача з результату запиту отримуються значення таких полів: ім'я, прізвище, електронна пошта, телефон, відділ. Якщо значення виявляються відсутніми (NULL), замість них вставляється значення

Цитирования: **0,01%** id: **74**

«NULL».

Отримані дані зберігаються в двовимірному масиві teachData. Після обробки всіх записів метод повертає масив даних. Таблиця 3.30. Основні компоненти класу Метод Тип повернення Опис teachtab() Object[][] Отримує дані всіх викладачів з таблиці Teachers і повертає їх у вигляді двовимірного масиву об'єктів. Змн. Арк. № докум. Підпис Дата Арк. 58 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Рис. 3.1. – Графічний інтерфейс запущеного сервера 3.2. Опис клієнтської частини Клієнтська частина системи

Цитирования: **0,01%** id: **75**

«Електронний деканат»

розроблена мовою Java та реалізована у вигляді настільного додатку з графічним інтерфейсом (GUI). Вона забезпечує зручний доступ до функціональності системи, включаючи перегляд курсів, студентів, викладачів, оцінок та управління цими даними. Таблиця класи клієнтської частини системи

Цитирования: **0,01%** id: **76**

«Електронний деканат»

(див. ДОДАТОК Б табл.Б.4.). Опис класу Client Загальний опис Клас Client відповідає за реалізацію клієнтської частини програми, що дозволяє користувачу зареєструватися, взаємодіючи з сервером. Включає графічний інтерфейс користувача (GUI), обробку подій для кнопок входу та реєстрації, а також хешування паролів для безпечної передачі через мережу. Змн. Арк. № докум. Підпис Дата Арк. 59 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Таблиця 3.31. Основні компоненти класу Метод Тип повернення Опис hashPassword(String password) String Метод для хешування пароля за допомогою алгоритму SHA-256. main(String[] args) void Головний метод програми, що ініціалізує графічний інтерфейс і обробляє події для кнопок входу та реєстрації. isLoggedIn() boolean Метод для перевірки, чи користувач успішно увійшов до системи. Алгоритм роботи Ініціалізація графічного інтерфейсу У головному методі main створюється вікно з полями для введення email та пароля, а також кнопками для входу та реєстрації. Всі ці елементи додаються до JPanel, а потім — до основного вікна. Обробка події кнопки

Цитирования: **0,01%** id: **77**

«Вхід»

При натисканні на кнопку

Цитирования: **0,01%** id: **78**

«Вхід»,

клієнт: Підключається до сервера, використовуючи методи з класу ServerConnection. Отримує введені користувачем email та пароль. Хешує пароль за допомогою методу hashPassword. Відправляє запит на сервер у форматі LOGIN;email;hashedPassword. Читає відповідь від сервера і в залежності від відповіді або відображає повідомлення про успішний вхід, або про невірний логін/пароль. Якщо вхід успішний, вікно входу закривається і відкривається головне меню. Обробка події кнопки

Цитирования: **0,01%** id: **79**

«Реєстрація»

При натисканні на кнопку

Цитирования: **0,01%** id: **80**

«Реєстрація»,

відкривається нове вікно для реєстрації користувача. Для цього викликається клас RegistrationForm. Метод хешування пароля Метод hashPassword використовує алгоритм SHA-256 для перетворення введеного пароля в хеш, що забезпечує безпеку передачі пароля через мережу. Пароль після хешування передається на сервер. Рис. 3.2 – Форма клієнта Опис класу Configuration Загальний опис Змн. Арк. № докум. Підпис Дата Арк. 60 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Клас Configuration є службовим класом конфігурації, який зберігає константні параметри підключення клієнта до сервера, зокрема IP-адресу сервера (HOST) та порт (PORT), через який здійснюється обмін даними між клієнтом і сервером. Таблиця 3.32. Основні компоненти класу Поле Тип Значення Опис HOST String

Цитирования: **0,03%** id: **81**

«192.168.186.146»

IP-адреса сервера, до якого підключається клієнт. PORT Int 12345 Порт, на якому працює сервер і приймає з'єднання. Опис класу CourseDetailsForm Загальний опис Клас CourseDetailsForm реалізує графічний інтерфейс користувача для перегляду, додавання та управління даними про навчальні дисципліни. Він дозволяє користувачу бачити перелік наявних дисциплін, додавати нові дисципліни та вибирати викладача зі списку. Клас здійснює обмін даними із сервером через сокет-з’єднання, використовуючи стандартні запити у форматі команд. Таблиця Основні компоненти класу (див. ДОДАТОК Б табл.Б.5.). Алгоритм роботи Змн. Арк. № докум. Підпис Дата Арк. 61 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Ініціалізація графічного інтерфейсу В конструкторі CourseDetailsForm створюється вікно з таблицею дисциплін, полями введення для назви дисципліни, опису, кількості кредитів та випадаючим списком викладачів. Всі елементи додаються до панелі з відповідним розташуванням (BorderLayout). Отримання списку викладачів з сервера Метод fetchTeachersDataFromServer надсилає запит GET_TEACHERS на сервер. Сервер повертає JSON-масив викладачів. Кожен викладач додається до випадаючого списку teacherComboBox. Отримання переліку дисциплін з сервера Метод fetchCoursesFromServer надсилає запит GET_COURSES на сервер. Отримана відповідь (JSON-масив дисциплін) додається до таблиці, що відображається користувачу. Додавання нової дисципліни Після натискання кнопки

Цитирования: **0,01%** id: **82**

«Додати дисципліну»,

дані з полів форми зчитуються, формується запит ADD_COURSE;courseName;description;credits; teacherName, який надсилається на сервер. Після відповіді — таблиця оновлюється. Методика взаємодії з сервером Всі запити відправляються через сокет-з’єднання до сервера за адресою Configuration.HOST та портом Configuration.PORT. Команди передаються у вигляді рядків, відповіді — у вигляді рядків або JSON. Рис. 3.3 – Форма для додавання дисциплін в систему Опис класу CoursesForm Загальний опис Змн. Арк. № докум. Підпис Дата Арк. 62 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Клас CoursesForm реалізує графічний інтерфейс користувача для перегляду, оновлення та видалення дисциплін у системі управління навчальним процесом. Клас надає зручний інтерфейс для відображення наявних дисциплін, з можливістю вибору дисципліни для видалення. Взаємодія з сервером здійснюється через сокет-з’єднання, а запити й відповіді обробляються у форматі команд. Таблиця 3.33. Основні компоненти класу Метод Тип повернення Опис CoursesForm() Конструктор Ініціалізує графічний інтерфейс форми, створює таблицю дисциплін, кнопки для оновлення та видалення дисциплін. Продовження таблиці 3.33. fetchCoursesFromServer() void Надсилає запит на сервер для отримання переліку дисциплін та відображає їх у таблиці. sendDeleteRequest(String request) void Надсилає запит на видалення дисципліни на сервер, після чого оновлює таблицю. Алгоритм роботи Змн. Арк. № докум. Підпис Дата Арк. 63 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Ініціалізація графічного інтерфейсу: У конструкторі класу створюється вікно з таблицею

для відображення дисциплін, кнопками для оновлення та видалення дисциплін, а також прогрес-баром для індикації процесів. Всі елементи додаються до панелі із відповідним розташуванням. Отримання переліку дисциплін з сервера: Метод fetchCoursesFromServer надсилає запит GET_COURSES до сервера для отримання списку дисциплін. Отримана відповідь (JSON-масив дисциплін) обробляється та відображається у таблиці. Видалення дисципліни: Користувач може вибрати дисципліну для видалення через таблицю. Після натискання кнопки

Цитирования: **0,01%** id: **83**

«Видалити дисципліну»

формується запит для видалення дисципліни, який надсилається на сервер. Після успішного видалення таблиця оновлюється. Методика взаємодії з сервером Запити та відповіді між клієнтом і сервером здійснюються через сокет-з'єднання на адресу, вказану в Configuration.HOST, та порт Configuration.PORT. Команди передаються у вигляді рядків, а відповіді — у вигляді рядків або JSON-масивів. Опис класу GForm Загальний опис Клас GForm реалізує графічний інтерфейс користувача для перегляду, оновлення та видалення груп у системі управління навчальним процесом. Клас надає зручний інтерфейс для відображення наявних груп, з можливіс- тю вибору групи для видалення. Взаємодія з сервером здійснюється через сокет-з'єднання, а запити й відповіді обробляються у форматі команд. Змн. Арк. № докум. Підпис Дата Арк. 64 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Алгоритм роботи Ініціалізація графічного інтерфейсу: У конструкторі класу створюється вікно з таблицею для відображення груп, кнопками для оновлення та видалення груп, а також прогрес-баром для індикації завантаження даних. Всі елементи додаються до панелі із відповідним розташуванням. Отримання переліку груп з сервера: Метод fetchDataFromServer надсилає запит GET_GROUP до сервера для отримання списку груп. Отримана відповідь (JSON-масив груп) обробляється та відображається у таблиці. Видалення групи: Користувач може вибрати групу для видалення через таблицю. Після натискання кнопки

Цитирования: **0,01%** id: **84**

«Видалити групу»

формується запит для видалення групи, який надсилається на сервер. Після успішного видалення таблиця оновлюється. Таблиця 3.34. Основні компоненти класу Метод Тип повернення Опис GForm() Конструктор Ініціалізує графічний інтерфейс форми, створює таблицю для груп, кнопки для оновлення та видалення груп. Продовження таблиці 3.34. fetchDataFromServer() void

Обнаружен Плагиат: **0,26%** https://www.xn--b1aqmii2j.com/shcho... id: **85**

Надсилає запит на сервер для отримання переліку груп та відображає їх у таблиці. sendDeleteRequest(String request) void Надсилає запит на видалення групи на сервер, після чого оновлює таблицю. Методика взаємодії з сервером Запити та відповіді між клієнтом і сервером

здійснюються через сокет-з'єднання на адресу, вказану в Configuration.HOST, та порт Configuration.PORT. Команди передаються у вигляді рядків, а відповіді — у вигляді рядків або JSON-масивів. Змн. Арк. № докум. Підпис Дата Арк. 65 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Опис класу GradesAdd Загальний опис Клас GradesAdd створює графічний інтерфейс для виставлення балів здобувачам освіти за різними дисциплінами. Клас надає можливість вибору освітянина та дисципліни, введення оцінки та відправлення цих даних на сервер. Окрім того, вікно також відображає таблицю з наявними балами здобувачів за дисциплінами. Дані отримуються та відправляються на сервер через сокет-з'єднання. Алгоритм роботи Ініціалізація графічного інтерфейсу: У конструкторі класу створюється форма з таблицею для відображення балів, випадаючими списками для вибору здобувача та дисципліни, а також полем для введення оцінки. Дані для здобувачів та дисциплін отримуються з сервера при завантаженні форми. Виставлення балів: Користувач вибирає здобувача, дисципліну та вводить оцінку, після чого натискає кнопку для відправлення даних на сервер. Оцінка разом із вибраними даними відправляється на сервер, і після отримання відповіді таблиця оновлюється. Отримання даних з сервера: Дані для здобувачів, дисциплін та балів отримуються за допомогою методів, що надсилають відповідні запити на сервер, а отримані дані відображаються в інтерфейсі. Підсумки відображаються в таблиці. Змн. Арк. № докум. Підпис Дата Арк. 66

ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Методика взаємодії з сервером Запити до сервера надсилаються через сокет-з’єднання на адресу, вказану в Configuration.HOST, та порт Configuration.PORT. Запити формуються як рядки, а відповіді повертаються у вигляді рядків. Для обробки кожної відповіді використовується функціональний інтерфейс ResponseHandler. Таблиця 3.35. Основні компоненти класу Метод Тип повернення Опис GradesAdd() Конструктор Ініціалізує графічний інтерфейс форми для виставлення балів, з таблицею та полями для введення даних. addGrade() void Виставляє бали здобувачу, перевіряючи наявність всіх необхідних даних, та надсилає їх на сервер. fetchStudentDataFromServer() void Отримує список здобувачів з сервера та заповнює випадаючий список для вибору здобувача. fetchGroupDataFromServer() void Отримує список дисциплін з сервера та заповнює випадаючий список для вибору дисципліни. fetchSortedStudentsFromServer() void Отримує бали освітян за дисциплінами з сервера і відображає їх у таблиці. Продовження таблиці 3.35. sendRequestToServer(String, ResponseHandler) void Надсилає запит на сервер та обробляє відповідь через переданий обробник. Рис. 3.4 – Форма для виставлення підсумків в системі Змн. Арк. № докум. Підпис Дата Арк. 67 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Опис класу GradesForm Загальний опис Клас GradesForm реалізує графічний інтерфейс користувача для перегляду та управління балами здобувачів освіти за дисциплінами. Клас дозволяє користувачу побачити таблицю з балами здобувачів освіти, дисциплін та можливістю видаляти бали для конкретного здобувача освіти. Клас здійснює обмін даними із сервером через сокет-з’єднання за допомогою стандартних запитів у форматі команд. Алгоритм роботи Ініціалізація графічного інтерфейсу У конструкторі GradesForm створюється вікно з таблицею для відображення дисциплін, полями для відображення балів здобувачів освіти, прогрес-баром та кнопками для оновлення таблиці та видалення балів. Отримання даних про бали здобувачів освіти з сервера Метод fetchFromServer надсилає запит GET_GRADES на сервер, отримує список балів здобувачів освіти та дисциплін у форматі JSON і заповнює таблицю відповідними даними. Видалення балів здобувача освіти Після натискання кнопки

Цитирования: **0,02%** id: **86**

«Видалити бали студента»

дані з вибраного рядка таблиці передаються на сервер для видалення балів здобувача освіти за відповідною дисципліною. Запит на сервер формується в JSON-форматі. Методика взаємодії з сервером Змн. Арк. № докум. Підпис Дата Арк. 68 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Усі запити відправляються через сокет-з’єднання до сервера за адресою Configuration.HOST та портом Configuration.PORT. Запити передаються у вигляді рядків, відповіді — у вигляді рядків або JSON. Таблиця 3.36. Основні компоненти класу Метод Тип повернення Опис GradesForm() Конструктор Ініціалізує графічний інтерфейс форми, створює таблицю для відображення балів, прогрес-бар, кнопки для оновлення таблиці та видалення балів здобувачів освіти. deletegrades() void Видаляє бали здобувача освіти за вибраною дисципліною з таблиці. sendDeleteRequest(String request) void Надсилає запит на сервер для видалення балів здобувача освіти за дисципліною. fetchFromServer() void Надсилає запит на сервер для отримання даних про бали здобувачів освіти та дисциплін і відображає їх у таблиці. Опис класу GroupDetailsForm Загальний опис Клас GroupDetailsForm реалізує графічний інтерфейс для додавання нових груп та перегляду існуючих груп у системі. Клас забезпечує взаємодію з сервером для отримання та додавання даних про групи здобувачів освіти, відображаючи їх у таблиці, а також надає можливість додавати нові групи через форму введення. Алгоритм роботи Ініціалізація графічного інтерфейсу У конструкторі GroupDetailsForm створюється вікно для введення нової групи, таблиця для відображення груп та прогрес-бар для індикації завантаження даних. Змн. Арк. № докум. Підпис Дата Арк. 69 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Отримання даних про групи з сервера Метод fetchDataFromServer надсилає запит до сервера для отримання даних про групи здобувачів освіти, отримує відповідь у форматі JSON і оновлює таблицю. Таблиця 3.37. Основні компоненти класу Метод Тип повернення Опис GroupDetailsForm() Конструктор Ініціалізує графічний інтерфейс для додавання групи та перегляду даних про групи, налаштовує таблицю та форми введення. sendAddStudentRequest(String groupName, String specialty, int enrollmentYear) void Відправляє запит на сервер для додавання нової групи з відповідними даними. fetchDataFromServer() void Отримує список груп здобувачів освіти з сервера та оновлює таблицю. Отримання даних про групи з сервера Метод fetchDataFromServer надсилає запит до сервера для отримання даних про групи здобувачів освіти, отримує

відповідь у форматі JSON і оновлює таблицю. Додавання нової групи При натисканні кнопки

Цитирования: **0,01%** id: **87**

«Створити групу»,

введені дані (назва групи, спеціальність та рік вступу) відправляються на сервер для додавання нової групи через метод sendAddStudentRequest. Методика взаємодії з сервером Запити відправляються через сокет-з’єднання до сервера за адресою Змн. Арк. № докум. Підпис Дата Арк. 70 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Configuration.HOST і портом Configuration.PORT. Запит на отримання груп містить команду

Цитирования: **0,01%** id: **88**

«GET_GROUP»,

а для додавання групи —

Цитирования: **0,01%** id: **89**

«ADD_GROUP»

із відповідними параметрами. Рис. 3.5 – Форма для додавання груп в системі Опис класу MenuForm Загальний опис Клас MenuForm реалізує головне вікно програми з вкладками для управління різними аспектами системи. У вікні є вкладки для роботи з формамиздобувачів, викладачів, дисциплін, груп, балів і загальних дій (вихід). Кожна вкладка містить набори кнопок, що дозволяють відкривати відповідні форми для взаємодії з даними. Алгоритм роботи Ініціалізація інтерфейсу У конструкторі MenuForm налаштовуються вкладки для роботи з різними формами, такими якздобувачі, викладачі, дисципліни тощо. Для кожної вкладки створюється окрема панель з кнопками, кожна з яких викликає відповідну дію. Відкриття форм Для кожної вкладки на панелі є кнопки, що відкривають різні форми, наприклад, для додавання або перегляду даних проздобувачів, викладачів, дисципліни тощо. Кожна кнопка викликає відповідний метод, який відкриває нову форму. Методика взаємодії з сервером У даному класі немає безпосереднього взаємодії з сервером, але всі виклики методів відкривають відповідні форми для керування даними. Таблиця 3.38. Змн. Арк. № докум. Підпис Дата Арк. 71 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Основні компоненти класу Метод Тип повернення Опис MenuForm() Конструктор Ініціалізує головне вікно з вкладками для різних розділів системи (освітяни, викладачі, дисципліни, групи, оцінки, загальні дії). createPanelWithButtons() JPanel Створює панель з кнопками для певної вкладки, кожна кнопка має свою дію, що викликає відповідну форму. styleButton() void Налаштовує стиль для кнопок, встановлюючи шрифт, кольори та інші параметри. Методи для виклику різних форм void Кожен метод відкриває відповідну форму для роботи з даними (наприклад, openStudentForm() для відкриття форми студентів, addStudentForm() для додавання здобувачів). Рис. 3.6 – Форма меню системи Опис класу RegistrationForm Загальний опис Клас RegistrationForm відповідає за створення форми для реєстрації нового користувача в системі. Форма включає поля для введення імені, прізвища, електронної пошти, пароля, телефону та дати народження. Після заповнення форми та натискання кнопки

Цитирования: **0,01%** id: **90**

«Зареєструватися»,

відбувається перевірка введених даних, хешування пароля та відправка даних на сервер для реєстрації користувача. Змн. Арк. № докум. Підпис Дата Арк. 72 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Таблиця 3.39. Основні компоненти класу Метод Тип повернення Опис RegistrationForm(String host, int port) Конструктор Ініціалізує форму для реєстрації нового користувача, включаючи поля для введення даних та кнопку для відправки інформації на сервер. hashPassword(String password) String Хешує пароль користувача за допомогою алгоритму SHA-256 для забезпечення безпеки пароля перед відправкою на сервер. Алгоритм роботи Ініціалізація інтерфейсу У конструкторі класу створюється форма з полями для введення даних користувача (ім'я, прізвище, email, пароль, телефон, дата народження) та кнопка для відправки цих даних на сервер. Валідація введених даних Після натискання кнопки

Цитирования: **0,01%** id: **91**

«Зареєструватися»

перевіряється, чи заповнені всі поля. Якщо хоча б одне поле порожнє, з'являється

повідомлення про помилку. Також перевіряється формат email за допомогою регулярного виразу. Хешування пароля Для забезпечення безпеки пароля, перед відправкою на сервер, він хешується за допомогою алгоритму SHA-256. Відправка даних на сервер За допомогою сокетів здійснюється з'єднання з сервером. Дані про користувача (ім'я, прізвище, email, хешований пароль, телефон, дата народження) відправляються на сервер у вигляді рядка. Сервер відповідає результатом реєстрації, який відображається користувачу. Методика взаємодії з сервером Дані відправляються на сервер за допомогою сокетів через TCP-з'єднання. Клієнт надсилає запит на реєстрацію у форматі

Цитирования: **0,05%** id: **92**

«REGISTER;name;surname;email;hashedPassword;phone;birthdate»,

а сервер відповідає повідомленням про успіх або помилку. Рис. 3.7 – Форма реєстрації в системі Опис класу ServerConnection Загальний опис Клас ServerConnection відповідає за ініціалізацію

Обнаружен Плагиат: **0,09%** https://fosslook.com.ua/articles/ssl-s... id: **93**

з'єднання з сервером та управління сокетами для обміну даними між клієнтом та сервером.

Цей клас дозволяє підключитися до сервера, отримати потоки вводу/виводу для обміну даними, а також закрити з'єднання. Таблиця 3.40. Змн. Арк. № докум. Підпис Дата Арк. 73 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Основні компоненти класу Метод Тип повернення Опис initialize(String serverHost, int serverPort) void Ініціалізує параметри сервера (адресу та порт), якщо вони ще не були ініціалізовані. getSocket() Socket Отримує активне з'єднання до сервера або створює нове, якщо з'єднання ще не існує. getOut() PrintWriter Отримує об'єкт для виведення даних на сервер (потік виводу). getIn() BufferedReader Отримує об'єкт для вводу даних від сервера (потік вводу). closeConnection() void Закриває з'єднання з сервером, якщо воно відкрите. Алгоритм роботи Ініціалізація параметрів сервера Метод initialize() використовується для налаштування параметрів сервера (хосту та порту). Якщо сервер вже ініціалізований, нові параметри не встановлюються, і виводиться повідомлення про це. Підключення до сервера Метод getSocket() відповідає за підключення до сервера. Якщо з'єднання вже існує, воно використовується повторно. Якщо ж сокет ще не створено або закрито, здійснюється підключення до сервера через задану адресу та порт. Отримання потоку виводу Метод getOut() повертає об'єкт PrintWriter, який використовується для відправки даних на сервер. Потік створюється при підключенні до сервера. Отримання потоку вводу Метод getIn() повертає об'єкт BufferedReader, який використовується Змн. Арк. № докум. Підпис Дата Арк. 74 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ для отримання даних від сервера. Потік також створюється при підкдючен- ні. Закриття з'єднання Метод closeConnection() закриває сокет та звільняє ресурси, якщо з'єднання більше не потрібне. Методика взаємодії з сервером Клас взаємодіє з сервером через сокети. Після отримання активного з'єднання з сервером можна відправляти дані через PrintWriter та отримувати відповіді через BufferedReader. Підключення здійснюється один раз за допомогою методу initialize(), а після цього використовується для обміну даними через відповідні потоки. Опис класу Sform Загальний опис Клас Sform є графічним інтерфейсом для управління даними здобувачів освіти. Він забезпечує відображення таблиці здобувачів, а також дозволяє виконувати операції оновлення таблиці та видалення здобувачів через сервер. Всі операції здійснюються через сокети, і дані передаються у форматі JSON. Таблиця Основні компоненти класу (див. ДОДАТОК Б табл.Б.6.). Алгоритм роботи Ініціалізація форми Конструктор Sform() налаштовує інтерфейс користувача, створюючи таблицю для відображення даних студентів і додаючи кнопки для оновлення таблиці та видалення здобувачів. Прогрес-бар встановлюється невидимим і відображається лише під час завантаження даних. Отримання даних зі сервера Метод fetchDataFromServer() виконується в окремому потоці через SwingWorker і надсилає запит

Цитирования: **0,01%** id: **94**

«GET_STUDENTS»

на сервер. Відповідь, що містить дані здобувачів у форматі JSON, обробляється і виводиться в таблицю. Змн. Арк. № докум. Підпис Дата Арк. 75 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Оновлення таблиці Коли дані про студентів отримано, таблиця очищається, і в неї додаються нові рядки з інформацією, отриманою з сервера. Прогрес-бар

ховається після завершення завантаження. Видалення здобувача Метод sendDeleteRequest() дозволяє видалити здобувача з бази даних за допомогою запиту

Цитирования: **0,01%** id: **95**

«DELETE_STUDENT».

Після успішного видалення таблиця оновлюється, щоб відобразити зміни. Методика взаємодії з сервером Комунікація з сервером відбувається через сокети, використовуючи протокол текстових повідомлень. Для отримання даних використовуються запити типу

Цитирования: **0,01%** id: **96**

«GET_STUDENTS»,

а для видалення освітян —

Цитирования: **0,01%** id: **97**

«DELETE_STUDENT».

Сервер повертає дані у форматі JSON, що дозволяє надійно обробляти інформацію проздобувачів. Опис класу Stgradd Загальний опис Клас Stgradd є графічним інтерфейсом для відсортування здобувачів освіти за групами. Він забезпечує можливість вибору здобувача освіти та групи зі списків, а також додає здобувача освіти до вибраної групи. Інтерфейс використовує сокетні з'єднання для комунікації з сервером, отримуючи дані здобувачів освіти, груп і інформацію про вже відсортованих здобувачів освіти. Алгоритм роботи Ініціалізація форми Конструктор Stgradd() налаштовує інтерфейс користувача, створюючи таблицю для відображення здобувачів освіти, випадаючі списки для вибору здобувача освіти та групи, а також кнопку для додавання здобувача освіти до групи. Змн. Арк. № докум. Підпис Дата Арк. 76 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Прогрес-бар використовується для візуалізації завантаження даних. Отримання даних зі сервера Метод fetchStudentDataFromServer() отримує список здобувачів освіти, fetchGroupDataFromServer() — список груп, а fetchCoursesFromServer() — інформацію про здобувачів освіти, що вже належать до груп. Ці дані отримуються через сокетні з'єднання з сервером у форматі JSON. Вибір здобувача освіти та групи Користувач вибирає здобувача освіти і групу з відповідних списків. При натисканні кнопки

Цитирования: **0,01%** id: **98**

«Відсортувати»,

виконується запит до сервера для додавання здобувача освіти до вибраної групи. Оновлення таблиці Таблиця оновлюється після успішного додавання здобувача освіти до групи або після отримання нових даних від сервера. У таблиці відображаються імена здобувачів освіти та назви груп, до яких вони належать. Методика взаємодії з сервером Взаємодія з сервером здійснюється через сокети. Запити на сервер можуть бути такими: GET_STUDENTS — отримати список здобувачів освіти. GET_GROUP — отримати список груп. GET_STUDGROUP — отримати дані про здобувачів освіти і групи. ADD_STUDENT_TO_GROUP — додати здобувача освіти до групи. Сервер відповідає у форматі JSON, який обробляється і відображається в інтерфейсі користувача. Змн. Арк. № докум. Підпис Дата Арк. 77 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Рис. 3.8 – Форма відсортування здобувачів за академічними групами Опис класу StGrForm Загальний опис Клас StGrForm є графічним інтерфейсом для відображення здобувачів освіти за групами. Він надає можливість вибору здобувача освіти і групи, а також дає можливість видаляти здобувачів освіти з груп. Клас використовує сокетні з'єднання для комунікації з сервером для отримання та оновлення даних. Таблиця 3.41. Основні компоненти класу Метод Тип повернення Опис studentIDs ArrayList Integer Список ідентифікаторів здобувачів освіти, отриманих з сервера. groupIDs ArrayList Integer Список ідентифікаторів груп, отриманих з сервера. Алгоритм роботи Ініціалізація форми Конструктор StGrForm() налаштовує інтерфейс користувача, створюючи таблицю для відображення здобувачів освіти та груп, додаткові кнопки для оновлення таблиці та видалення здобувача освіти з групи. Прогрес-бар використовується для індикації процесу завантаження даних. Отримання даних зі сервера Метод fetchFromServer() отримує список здобувачів освіти, груп та інформацію про вже відсортованих здобувачів освіти за групами через Змн. Арк. № докум. Підпис Дата Арк. 78 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ сокетні з'єднання у форматі JSON. Видалення здобувача освіти з групи Користувач може вибрати здобувача освіти у таблиці і натиснути кнопку для видалення здобувача освіти з групи. Запит на видалення

відправляється на сервер через сокетне з'єднання. Оновлення таблиці Після видалення здобувача освіти з групи або отримання нових даних, таблиця оновлюється з новою інформацією про здобувачів освіти та їх групи. Методика взаємодії з сервером Запити на сервер для взаємодії з даними можуть бути наступними: GET_STUDGROUP — отримати список здобувачів освіти та їх груп. DELETE_STUDENT_FROM_GROUP — видалити здобувача освіти з групи. Сервер відповідає у форматі JSON, який обробляється для відображення результатів на інтерфейсі. Опис класу StudentDetailsForm Загальний опис Клас StudentDetailsForm є графічним інтерфейсом для додавання нового здобувача освіти до системи та перегляду списку існуючих здобувачів освіти. Клас використовує сокетне з'єднання для комунікації з сервером, зокрема для надсилання запитів на додавання студента та отримання списку студентів. Прогрес-бар показує статус завантаження даних або виконання запитів. Алгоритм роботи Ініціалізація форми Конструктор StudentDetailsForm() налаштовує інтерфейс користувача, додаючи таблицю для відображення даних про здобувачів освіти, а також форму для введення нового студента. Змн. Арк. № докум. Підпис Дата Арк. 79 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Прогрес-бар спочатку невидимий і з’являється під час виконання запитів до сервера. Додавання студента Після заповнення форми і натискання кнопки

Цитирования: **0,01%** id: **99**

«Додати студента»,

дані з полів форми відправляються на сервер через сокетне з'єднання за допомогою методу sendAddStudentRequest(). Запит на сервер форматується як рядок і відправляється за допомогою методу PrintWriter. Отримання даних про студентів Метод fetchDataFromServer() надсилає запит на сервер для отримання списку здобувачів освіти у форматі JSON. Отримані дані перетворюються у формат, зручний для відображення в таблиці. Оновлення таблиці Після отримання відповіді від сервера або додавання нового студента, таблиця оновлюється з новими даними. Методика взаємодії з сервером Запити на сервер для взаємодії з даними можуть бути наступними: ADD_STUDENT — додати нового студента. GET_STUDENTS — отримати список здобувачів освіти. Сервер відповідає у форматі JSON, який обробляється для відображення результатів на інтерфейсі. Рис. 3.9 – Форма для додавання даних про здобувачів освіти в систему Змн. Арк. № докум. Підпис Дата Арк. 80 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Опис класу TableStyle Загальний опис Клас TableStyle відповідає за налаштування стилю таблиці та кнопок у графічному інтерфейсі. Це дозволяє забезпечити гармонійний вигляд таблиць із даними та кнопок для взаємодії з користувачем. В ньому реалізовані методи для налаштування фону таблиці, сітки, вирівнювання тексту, а також стилізації кнопок [23]. Таблиця 3.42. Основні компоненти класу Метод Тип повернення Опис applyTableStyle(JTable table) void Налаштовує стилі для таблиці: фон, сітка, заголовки та вирівнювання тексту. styleButton(JButton button) void Налаштовує стилі для кнопок: шрифт, фон, колір тексту та відсутність фокусу. Алгоритм роботи Налаштування таблиці Метод applyTableStyle(JTable table) налаштовує: Фон таблиці: задається світлий фон для таблиці. Сітка таблиці: включається відображення сітки з чорним кольором. Заголовки таблиці: встановлюється світло-блакитний фон, чорний колір тексту та жирний шрифт для заголовків. Вирівнювання тексту: усі клітинки таблиці будуть відображатися з вирівнюванням тексту по центру. Чередування кольорів рядків: для парних і непарних рядків таблиці встановлюються різні фони — білий та світло-блакитний. Вибраний рядок: фон вибраного рядка змінюється на світло-блакитний, а текст — на чорний. Змн. Арк. № докум. Підпис Дата Арк. 81 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Налаштування кнопок Метод styleButton(JButton button) налаштовує: Шрифт: задається жирний шрифт

Цитирования: **0,02%** id: **100**

«Times New Roman»

розміру 14. Фон кнопки: встановлюється темно-синій фон. Колір тексту кнопки: текст на кнопці буде жовтого кольору. Фокусування: вимикається відображення фокусу при натисканні на кнопку. Методика взаємодії з сервером Цей клас не взаємодіє безпосередньо з сервером, але забезпечує естетичний вигляд компонентів інтерфейсу, що полегшує взаємодію з користувачем. Опис класу TeachersDetailsForm Загальний опис Клас TeachersDetailsForm створює графічний інтерфейс для додавання викладачів до системи. Він включає таблицю для відображення списку викладачів, форму для введення нових даних про викладачів, а також прогрес-бар для індикації процесу завантаження даних або

додавання інформації. Взаємодія з сервером здійснюється через сокети, що дозволяє отримувати та надсилати дані. Алгоритм роботи Конструктор TeachersDetailsForm(): Налаштовує інтерфейс користувача з таблицею для відображення викладачів, формою для введення нових даних та прогрес-баром для індикації завантаження. Змн. Арк. № докум. Підпис Дата Арк. 82 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Кнопка

Цитирования: **0,01%** id: **101**

«Додати викладача»

ініціює відправку введених даних на сервер. Метод sendAddTeacherstRequest(...): Створює запит для додавання викладача через сокет-з'єднання з сервером. Оновлює таблицю після отримання відповіді від сервера. Метод fetchDataFromServer(): Надсилає запит на сервер для отримання списку викладачів. Отримує відповідь у форматі JSON, яку перетворює в об'єкти та відображає в таблиці. Методика взаємодії з сервером Для взаємодії з сервером використовується сокетне з'єднання. У методах sendAddTeacherstRequest та fetchDataFromServer створюються сокетні з'єднання з сервером, після чого надсилаються запити (

Цитирования: **0,01%** id: **102**

«ADD_TEACHERS»,

Цитирования: **0,01%** id: **103**

«GET_TEACHERS»

), і отримується відповідь. Відповідь сервера обробляється: у разі отримання даних — вони додаються в таблицю, у разі помилки — виводиться повідомлення про помилку. Таблиця 3.43. Основні компоненти класу Метод Тип повернення Опис TeachersDetailsForm() Конструктор Конструктор, який ініціалізує інтерфейс форми для додавання викладача. sendAddTeacherstRequest(...) void Надсилає запит на сервер для додавання викладача до бази даних. fetchDataFromServer() void Завантажує список викладачів з сервера та відображає його в таблиці.

Обнаружен Плагиат: **0,15%** https://ela.kpi.ua/bitstreams/1e5f796... + 3 ресурсов! id: **104**

Змн. Арк. № докум. Підпис Дата Арк. 83 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Змн. Арк. № докум. Підпис Дата Арк.

84 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Рис.3.10 – Форма для додавання даних про викладачів в систему Опис класу Tform Загальний опис Клас Tform створює графічний інтерфейс для управління викладачами в системі. Він включає таблицю для відображення списку викладачів, кнопки для оновлення даних та видалення викладача з бази даних. Взаємодія з сервером здійснюється через сокети для запитів щодо викладачів. Таблиця 3.44. Основні компоненти класу Метод Тип повернення Опис Tform() Конструктор Конструктор, що ініціалізує інтерфейс для роботи з викладачами. fetchDataFromServer() void Завантажує дані викладачів із сервера та відображає їх у таблиці. sendDeleteRequest(String) void Надсилає запит на сервер для видалення викладача за його email. Алгоритм роботи Конструктор Tform(): Ініціалізує вікно з таблицею для відображення викладачів, кнопками для оновлення даних та видалення викладачів. При натисканні кнопки

Цитирования: **0,01%** id: **105**

«Оновити таблицю»

відправляється запит для завантаження нових даних викладачів. При натисканні кнопки

Цитирования: **0,01%** id: **106**

«Видалити викладача»

видаляється вибраний викладач із системи за допомогою його email. Метод fetchDataFromServer() Відправляє запит на сервер для отримання списку викладачів. Отримує відповідь у форматі JSON, яку перетворює в об'єкти та відображає у таблиці. Метод sendDeleteRequest(String): Відправляє запит на сервер для видалення викладача, використовуючи його email. Після успішного видалення перезавантажує таблицю з оновленими даними. Методика взаємодії з сервером Для взаємодії з сервером використовується сокетне з'єднання. У методах fetchDataFromServer та sendDeleteRequest створюються сокетні з'єднання для надсилання запитів на сервер (запит на отримання

викладачів:

Цитирования: **0,01%** id: **107**

«GET_TEACHERS»

; запит на видалення викладача:

Цитирования: **0,01%** id: **108**

«DELETE_TEACHERS;email»

). Додатково були розроблені, два універсальні класи захисту паролів які працюють як на серверній так і на клієнтській частинах. Це класи SecureEncryption який потім зібраний в jar файл та в подальшому використовується як бібліотека в проєкті, та клас ConfigReader. Опис класу SecureEncryption Змн. Арк. № докум. Підпис Дата Арк. 85 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Загальний опис Клас SecureEncryption забезпечує функціональність симетричного шифрування та дешифрування текстових даних з використанням алгоритму AES у режимі CBC. Ключ генерується на основі пароля користувача та випадкової

Цитирования: **0,01%** id: **109**

«солі»

за допомогою алгоритму PBKDF2WithHmacSHA256. Клас забезпечує надійний захист даних за допомогою сучасних криптографічних стандартів. Таблиця Основні компоненти класу (див. ДОДАТОК Б табл.Б.7.). Алгоритм роботи Конструктор SecureEncryption() Ініціалізує об’єкт класу без жодних параметрів. У даній реалізації не виконує додаткових дій. Метод generateKeyFromPassword(String password, byte[] salt) Використовує алгоритм PBKDF2WithHmacSHA256 для генерації ключа. На основі пароля та солі створюється 256-бітний ключ AES. Метод encryptAES(String data, String password) Генерується випадкова сіль (16 байт) і IV (ініціалізаційний вектор, 16 байт). Генерується ключ на основі пароля та солі. Дані шифруються з використанням AES у режимі CBC. У результаті формується один масив із солі + IV + зашифрованих даних, який кодується у формат Base64 для передачі або збереження. Метод decryptAES(String encryptedData, String password) Декодує дані з Base64 у байтовий масив. Виділяє сіль, IV та зашифроване повідомлення. Змн. Арк. № докум. Підпис Дата Арк. 86 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Генерує ключ за тією ж методикою. Дешифрує дані та повертає оригінальний текст. Методика взаємодії з даними Шифрування відбувається локально, без взаємодії з сервером або мережею. Клас не залежить від зовнішніх систем і може використовуватись у будь-яких клієнт-серверних або автономних системах для забезпечення захисту даних (наприклад, паролів). Опис класу ConfigReader Загальний опис Клас ConfigReader відповідає за зчитування конфігураційних параметрів із зовнішнього файлу config.properties. Основною метою є отримання значення головного криптографічного ключа MASTER_KEY, який використовується у системі для шифрування або інших захищених операцій. Таблиця 3.45. Основні компоненти класу Метод Тип повернення Опис getMasterKey() String Зчитує значення MASTER_KEY із конфігураційного файлу та повертає його. У разі помилки повертає null Алгоритм роботи Метод getMasterKey() Створюється об’єкт Properties для зчитування параметрів. Відкривається файл config.properties, шлях до якого вказано у змінній CONFIG_FILE. Змн. Арк. № докум. Підпис Дата Арк. 87 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Завантажуються властивості з файлу. Проводиться пошук параметра MASTER_KEY. Якщо параметр знайдено — метод повертає його значення. Якщо параметра не знайдено або виникла помилка зчитування файлу — виводиться повідомлення про помилку і повертається null. Методика взаємодії з системою Клас працює автономно та не взаємодіє з мережею чи сервером. Основна роль — забезпечити централізоване зчитування конфігураційного параметра (MASTER_KEY), що використовується іншими класами системи, наприклад, у класі SecureEncryption. Залежність: клас очікує, що файл config.properties міститиме ключ MASTER_KEY у форматі: MASTER_KEY=ваш_ключ. Змн. Арк. № докум. Підпис Дата Арк. 88 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ Висновки до розділу 3 В третьому розділі було здійснено повну реалізацію клієнт-серверної інформаційної системи

Цитирования: **0,01%** id: **110**

«Електронний деканат»,

яка охоплює всі основні аспекти розробки програмного забезпечення: серверну частину,

клієнтську частину та їхню взаємодію. По-перше, реалізовано серверну частину системи на мові Java з використанням архітектури TCP-сервер – клієнт. Сервер забезпечує прийом і обробку запитів, а також взаємодіє з базами даних для збереження, оновлення та вилучення інформації. Розроблений графічний інтерфейс адміністратора дозволяє зручно керувати роботою сервера та переглядати активні підключення користувачів. По-друге, розроблена клієнтська частина реалізована у вигляді десктопного додатку з дружнім графічним інтерфейсом, що дозволяє користувачам реєструватися, входити до системи, переглядати та редагувати інформацію про студентів, викладачів, дисципліни, групи та оцінки. По-третє, реалізовано ефективну взаємодію між клієнтом і сервером за допомогою протоколу TCP та обміну командами у форматі текстових запитів або JSON. Усі компоненти системи забезпечують логічну цілісність, узгодженість даних і стабільну роботу в умовах багатокористувацького середовища. Таким чином, реалізація системи

Цитирования: **0,01%** id: **111**

«Електронний деканат»

демонструє можливості побудови сучасного програмного забезпечення для автоматизації навчального процесу з використанням Java та баз даних MySQL, забезпечуючи ефективне управління освітнім процесом. Змн. Арк. № докум. Підпис Дата Арк. 89 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ ВИСНОВКИ В ході виконання даної бакалаврської роботи було здійснено розробку клієнт-серверної системи

Цитирования: **0,01%** id: **112**

«Електронний деканат»

засобами Java, що забезпечує автоматизацію управління академічним процесом у закладах вищої освіти. Перший розділ містив аналіз предметної області, визначення вимог до системи та вибір технологій для її реалізації. Було розглянуто основні проблеми існуючих систем управління обліком здобувачів освіти та запропоновано підхід для їх усунення шляхом створення централізованої інформаційної системи на основі клієнт-серверної архітектури. Було обґрунтовано вибір мови програмування, бази даних та фреймворків, які забезпечують надійну та ефективну роботу системи. Другий розділ був присвячений процесу проектування системи, включаючи визначення архітектури клієнт-серверної взаємодії, структури бази даних, проектування

Обнаружен Плагиат: **0,26%** https://openarchive.nure.ua/bitstream... id: **113**

серверної та клієнтської частин. Було розроблено UML-діаграми для відображення зв’язків між основними компонентами системи, а також описано механізми автентифікації користувачів, обробки запитів та управління даними. У третьому розділі розглянуто безпосередню реалізацію системи. Детально описано структуру серверної та клієнтської частин,

механізм обробки запитів клієнта, взаємодію з базою даних та реалізацію заходів безпеки. Була впроваджена система авторизації користувачів, що гарантує безпечний доступ до інформації, а також оптимізовано запити до бази даних для підвищення продуктивності системи. Змн. Арк. № докум. Підпис Дата Арк. 90 ІТС.4КІ.0124.03-ПЗ Розроб. Курячий О.В. Керівник Переяславська С.О. Реценз. Козуб Ю.Г. Н. Контр. Зав. каф. Семенов М.А. ВИСНОВКИ Літ. Акрушів 2 ЛНУ Кафедра ІТС, Гр.4КІ . В результаті виконаної роботи була створена повнофункціональна клієнт-серверна система

Цитирования: **0,01%** id: **114**

«Електронний деканат»,

яка дозволяє автоматизувати управління академічними процесами, зменшити адміністративне навантаження та підвищити ефективність роботи викладачів, здобувачів освіти та адміністрації навчального закладу. Реалізована система відповідає сучасним вимогам інформаційної безпеки та зручності використання, що робить її придатною для впровадження у реальних навчальних закладах. Змн. Арк. № докум. Підпис Дата Арк. 91 ІТС.4КІ.0124.03-ПЗ НАЗВА ДОКУМЕНТУ НАЗВА ДОКУМЕНТУ

Заявление об ограничении ответственности:

Этот отчет должен быть правильно истолкован и проанализирован квалифицированным специалистом, который несет ответственность за оценку!

Любая информация, представленная в этом отчете, не является окончательной и подлежит ручному просмотру и анализу.
Пожалуйста, следуйте инструкциям: Рекомендации по оценке

Детектор Плагиата - Ваше право на оригинальность! □ SkyLine LLC