Хмельницький національний університет
Факультет програмування та комп'ютерних і телекомунікаційних систем
Кафедра комп'ютерних наук та інформаційних технологій

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему Автоматизований електронний органайзер роботи викладачів навчальних курсів

Галузь знань *12 – Інформаційні технології*
Шифр і назва галузі знань

Спеціальність *122 – Комп'ютерні науки*
Шифр і назва спеціальності

Освітня програма *Комп'ютерні науки*
Назва освітньої програми

Виконала: *студентка 4 курсу, група КН-17-1* — Підпис — *А.А. Палій*
Курс, група виконавця; Ініціали, прізвище

Керівник: *к.т.н., доцент кафедри КНІТ* — Підпис — *О.В. Мазурець*
Науковий ступінь, посада; Ініціали, прізвище

Нормоконтроль: *к.т.н., доцент кафедри КНІТ* — Підпис — *Р.О. Багрій*
Науковий ступінь, посада; Ініціали, прізвище

До захисту допускаю:
Зав. кафедри КНІТ, д.т.н., професор — Підпис — *О.В. Бармак*
Ініціали, прізвище

08 червня 2021 р.

Хмельницький 2021

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет *програмування та комп'ютерних і телекомунікаційних систем*
Кафедра *комп'ютерних наук та інформаційних технологій*
Освітній ступінь *бакалавр*
Галузь знань *12 – Інформаційні технології*
Спеціальність *122 – Комп'ютерні науки*

ЗАТВЕРДЖУЮ
Завідувач кафедри *комп'ютерних наук та інформаційних технологій*

(підпис)
д.т.н., професор О.В. Бармак
«*08*» *травня* *2021* року

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА

1. Тема кваліфікаційної роботи бакалавра: «Автоматизований електронний органайзер роботи викладачів навчальних курсів»

2. Завдання видано студентці *Палій Анастасії Анатоліївній*
(прізвище, ім'я, по батькові)

3. Керівник роботи *доцент кафедри КНІТ Мазурець Олександр Вікторович*
(посада, прізвище, ім'я, по батькові)

4. Затверджено наказом університету від «*05*» *травня* *2021* р. № *11*

5. Зміст пояснювальної записки (перелік задач) та вихідні дані:

Мета роботи – розробити автоматизований електронний органайзер роботи викладачів навчальних курсів, який надасть змогу автоматизувати роботу викладачів та забезпечити відповідний контроль. Для цього необхідно забезпечити доступ до бази даних, відображення її компонентів, можливість їх редагування та видалення, реалізувати вхід у систему під різними ролями. Слід автоматизувати функції викладача (проведення занять, заміна інших викладачів, введення даних в систему) та адміністратора (перегляд наявних даних, формування навчальних груп, реєстрація нових студентів, працівників, ведення робочих розкладів.

Виконавець: *студентка 4 курсу, група КН-17-1* (Курс, група виконавця) _____ (Підпис) *А.А. Палій* (Ініціали, прізвище)

Керівник: *к.т.н., доцент кафедри КНІТ* (Науковий ступінь, посада) _____ (Підпис) *О.В. Мазурець* (Ініціали, прізвище)

Анотація

Тема кваліфікаційної роботи бакалавра: *«Автоматизований органайзер роботи викладачів навчальних курсів»*

Виконавець кваліфікаційної роботи бакалавра: *студентка групи КН-17-1 Палій Анастасія Анатоліївна*

Керівник кваліфікаційної роботи бакалавра: *к.т.н., доцент кафедри КНІТ Мазурець Олександр Вікторович*

Кваліфікаційна робота бакалавра містить:

Пояснювальна записка				Кількість додатків
Сторінок	Рисунків	Таблиць	Джерел інформації	
56	29	9	42	3

Метою кваліфікаційної роботи бакалавра є розробка автоматизованого електронного органайзера роботи викладачів навчальних курсів на платформі Android. Для розробки інформаційної системи було використано програмне середовище AndroidStudio, мову програмування Java, систему керування базами даних MySQL і графічний клієнт для роботи з сервером MySQL Workbench.

Розроблена система призначена для адміністраторів навчальних центрів, де проводиться надання інформаційних послуг клієнтам. Напрямками практичного використання даної програми є контроль виконання викладачами службових обов'язків: проведення занять, заміна інших викладачів, введення даних в систему та функціонал адміністратора, який може продивлятися наявну інформацію, формувати навчальні групи, реєструвати нових студентів, працівників, робочі розклади.

Ключові слова: інформаційні технології, навчальні курси, навчальні групи, додаткова освіта, органайзер роботи.

Виконавець: *студентка 4 курсу, група КН-17-1* _____ *А.А. Палій*

Курс, група виконавця | Підпис | Ініціали, прізвище

Зміст

Перелік скорочень

Скорочення, термін, позначення	Пояснення
АП	Асинхронні потоки
БД	База даних
ВНЗ	Вищий навчальний заклад
ІС	Інформаційна система
ІСР	Інтегроване середовище розробки
ІТ	Інформаційні технології
КН	Комп’ютерні науки
КНІТ	Комп’ютерні науки та інформаційні технології
КРБ	Кваліфікаційна робота бакалавра
ОП	Оперативна пам’ять
ОС	Операційна система
ПП	Програмний продукт
ПС	Програмне середовище
СКБД	Система керування базами даних
ХНУ	Хмельницький національний університет
IDE	Integrated development environment
JDK	Java Development Kit
JVM	Java Virtual Machine
MS	Microsoft
OS	Operation System
SQL	Structure Query Language

Вступ

З початком 21 століття – віку ІТ, стало популярним серед загалу прагнення до пізнання нового і попит на спеціалістів широкого спектру. Окрім самостійного опрацювання матеріалу в Інтернеті, є ще багато способів отримання необхідних вмінь та навичок для майбутніх працівників: ВНЗ, семінари, індивідуальні заняття та навчальні курси [1]. У кожного з цих видів є свої аспекти, пропоновані плани, переваги та недоліки (наприклад, вікові обмеження). І на шляху до отримання майбутньої бажаної професії, для поглиблення своїх здібностей, зараз стало актуальним проходити навчальні курси. Після завершення їх, слухачам вручається спеціальний документ, що свідчить про те, що дана особа отримала деякі теоретичні знання та практичні навички в певній сфері [2].

Установи, що займаються освітою, виділяються такими видами працівників: викладачі (основна робоча одиниця, надає інформаційні послуги), адміністратори (корегує роботу перших, вирішує організаційні питання відповідно до інструкцій). Завдання викладача – забезпечити необхідною інформацією слухачів курсів відповідно плану.

Викладач, в загальному розумінні, зараз грає не таку роль, як колись. На даний момент педагогічний потенціал спрямований на отримання якомога більшого результату різними способами. Освіта це не тільки суха теорія і практика по звіднику правил, в деякій мірі це творчий процес. Тому, щоб викладачі могли виконувати свою роботу, спрямовувати свій потенціал в необхідний напрямок, бажано перекласти роботу з запам’ятовуванням важливої вже пройденої організаційної інформації на когось іншого, а бажано на пристрій, так як він є більш надійним, ніж людський фактор [3].

Щоб розуміти обсяги запланованої роботи, спрогнозувати можливий спектр розвитку для фірми необхідна система, яка буде з одного боку контролювати роботу одних, а з іншої – надавати у швидкому та зручному режимі потрібну інформацію.

Саме для цього установам іноді необхідно мати «автоматизований електронний органайзер роботи викладачів навчальних курсів», де вони будуть зазначати необхідну інформацію. Тоді уповноважена особа (така як адміністратор, старший викладач, керівник навчального складу) зможе звіритись з цими даними і на базі них побудувати майбутній план роботи без необхідності уточняти цю інформацію і збирати її усюди.

В загальній сутності органайзер є паперовою записною книжкою, куди власник її може зазначати важливу інформацію, стосовно планів, організації роботи, тощо [4]. Електронний аналог повторює ті ж самі функції, але знаходиться на пристрої, тобто це не дає змогу забути цю річ і завжди мати потрібну інформацію у швидкому доступі, до того ж, записник часто є більш особистою річчю і для великих фірм він не підходить, так як не має чітко визначену структуру, а для мобільних девайсів програма з таким функціоналом найкраще підходить, так як збільшує мобільність та зручність використання. Це спрощує, пришвидшує і надає перевагу серед тих навчальних курсів, де такої системи не існує. Відповідно актуальною є розробка системи, яка б забезпечувала автоматизацію функцій проведення занять, заміни інших викладачів, введення даних в систему і функціонал адміністратора, що може продивлятися наявну формувати навчальні групи, інформацію, реєструвати працівників, нових студентів, вести робочі розклади.

Розділ 1

Характеристика предметної області та постановка задачі

1.1 Аналіз предметної області

З боку ринкових відносин – ринок навчальних закладів є системою обігу купівлі-продажу послуг навчального характеру. Говорячи про освіту, як суб'єкт ринкової економіки, то на цьому ринку повинні бути продавець, покупець та товар. Це є механізмом, за допомогою яких освітня послуга реалізовується в економічній сфері.

Ринок освітніх послуг забезпечує споживача водночас товаром та послугою, продавцями ринку освітніх послуг є навчальні заклади у вигляді:

– дошкільних закладів (ясла, дитячі садки, установи розвитку дітей дошкільного віку);

– шкільні та позашкільні заклади (школа, ліцей, колегіум, гімназія, школа-інтернат, а також клуби, *курси*, табори, студії мистецтв, спортивні заклади);

– професійно-технічні(училища, комбінати);

– вищі (університет, коледж, академія);

– післядипломна і додаткова освіта (академія, *курси* підвищення кваліфікації, так звані, клуби, семінари).

Навчальний заклад є установою, що на регулярній основі здійснює освітній процес з метою навчання з боку отримання певних навичок та вмінь, а також розвиток певних якостей в людині. Освітній заклад є ланкою системи освіти та інституційною основою педагогіки.

Створення навчального закладу здійснюється з такими початковими факторами: потребою в освітній діяльності на певній території, а з іншої — наявністю необхідної матеріально-технічної, науково-методичної бази, педагогічних кадрів.

Завдання навчальних закладів полягають у всебічному розвитку людини, розвитку її талантів, розумових і фізичних здібностей, вихованні високих

моральних якостей, забезпеченні народного господарства кваліфікованими фахівцями [5].

Заклади ще між собою можна поділити на вікові категорії: перша – дитячий сегмент споживачів (контрагентом-платником виступають зазвичай батьки або держава); друга – дорослий сегмент (контрагентом може виступати і споживач, і родичі, і друзі, і держава, не є обов'язковим для проходження але бажаним з соціальної та правової точки зору, також часто виступає необхідним критерієм для отримання високооплачуваної роботи); змішані – заклади відносяться до більш загального характеру і так само, як і попередньо представлений варіант є необов'язковими для проходження. В останньому варіанті групи навчання все-одно часто класифікуються на «дитячі» та «дорослі», іноді між цими варіаціями бувають більш конкретні вікові рамки.

Товаром на ринку освітніх послуг є знання, вміння і навички, запропоновані суб'єктами цього ринку (ВНЗ, приватними викладачами, училищами, коледжами тощо). Угоди між суб'єктами, які пропонують ці послуги у сфері освіти, та його споживачами здійснюються у вигляді обміну. Вочевидь, що на даний час угоди здійснюються так: продавець освітньої послуги навчає чомусь споживача і підкріплює отриману освіту дипломом, посвідченням, атестатом, а споживач вносить плату за навчання, або за нього це робить, наприклад, держава [6].

Організованою, двосторонньою діяльністю, спрямовану на максимальне засвоєння та усвідомлення навчального матеріалу і подальшого застосування отриманих знань, умінь та навичок на практиці називають навчанням. В оперативно-діяльнісний компонент навчання входять форми організації навчання. До загальних форм відносять індивідуальну, парну, групову і колективну [7].

У навчальній системі, окрім взаємодії споживача (або контрагента) з навчальним закладом є ще важливий фактор взаємодії споживачів та викладачів. Цей процес є головною структурною одиницею цілеспрямованої й організованої

роботи з метою передачі необхідного теоретичного матеріалу та практичного досвіду, інколи ще й соціального виховання [8].

Необхідно розібрати особу, що здійснює навчання: є педагог, викладач, вчитель – за принципом роботи вони виконують близькі функції, тому далеко не всі дійсно знають принципову різницю між ними. Незважаючи на це, вона існує і відображає різну професійну діяльність.

Учитель – спеціальність, яку набувають випускники педагогічних інститутів і середніх спеціальних навчальних закладів для роботи в шкільних закладах. Отримуючи диплом, вони стають основною ланкою навчально-виховної системи. Викладачі – випускники університетів, яка дає право особі, яка отримала відповідний диплом, займатися науково-викладацькою діяльністю в області своєї спеціалізації.

Викладач – кваліфікація, яка присвоюється випускникам навчальних закладів, що мають статус університету чи академії. Учитель – педагогічна спеціальність. Мета викладацької діяльності – надання наукової та методичної інформації. Учитель навчає учнів предмету і формує у них навички самостійної навчальної роботи [9].

Можна ввести ще такий вид поняття лектора як «спеціаліст» [10]. Зазвичай це особа без професійної освіти викладача, педагога або вчителя, але з необхідними знаннями та навичками у певній сфері. Інколи до багажу знань додається ще диплом, сертифікат про отримання освітніх послуг в навчальному закладі. Тобто, це людина, яка знає і вміє, як виконувати певний вид роботи, з цієї сторони вона може ділитись своїм досвідом та знаннями зі слухачами. Так як в обов'язки викладача входить надання освітніх послуг, перш за все, цій особі потрібно самій розбиратись у тому, що вона вчить. Тому інколи в навчальних закладах додаткової освіти, вимоги до опанування людини певної галузі по пріоритетності прийому на роботу, стають вищими, ніж академічна освіта викладача.

В даній темі використовується така робоча одиниця, як викладач, а не педагог, тому що останній, у свою чергу, більше зосереджений на формуванні

особистості і використовується лише у сфері освітніх послуг дитячого виховання.

Сучасні дослідники розглядають процес викладання як динамічну систему, спрямовану на вирішення розвивальних і освітніх завдань, отримання певних професійних навиків або здобутків; засвоєння перелічених аспектів учнями, студентами, слухачами.

Сутність надання викладачем освітніх послуг зазвичай об'єднує поняття теоретичних знань і практичних навичок. Саме цим обумовлюються його функції:

– навчальна (формування мотивації, досвіду навчально-пізнавальної та практичної діяльності, засвоєння основ наукових знань, ціннісних орієнтацій особистості, перевірка засвоєного матеріалу);

– практична (демонстрація створення продукту, пояснення на всіх стадіях виробництва, відтворення його слухачами під наглядом досвідченого професіоналу, узагальнення отримання навичок) [10].

Тобто, згідно отриманої вище інформації, заняття можна ділити за такими факторами як: теоретичні або практичні, самостійні, в ході отримання яких, людина отримує необхідні навички та уміння.

Освітні організації, що реалізують додаткові освітні програми:

1) організація додаткової освіти – освітня організація, що здійснює як основну мету її діяльності освітню діяльність за додатковими загальноосвітніми програмами;

2) організація додаткової професійної освіти – освітня організація, що здійснює як основну мету її діяльності освітню діяльність за додатковими професійними програмами [11].

Отже основними параметрами предметної області для предметної області організації роботи викладачів навчальних курсів є: Студент, Викладач та Навчальна група (так як представляють собою таблиці, що мають взаємозв’язки з усіма іншими таблицями) (таблиця 1.1).

Таблиця 1.1 – Параметри предметної області

Параметр предметної області	**Опис**
Студент	Сутність «Студент» уявляє собою особу, яка відвідує курс навчання.
Викладач	Сутність «Викладач» уявляє собою особу, яка виконує свою роботу і отримає за неї гроші.
Навчальна група	Сутність «Навчальна група» уявляє собою клас студентів, що сумісно проходить відповідно до поставленого плану заняття за певним розкладом.
Заняття	Сутність «Заняття» являє собою відповідність реального ходу подій відповідно поставленого плану в «Навчальній групі». В багато чому ці дві таблиці мають сходитись.
Розклад занять	Сутність «Розклад занять» представляє собою різні розклади для навчальних груп.
Відвідування студентом заняття	Сутність «Відвідування студентом заняття» пов’язує таблицю «Студент» та «Заняття» за допомогою атрибута «Присутність».
Навчання студента в групі	Сутність «Навчання студента в групі» пов’язує таблицю «Студент» і «Навчальна група» для того, щоб відмітити навчання 1 студента в 1 групі.
Дисципліна	Сутність «Дисципліна» містить інформацію про наявні дисципліни в даному навчальному закладі, відповідно яких формуються навчальні групи.
Аудиторія	Сутність «Аудиторія» містить інформацію про наявні аудиторії, в яких проводяться заняття в навчальних групах (зв’язок з 2 таблицями «Навчальна група» і «Заняття»)

Отже, система для ролей «адміністратор» і «викладач» має виконувати: роботу з даними викладачів, учнів; долучення студентів до їх навчальних курсів; побудову планів навчання і відповідність їх реальному стану речей; створення, видалення, редагування, перегляд інформації щодо навчальних груп, курсів, аудиторій, розкладів тощо; ведення звітів відвідуваності учнів на заняттях.

1.2 Аналіз інформаційного забезпечення предметної області

Попередній розділ містив розгляд аналіз теми КРБ та цілі навчальних курсів. Тепер необхідно розібратися ще з наявним програмним забезпеченням зі схожим функціоналом.

Для організації роботи навчальних закладів існують CRM-системи. Зазвичай це ПП у вигляді вебсайту. Розглядаючи їх, можна виділити 2 фактори: в інформаційної системи зазвичай або складний функціонал або відсутній аналог для мобільних систем. Говорячи про впровадження в роботу підприємства таких вебсайтів, для деяких ПП потрібно додатково витрачати час адміністраторів на те, щоб вони навчились ним користуватись. Інколи процес вивчення функціоналу може займати навіть місяці або вивчатись на додаткових курсах, так як освоєння певного продукту може бути надважкою задачею для рядового користувача. Тобто є певний взаємозв'язок – чим складніша система сама по собі, тим більше шансів, що в неї є аналогічний додаток (який для даної задачі є дійсно необхідністю, тому що додає мобільності і економить час в контролюванні процесів, побудові планів і звітуванні).

Одним з прикладом CRM-систем є вебсайт «Мій клас» – рисунок 1.1 [12].

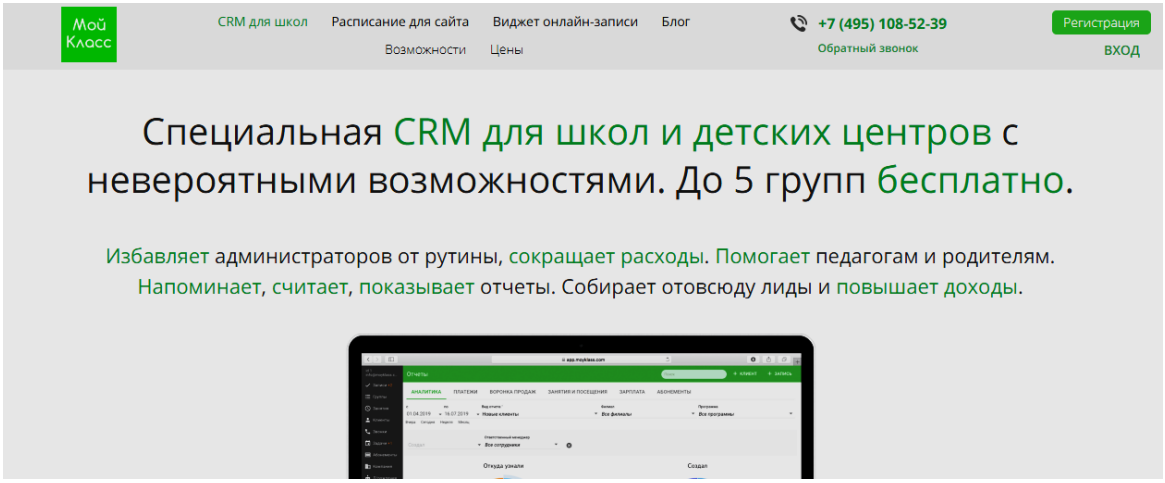


Рисунок 1.1 – Головна сторінка CRM-системи «Мій класс» [12]

Дизайн вебсайту простий і стильний. В загальних рисах, «Мой класс» використовує функціонал, який є і на інших подібних системах, його доволі багато і розібратися в ньому не є проблемою.

По своїй структурі він схожий на електронний аналог певних журналів або певних офісних декстопних варіантів офісних додатків.

Дане ПЗ створене для роботи навчальних закладів, тобто орієнтація є на цього користувача. (рисунок 1.2)

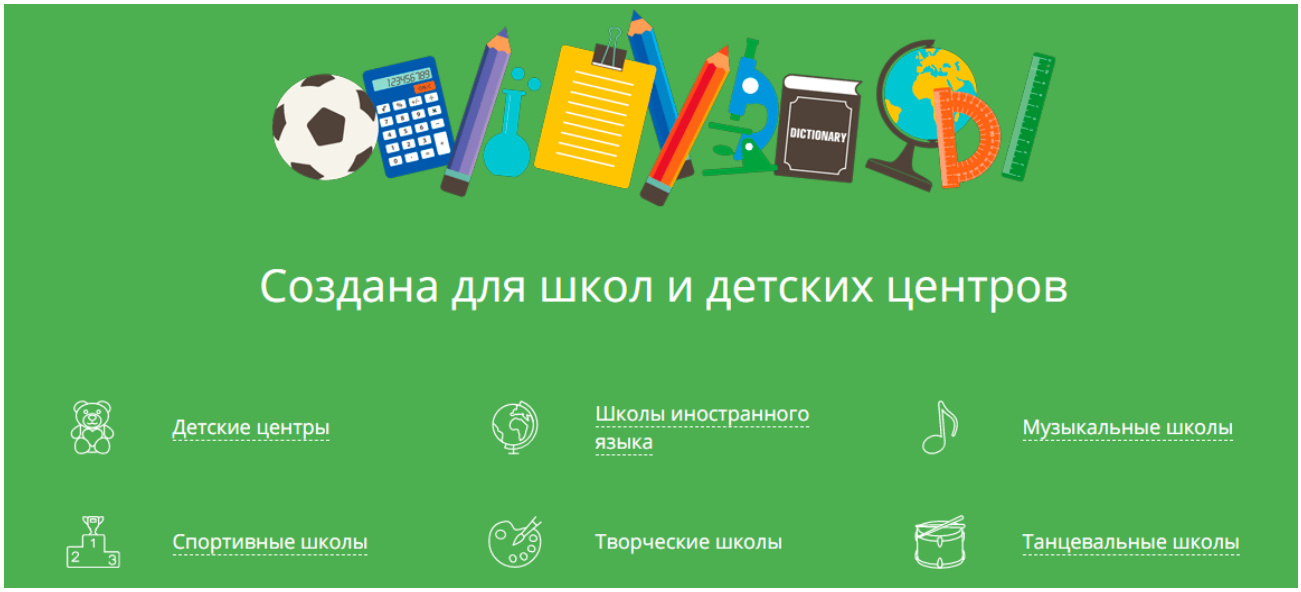


Рисунок 1.2 – Види напрямків, для яких створений веб-сайт [12]

На рисунку 1.3 продемонстрована адміністраторська панель з наявними модулями: навчальні групи, ліди, заняття, клієнти, задачі, побудови звітів, Дні народження, розсилки, зарплати викладачів, тощо.

Функціонал даного вебсайту є доволі суттєвим і нескладним у вивченні, але з недоліків даної системи хотілось би виділити відсутність впровадження даного продукту під мобільні пристрої. Є ще такий аспект, як відсутність певних функцій для певних індивідуальних випадків, але це б ускладнило саму систему або навантажило б її, що привело би до збоїв роботи, так що його можна не враховувати.

Тепер розглянемо «HOLLIHOP» CRM-систему, яка додатково ще має мобільний додаток – рисунок 1.4 [13].

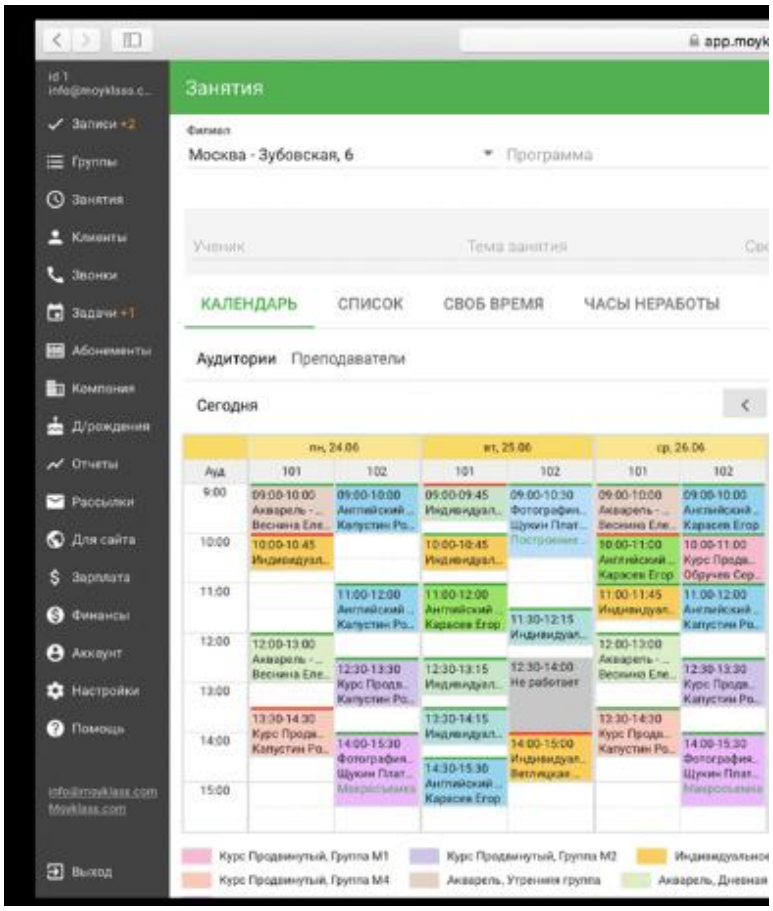


Рисунок 1.3 – Адміністраторська панель вебсайту «Мій клас» [12]

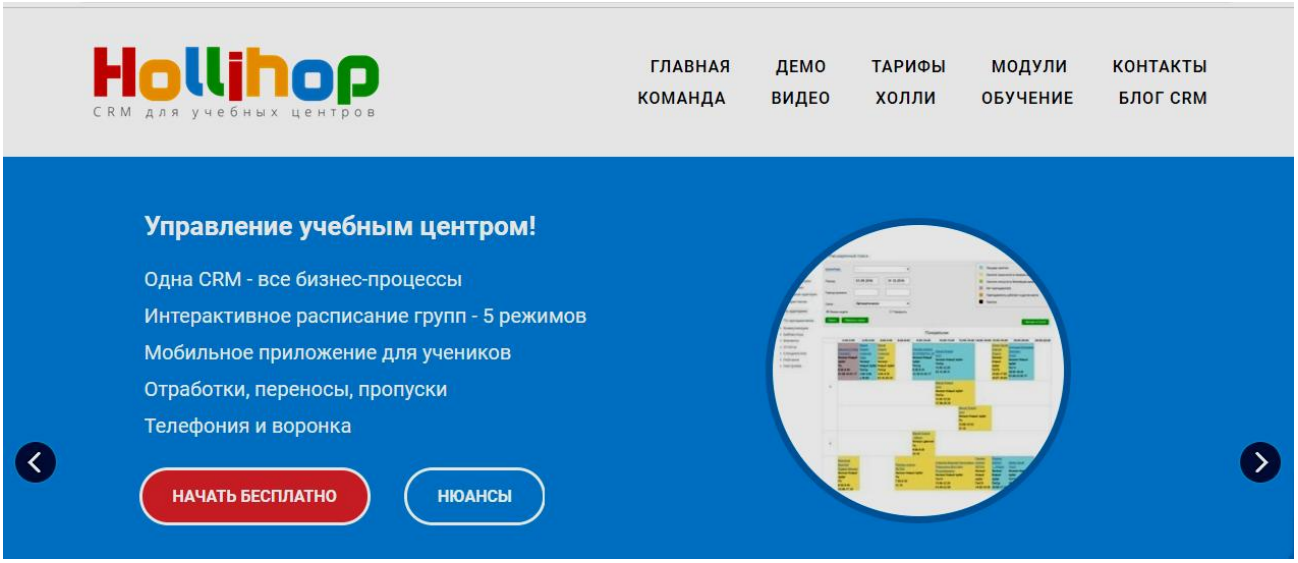


Рисунок 1.4 – Головна сторінка вебсайту «HOLLIHOP» [13]

Якщо порівнювати з попередньою системою, дизайн сайту є більш яскравим та цікавим.

Функціонал роботи системи ділиться на 4 основних модулі (рисунок 1.5).

Рисунок 1.5 – Основні модулі для роботи «HOLLIHOP» [13]

Незважаючи на те, що основних модулів, здавалось би, чотири, але кожних з них включає в себе ще по 10-15 модулів для роботи, що створює таку незручність при роботи, як «важкість» осягнення самої системи. Тобто вебсайт не є простим у вивченні і для того, щоб вільно ним користуватись, потрібно додатково розбиратись в налаштуванні. А це призводить у свою чергу до нових помилок у вигляді: неправильно сформованих даних запитів і додаткове витрачання часу на впровадження під свою наявну організацію навчання.

Стосовно мобільного додатку – це дуже спрощена система для особистого кабінету для клієнтів. Але незважаючи на те, що дана система не включає в себе дуже багато функціоналу, вона є зручною в сучасних реаліях.

Загалом даний вебсайт порівняно з недоліками має велике число переваг у вигляді: тех-підтримки, великої кількості впроваджених функцій і ненавантаженої системи при цьому.

Окрім того, «HOLLIHOP» постійно оновлюється додає у свою структуру все більше нових опцій (в тому числі особистий кабінет для учнів навчального закладу – відносно нова технологія для них).

Підсумовуючи вищесказане, хотілось знову би звернути увагу, що для поставленого завдання існують необхідні системи, й кожна, так чи інакше, володіє своїми перевагами і недоліками. Як основний фактор хотілось би виділити, що майже всі системи створені лише на веб-платформі. А ті, які мають мобільні додатки – до цієї переваги у своєму основному ПП також включають

такі недоліки як: складність системи для рядового користувача або перевантаженість системи, некоректно працюють, тому що є певні функції, що ускладнюють створення тих чи інакших модулів. А також додається ще такий аспект – що подібні системи напрямлені переважно на російські навчальні заклади, а для українського користувача ПП подібного рівня не існує, а зважаючи на наявні відмінності потреба в таких системах суттєва.

1.3 Аналіз сучасних засобів створення програмного забезпечення

Java є досить затребованою мовою програмування, зокрема зазнала великої популярності серед розробників додатків під систему Android. З часу свого створення, мова розвивалась і продовжує розвиватись, завдяки чому Java-девелопери мають широкий спектр створення різноманітних додатків, ігор та інструментів [14]. Допоміжною мовою для оформлення зовнішнього вигляду додатків буде використовуватись XML [15].

Згідно зі статистикою, опублікованою в Datareportal, 67% дорослих людей в усьому світі використовують смартфони щодня, а це майже 5,19 млрд осіб (при загальній кількості населення в 7,75 млрд) [16].

За останні роки було помічено тенденцію збільшення використання мобільних гаджетів порівняно з персональними комп’ютерами. Хіба що в плані вимог ринку, web-розробка також набирає оборотів, але темпи і попит на розробників мобільних додатків до сих пір ще зростає, а ринок ще не насичений мобільними розробниками. Це говорить про те, що Mobile- програмування є досить актуальним зараз, до того ж можна цілком затвердити, що вимога у програмному додатку для смартфонів і планшетів є необхідністю, тому що саме ця вибрана платформа зможе пришвидшити і упростити виконання поставленої задачі з «Автоматизації системи» [17].

В цілому, головним завданням розробника є створення зручного мобільного додатку, який буде безвідмовно працювати, а також буде інтуїтивно

зрозумілим, корисним і багатофункціональним для кінцевого користувача, і, звичайно ж, прибутковим для замовника.

Більшість мобільних розробників можна розділити на дві категорії, в залежності від програмного забезпечення, для якого вони створюють програми – iOS розробники і Android-розробники. Фахівці з першої категорії вважаються найбільш прибутковими на ринку праці, більш того, після появи мов Swift і Objective-C створення додатків для Apple стало дуже легким і зручним [18]. Програмісти, які створюють програми для Android, використовують у своїй діяльності мову Java, яка вважається найнадійнішою для розробки мобільних додатків для цієї операційної системи. Окрім вищезазначених , можна назвати ще такі: Windows Phone, Java ME, Symbian, але їх частка на ринку досить таки невелика, порівняно з «гігантами» Android і iOS [19].

Для створення ПП під систему Android нам потрібно Інтегроване середовище розробки(IDE), яка повинно містити в собі:

– редактор для запису коду;

– компілятор і відладчик;

– бажаним також обрати те середовище, яке буде містити різні шаблони, функцію підкреслення, виправлення коду і т.д.

Так само, система ще може містити в собі різні форми впроваджень коду до різних версій ОС (т.я. регулярне оновлення версії є однією з особливостей мобільного програмування). Також бажано, щоб система була зручною для використання при об’єктно-орієнтованої розробки ПЗ. Прикладами середовищ розробки мобільних додатків є:

– Android Studio;

– NetBeans;

– Eclipse;

– Qt Creator;

– Embarcadero RAD Studio;

– Microsoft Visual Studio;

Більшість з них є середовищами для розробки різного типу додатків та під різні мови і при програмуванні на базі Android, знадобиться встановлювати необхідні плагіни, налаштовувати софт під свої потреби [20].

AndroidStudio є вбудованим SE для розробки мобільних пристроїв на основі IntelliJ IDEA, який включає в себе засоби для розробки додатків для портативних пристроїв на базі Android (смарт-часи, окуляри, автоаудіосистеми), спеціальний плагін для імпорту проекту, зробленого в Eclipse в саме середовище, засоби для тестування під різні версії ОС Android, емулятор (спеціальна віртуальна машина, яка включає в себе деякий функціонал портативних гаджетів) (рисунок 1.6).

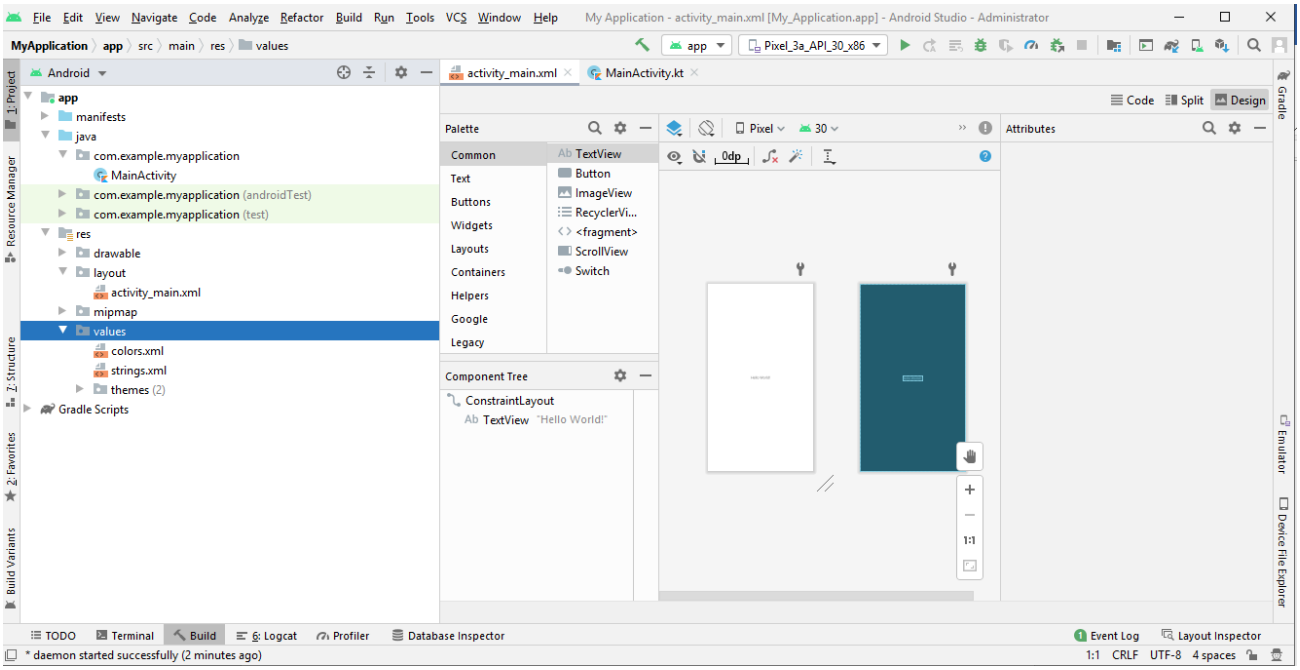


Рисунок 1.6 – Робота програми Android Studio [21]

На поточний момент актуальною є версія 4.

Переваги даного ПЗ:

– середовище надає достатньо широкий арсенал інструментів, який одночасно підлаштовується під вимоги розробника, надає певні шаблони та функції тестування і є досить гнучким для забезпечення базових і спеціальних вимог девелопера;

– дозволяє створювати додатки не тільки для смартфонів і планшетів, а й для інших портативних пристроїв;

– розробка додатку для Android N – найостаннішої версії операційної системи.

Недоліки також є:

– вимогливість студії до продуктивності апаратної основи ПК, на якому повинно здійснюватись тестування

– неможливість написати серверні проєкти на мові Java для ПК [22].

IDE Eclipse – середовище розробки під Android, що базується на Eclipse. (рисунок 1.7).

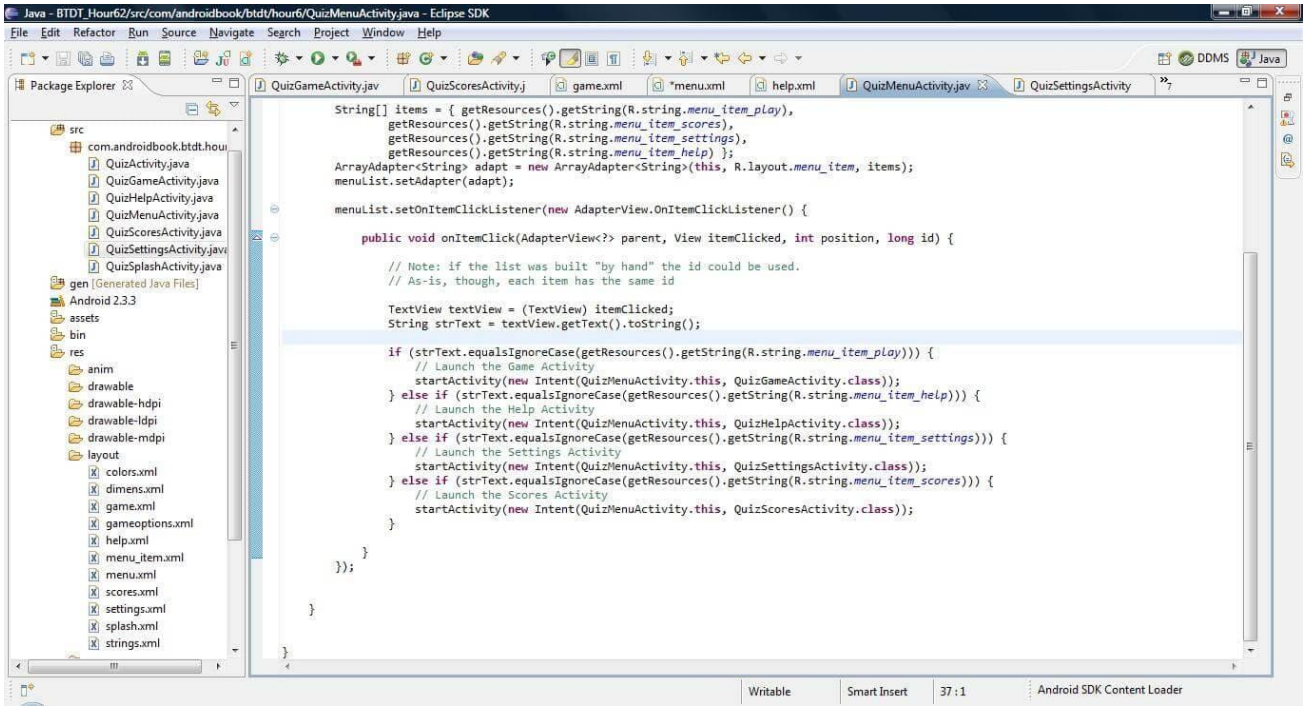


Рисунок 1.7 – Робота програми Eclipse [23]

Воно містить:

– засоби розробки(автодоповнення коду, перевірка помилок в реальному часі, рефакторинг і т.д.);

– менеджер SDK для управління версіями SDK;

– емулятор для розробки і тестування мобільних додатків без залучення реальних пристроїв і т.д.

Переваги даного ПЗ:

– досить просте у використанні;

– повністю безкоштовне;

– чудово підходить для командних проектів;

– тонко налаштовується і розширюється додатковими функціями.

Недоліки:

– спеціально створена для програмування Java, хоча й має плагіни для інших мов, але значно програє конкурентам у цій сфері;

– використовує дуже багато системних ресурсів та пам’яті;

– дуже повільно запускається;

– займає багато місця [24].

NetBeans з'явилась як студентський проект в Празі в 1996 році. У 1997 році IDE стала комерційним продуктом, а в 1999 році її викупила компанія Sun Microsystems (батьки Java) і вже на наступний рік представила реліз з відкритим вихідним кодом. (рисунок 1.8)

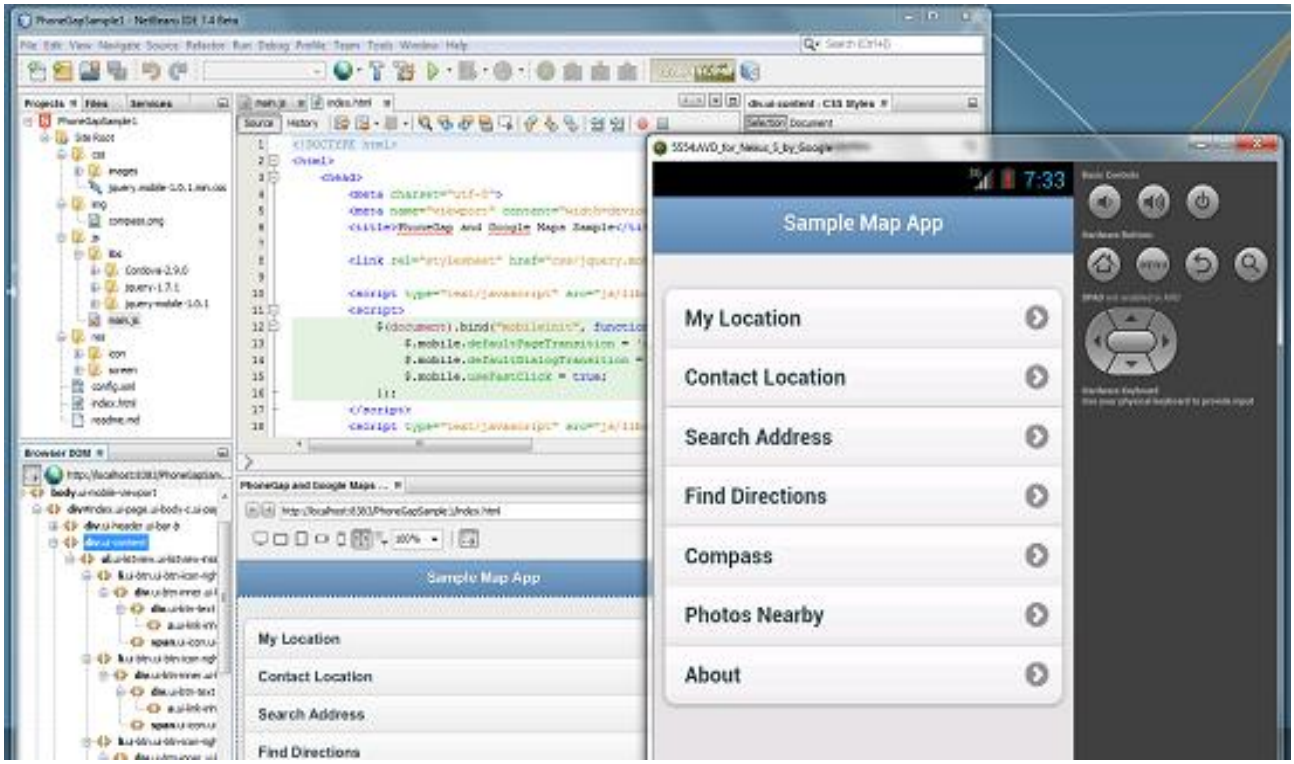


Рисунок 1.8 – Робота програми NetBeans [25]

Переваги:

– Редактор NetBeans виявляє помилки в той час, коли ви друкуєте, і допомагає вам за допомогою спливаючих підказок і «розумним» автодоповненням коду.

– У NetBeans є чудова вбудована підтримка Maven и Ant і Gradle.

– Гарна підтримка Java 8.

Недоліки:

– Вимоглива з боку системи, важка.

– Уступає конкурентам у кількості можливо вбудованих плагінів.

– Невідомий вектор розвитку якості ПЗ в майбутньому [26].

Таким чином, для розробки автоматизованого органайзера роботи викладачів навчальних курсів було обрано програмне середовище AndroidStudio, так як воно володіє рядом переваг, такі як: широкий функціонал, пакет інструментів для можливості тестування на пристроях найостаннішої версії, емулятор, тощо. Також серед конкурентів можна виділити, що є загальні недоліки для всіх ПС (такі як: вимогливість до апаратних засобів), але порівняно серед інших, своїх недоліків AndroidStudio майже не має.

1.4 Постановка задачі та вимоги до розробки системи

Метою кваліфікаційної роботи бакалавра є розробка автоматизованого електронного органайзера роботи викладачів навчальних курсів на платформі Android, що виконує наступні функції:

– планування проведення занять викладачів із навчальними групами;

– реєстрація присутності учнів на заняттях;

– робота з даними студентів, викладачів, аудиторій та навчальних курсів;

– долучення студентів у групи.

Розроблюваний автоматизований електронний органайзер роботи викладачів навчальних курсів має дозволяти співставляти групам учнів базові навчальні аудиторії та цільові курси.

Розділ 2

Проєктування інформаційної системи

2.1 Функціональна структура та бізнес-процеси системи

Згідно із поставленим завданням, необхідно виділити наступні бізнес-процеси, які потрібно реалізувати:

– вивід та введення основної та додаткової інформації в базу даних;

– побудова плану та перегляд, чи відповідає йому реальний стан речей;

– долучення студентів в групи та викладачів, що проводять заняття в них;

– відмічання присутності студентів на заняттях.

Бізнес-процес «Планування ведення занять викладачів з навчальними групами» передбачає побудову запланованого розкладу занять відповідно до атрибутів дати початку та кількості, що може допомагати формувати дні та час відповідно до якого можуть бути зайняті викладачі, аудиторії.

Користувач за роллю «адміністратор» може створювати дані розклади, редагувати їх та вилучати, а також перевіряти їх вірність та відсутність збоїв.

Також при проведенні занять є можливість заміни вільними аудиторіями, днями та часу або викладачами при реальних ситуаціях незбігів відповідно до поставлених планів.

Бізнес-процес «Реєстрація учнів на заняттях» дає можливість контролювати відвідуваність та успішність певного студента, вказувати причину відсутності, якщо вона є, уточняти, що клієнт був не все заняття, наприклад: поважна причина, пропустив по хворобі, був першу або другу половину.

Бізнес-процес «Робота з основними та додатковими даними» в основному лежить на адміністраторі, так як він має найбільшу роль в заданій ІС і під собою він містить складну систему різних значень.

Вищесказаний процес дає можливість створювати навчальну групу, студентів, викладачів, аудиторії, дисципліни або курси, редагувати їх або видаляти, пов’язувати з іншими даними в існуючій системи.

При створенні або редагуванні існуючих даних, вказаний тип введення додатково проходить валідацію для уникнення введення неправильної інформації.

Також потрібна функція перегляду усіх існуючих записів у комфортному форматі.

Бізнес процес «Навчання студента в групі» є основним у вже наявній системі, так як по предметній області в наявному підприємстві дохід отримується за рахунок того, що клієнтові надаються консультативно-інформативні послуги.

Так як розібрано вище, у користувача «адміністратор» є доступ до формування груп або включення в систему учнів, їх можна між собою зв’язувати одне до іншого, так як ці атрибути відповідають колективному навчанню дисципліні відповідно до поставленого плану. Тобто наявні два значення є взаємопов’язані і розгляд одного неможливо без існування іншого в заданій системі.

При розробці електронного автоматизованого органайзера викладача навчальних курсів були винесені такі основні структури інформаційних потоків, кожній з яких потрібно забезпечити взаємодію одну з одною.

– робота з інтерфейсом мобільного додатку;
– робота з БД;
– робота з серверною частиною.

Інтерфейс відповідає за зовнішній вигляд додатку. На перший погляд, функціональності він в собі не несе, але дизайн ПП зазвичай є «видимою областю» для користувача. Тобто якщо створення запитів до серверної частини або звернення до БД для нього є тою операцією, виконання якої він не бачить, то інтерфейс є те, що пов'язує користувача з тим сегментом операцій, які відбуваються на фоні. Інтерфейс представляє собою функціонал розміщення всіх елементів в логічних і видимих місцях. Додання до них текстового пояснення стосовно їхнього функціоналу (рисунок 2.1).

Рисунок 2.1 – Діаграма дій роботи інтерфейсу

Серверна частина дає можливість користувачеві мати доступ до віддаленого серверу та постійно мати взаємозв'язок з БД для обіну інформацій. Автоматизації підлягають наступні функції (рисунок 2.2):

- Обробка інформації відповідно по днях тижня, числу, місяцю.
- Виведення інформації стосовно розкладу.
- Побудова планової роботи на певний майбутній період часу.
- Звіти про проведену роботу, заняття.

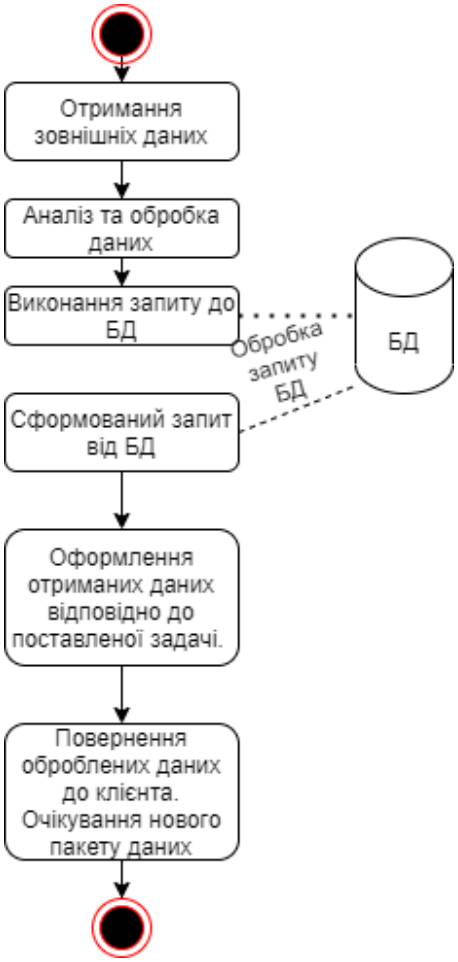


Рисунок 2.2 – Діаграма дій роботи серверної частини

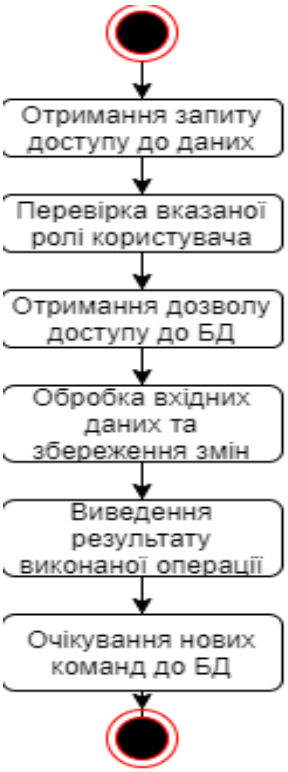


Рисунок 2.3 – Діаграма дій роботи з БД

Робота з БД дає можливість користувачу працювати з систематизованою інформацією, а тобто: створювати, редагувати, видаляти та зчитувати дані стосовно розкладу, занять, студентів, викладачів. (рисунок 2.3.) Автоматизуються функції створення, редагування та видалення даних.

2.2 Інформаційна структура системи

Для забезпечення збереження інформації та даних було розроблено БД автоматизованого електронного органайзера роботи викладачів навчальних курсів, даталогічна схема якої зображення на рисунку 2.4.

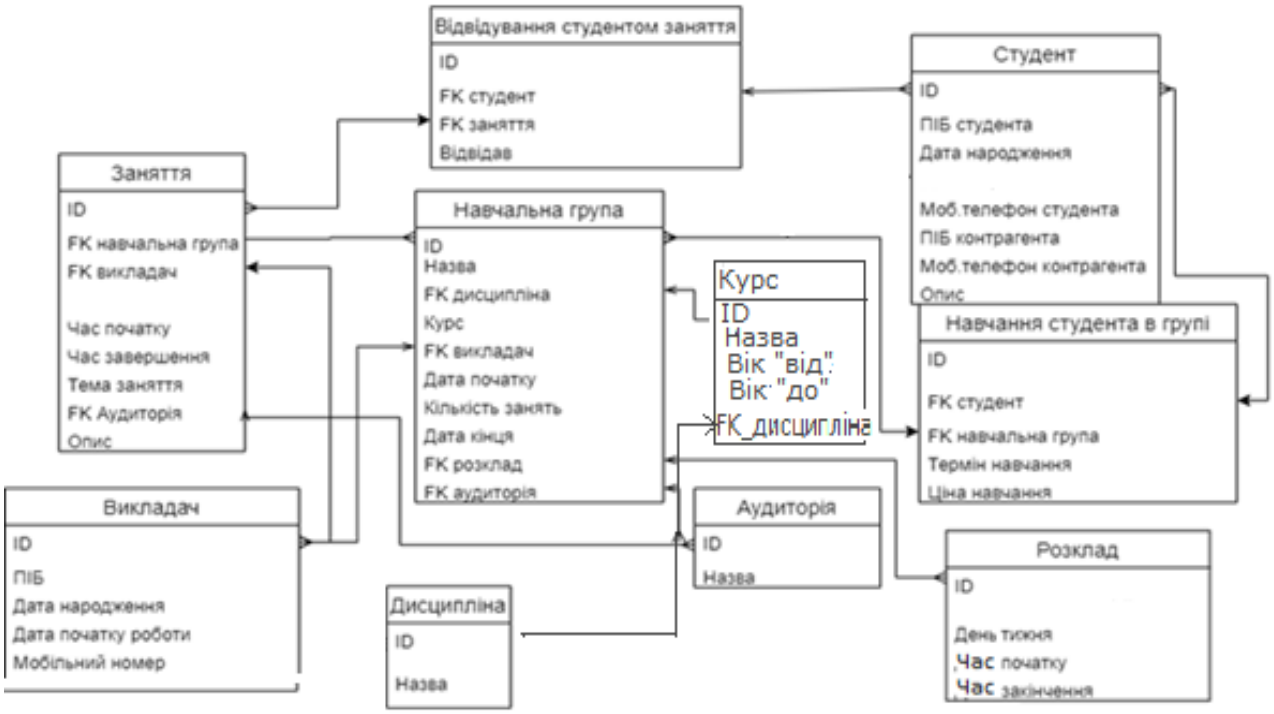


Рисунок 2.4 – Схема бази даних автоматизованого електронного органайзера роботи викладачів навчальних курсів

Головною таблицею в даній БД є «Відвідування студентом заняття», яка включає в себе не так багато інформації, але від цього пункту є досить багато гілок розвитку подій, відповідно до поставлених задач. У цій таблиці можна

вносить дані у відповідні поля: «Студент», «Заняття» та інформація про те, чи відвідав він його чи ні (таблиця 2.1).

Таблиця 2.1 – Атрибути таблиці «Відвідання студентом заняття»

№	Назва атрибуту	**Тип даних**	**Опис**
1	ID	int	Лічильник, унікальний
2	FK_Студент	int	Зовнішній ключ до таблиці Студент
3	FK_Заняття	int	Зовнішній ключ до таблиці Заняття
4	Відвідав	boolean	Відмітка про те, що студент відвідав заняття

Взаємозв’язок між вищесказаними сутностями існує і в таблиці «Навчальна група», так як група закріплена за сутністю «студента» і одночасно з тим, сутність «Заняття» також проводиться в даній групі. Єдине що для забезпечення атомарності значень, т.я. студент може навчатись в різних групах, автоматично значення в даному полі в таблиці «Студент» буде встановлюватись відповідно до того, до якої групи студент був занесений раніше. Але це значення можна буде редагувати.

Так як сценаріїв подій і потрібних відповідних результатів є доволі багато, розберемо кожну таблицю окремо.

Таблиця «Студент» представляє собою сутність, про яку відомо: прізвище, ім’я, по-батькові, дату народження, мобільний номер телефону (дане поле може бути порожнім), група, за якою він числиться як студент (або числився). Окрім того, існує таке поняття як «Контрагент», тобто людина, що оплачує навчання. Контрагентом може виступати і сама контактна особа, тоді дані потрібно буде просто продублювати: ПІБ та мобільний телефон (але це поле вже не може бути порожнім). Також є таке поле як «Опис», куди може бути занесена додаткова інформація стосовно студента (таблиця 2.2).

Таблиця 2.2 – Атрибути таблиці «Студент»

№	Назва атрибуту	Тип даних	Опис
1	ID	int	Лічильник, унікальний
2	ПІБ	varchar	Прізвище, ім’я, по-батькові студента
3	Дата народження	date	Дата народження студента
4	Мобільний телефон студента	int	Номер, може бути порожнім
5	ПІБ контрагента	varchar	Прізвище, ім’я, по-батькові контрагента
6	Мобільний телефон контрагента	int	Номер, обов’язкове для заповнення
7	Опис	varchar	Додаткова інформація, може бути порожнім

Між головною таблицею існує взаємозв’язок з таблицею «Заняття», відповідно до якої можна побудувати план на подальшу роботу та розклад зайнятості викладачів та аудиторій. Заняття зазвичай проводиться відповідно розкладу в навчальній групі, але відповідно до обставин, певні значення можуть не співпадати. Поля таблиці «Заняття»: «Навчальна група», «Викладач» (зазвичай той, який закріплений за групою), «Час початку», «Час завершення» (зазвичай термін проведення – чітко фіксований час), «Тема заняття», «FK аудиторія» (зазвичай – та, яка закріплена за групою), «Опис» (таблиця 2.3)

Розвідна таблиця «Навчальна група» в багато-чому є схожою на таблицю «Заняття», так як деякі поля співпадають, але сформована група лише описує теоретичний план занять, в той час, як таблиця «Заняття» пояснює реальний хід подій (таблиця 2.4).

Таблиця 2.3 – Атрибути таблиці «Заняття»

№	Назва атрибуту	Тип даних	Опис
1	ID	int	Лічильник, унікальний
2	FK_Навчальна група	int	Зовнішній ключ до таблиці Навчальна група
3	FK_Викладач	int	Зовнішній ключ до таблиці Викладач
4	Час початку	datetime	Дата та час початку заняття, округляється з кроком в «15 хвилин»
5	Час завершення	datetime	Дата та час завершення заняття, округляється з кроком в «15 хвилин»
6	Тема заняття	varchar	Інформація з порядковим номером, типом заняття (лекція, самостійна, практичне) та сама назва теми
7	FK_Аудиторія	int	Зовнішній ключ до таблиці Аудиторія
8	Опис	varchar	Додаткова інформація, може бути порожнім

Таблиця 2.4 – Атрибути таблиці «Навчальна група»

№	Назва атрибуту	Тип даних	Опис
1	ID	int	Лічильник, унікальний
2	Назва	varchar	Назва групи, яка може містити деякі значення інших полів
3	FK_дисципліна	int	Зовнішній ключ до таблиці Дисципліна
4	Курс	int	Номер курсу, також може бути порожнє значення
3	FK_Викладач	int	Зовнішній ключ до таблиці Викладач
4	Дата початку	date	Дата старту групи
5	Кількість занять	int	Номер
6	Дата початку	date	Дата випуску групи (відповідно до поставленого плану)
7	FK_Аудиторія	int	Зовнішній ключ до таблиці Аудиторія
8	FK_Розклад	int	Зовнішній ключ до таблиці Розклад

Таблиця «Навчання студента в групі» відповідає тому, що 1 студент може навчатись в декількох групах. Тому виділяється окрема сутність, яка відповідає навчання одного студента в одній навчальній групі (таблиця 2.5).

Таблиця 2.5 – Атрибути таблиці «Навчання студента в групі»

№	Назва атрибуту	Тип даних	Опис
1	ID	int	Лічильник, унікальний
2	FK_Студент	int	Зовнішній ключ до таблиці Студент
3	FK_Навчальна група	int	Зовнішній ключ до таблиці Навчальна група
4	Термін навчання	int	Число, відповідає кількості тижнів навчання (навіть якщо 1 день – все округлюється в більшу сторону)
5	Ціна навчання	int	Вартість навчання в гривнях

Таблиця «Дисципліна» представляє собою вид спеціальності, що вивчається в даній навчальній групі, «Дисципліна» ділиться по курсам відповідно рівню здобутих знань, але дана інформація співвідноситься лише до таблиці «Навчальна група» (таблиця 2.6).

Таблиця 2.6 – Атрибути таблиці «Дисципліна»

№	Назва атрибуту	Тип даних	Опис
1	ID	int	Лічильник, унікальний
2	Назва	varchar	Текстовий тип
3	Нижня межа віку	int	Число
4	Верхня межа віку	int	Число
5	Опис	varchar	Текстовий тип, може бути порожнім

Таблиця «Аудиторія» включає в себе лише назву аудиторії. Нас цікавить лише той час, в який та чи інакша аудиторія буде зайнята (за розкладом або назаплановано), тому ця таблиця взаємопов’язана з двома: «Навчальна група» і «Заняття» (таблиця 2.7).

Таблиця 2.7 – Атрибути таблиці «Аудиторія»

№	Назва атрибуту	Тип даних	Опис
1	ID	int	Лічильник, унікальний
2	Назва	varchar	Текстовий тип

Таблиця «Розклад» уявляє собою відповідність дня тижня відповідному часу, що взаємопов’язана з таблицею «Навчальна група» (таблиця 2.8).

Таблиця 2.8 – Атрибути таблиці «Розклад»

№	Назва атрибуту	Тип даних	Опис
1	ID	int	Лічильник, унікальний
3	День тижня	varchar	Текст
4	Час початку	time	Формат HH.MM.SS, відповідає часу початку заняття з кроком в 15 хвилин
5	Час початку	time	Формат HH.MM.SS, відповідає часу завершення заняття з кроком в 15 хвилин

Ще одна ключова важлива таблиця «Викладач», яка представляє сутність для 2 ситуацій: Людина, яка проводить заняття (облік цих занять) та працівника, що отримує за це кошти. Перший випадок прослідковується зв’язок з таблицями «Навчальна група» та «Заняття», т.я. вони включають поля, суттєву роль в яких приймає особа у вигляді працівника (за групою та за заняттями закріплюється викладач) (таблиця 2.9).

Таблиця 2.9 – Атрибути таблиці «Викладач»

№	Назва атрибуту	Тип даних	Опис
1	ID	int	Лічильник, унікальний
2	ПІБ	varchar	Прізвище, ім’я, по-батькові
3	Дата народження	date	Дата народження викладача
4	Дата прийому на роботу	date	Дата прийому на роботу
5	Мобільний номер	int	Номер, обов’язкове поле

Отже, було спроектовано БД автоматизованого електронного органайзера роботи викладачів навчальних курсів.

2.3 Вибір засобів розробки інформаційної системи

2.3.1 Вибір мови програмування

Говорячи про мобільне програмування, а саме про розробку під Android, завжди виділяють такі мови програмування: Java, Kotlin.

Java – мова, розроблена Джеймсом Гослінгом в 1995 році, що з початку зайняла провідні місця серед інших мов розробки та до сих пір знаходиться в Топ3.

Згідно з індексом Tiobe (рисунок 2.5) [27] , лише дві мови були на перших місцях за 20 років: Java та C. І як видно по рисунку, серед вище зазначених мов програмування, тільки Java і представлена на рейтингу місць. Що означає, що інші мови або ще не набули популярності своїх позицій, або дійсно не так багато де застосовуються.

Досліджуючи тенденцію розвитку мови Java (рисунок 2.6), помітно стрімкий зріст з початком 21 століття (тобто через 5 років, як вона була розроблена Джеймсом Гослінгом), і з 2006 року найчастіше опинялась на 1 місці.

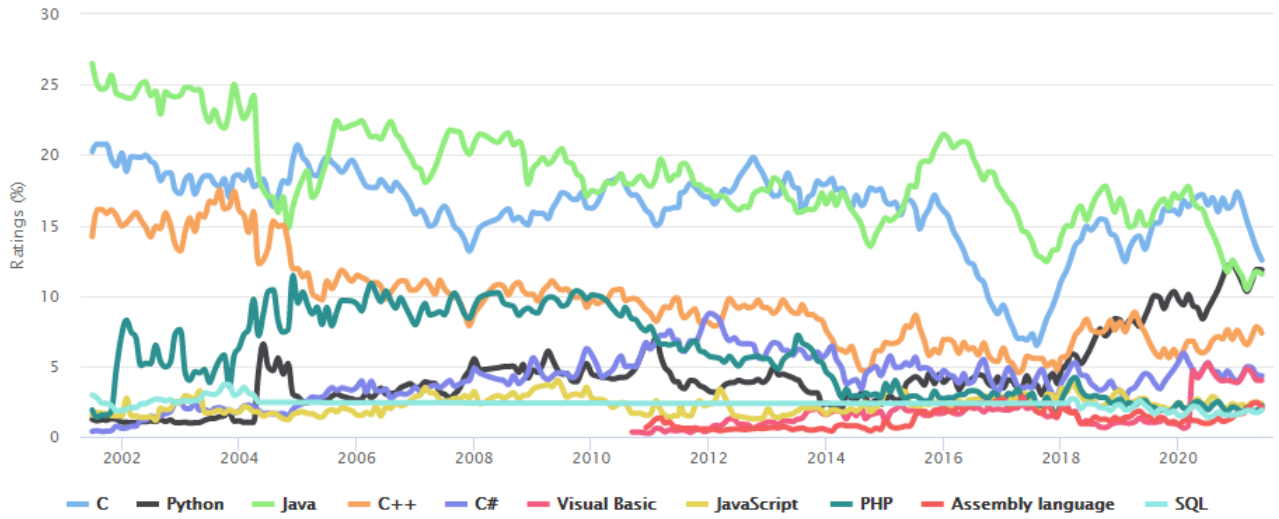


Рисунок 2.5 – Рейтинг мов за індексом Tiobe [27]

Programming Language	2021	2016	2011	2006	2001	1996	1991	1986
C	1	2	2	2	1	1	1	1
Java	2	1	1	1	3	25	-	-
Python	3	5	6	8	26	20	-	-
C++	4	3	3	3	2	2	2	8
C#	5	4	5	7	13	-	-	-
Visual Basic	6	13	-	-	-	-	-	-
JavaScript	7	7	10	9	10	28	-	-
PHP	8	6	4	4	11	-	-	-
SQL	9	-	-	-	-	-	-	-
R	10	17	29	-	-	-	-	-
Ada	35	28	18	16	21	8	3	2
Lisp	36	27	13	14	17	7	6	3
(Visual) Basic	-	-	7	5	4	3	4	5

Рисунок 2.6 – Мови по місцях за рейтингом Tiobe [27]

З 2009 року мова перейшла під керівництво компанії «Oracle». Java-програми компілюються у байт-код, який при виконанні інтерпретується віртуальною машиною для конкретної платформи.

Java є об’єктно-орієнтованою мовою програмування, що говорить про спосіб написання структурного коду Java, а саме: поділ коду на так звані

«класи», які запускаються разом, щоб забезпечити узгоджене створення об'єктів. Це призводить до універсального і організованого коду, який просто редагувати. До ще однієї переваги можна віднести кросплатформеність, що дає змогу використовувати програмний код на будь-якій системі, де наявна віртуальна машина Джава. Простіше кажучи: один раз код написали і використовуємо ПЗ де-завгодно. Мова містить можливість вбудованості, бути багатопотоковою, із динамічним зв'язуванням модулів». Java використовує автоматичний збирач сміття (GC - Garbage Collector) для керування пам'яттю під час життєвого циклу об'єкта [28].

Переваги мови Java:

– об'єктно-орієнтована мова програмування;

– незалежна від платформи, для програмування на ній та для запуску достатньо встановлення JVM;

– порівняно неважкий синтаксис;

– широкий ряд застосування;

– мова багатопотока, динамічна;

– є система виняткових випадків, що дозволяє уникати помилковість виконання та програмування, або швидко знаходити баги та своєчасно їх знищувати;

– автоматичне керування пам’яттю.

Недоліки:

– так як мова розроблялась як платформово-незалежна, вона має меншу низькорівневу структуру, що, у свою чергу, призводить до меншої швидкодії виконання програм;

– через причину вище, Java має менше можливостей для роботи з апаратним ПЗ [29].

Kotlin є новою мовою програмування, так як перший раз реліз відбувся в 2016 році. Можливо, через це, вона ще не зазнала такої популярності, як Java, хоча судячи з того, що вона входить в трійку кращих мов під програмування на

мобільних системах, вона є достойним конкурентом лідеру програмування для телефонних пристроїв.

Мова Kotlin є розробкою компанії JetBrains і поєднує в собі багато принципів з інших мов програмування, можна назвати її «гібридом» з гігантів IT-ринку (рисунок 2.7).

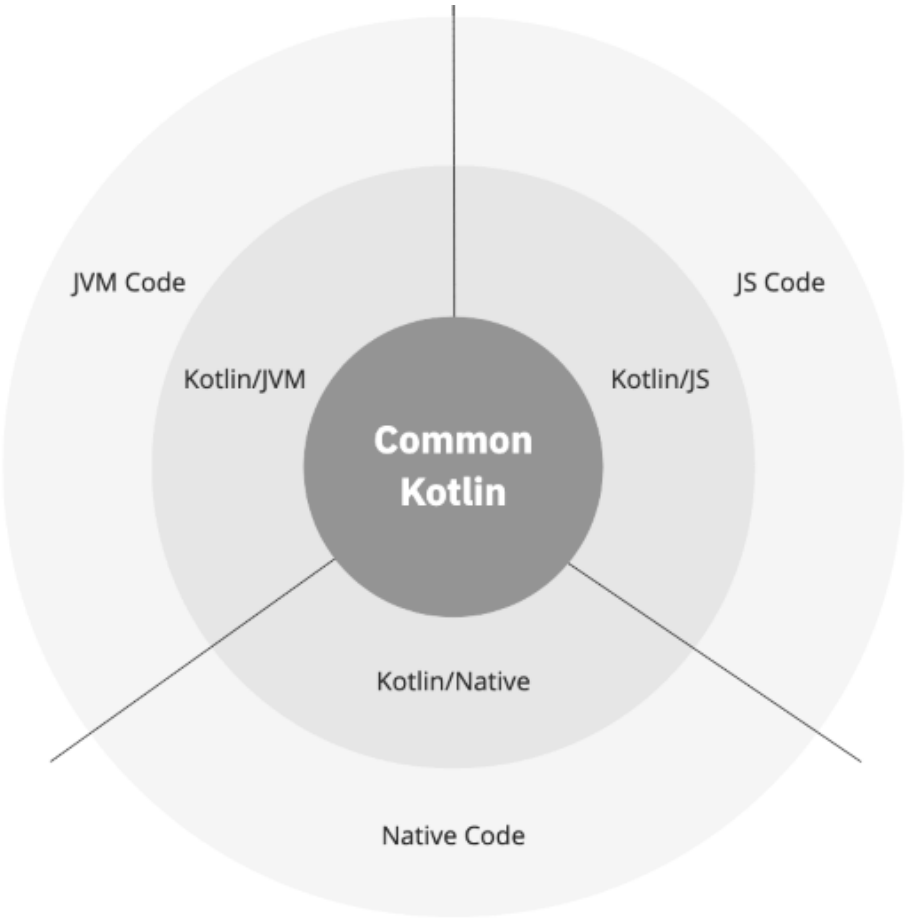


Рисунок 2.7 – Структура мови Kotlin, поєднання різних систем [30]

Спочатку розробники мови Kotlin займались лише розробкою якісних ICP (IDE) і працювали переважно як-раз таки на вже згаданій мові Java, але з плином часу, вони вирішили змінити свій профіль, тому що хотіли перестати на ній працювати, але у свої принципи Kotlin багато перейняв від Java, а також від інших мов програмування, ввібравши позитивні якості. До речі, стосовно взаємозв’язків Java і Kotlin, їх можна помітити в назві, яка йде як дані мові Java, так як Котлін – це острів в Фінському заливі, а одна з версій, чому Джава так називається це – в честь острова Ява, що розміщений в Індонезії [31].

Мова Kotlin побудована на принципах легкості читання і розуміння коду, динамічності (можливості вирішення однієї задачі багатьма способами), велику кількість бібліотек для реалізації різних функцій, кросплатформеність, зрозуміла документація і націлена вона для отримання міцної платформи підтримки товариства юзерів [32].

Загальна популярність і відомість Kotlin прийшли тоді, коли компанія Google надала свою підтримку, зробивши її офіційною мовою для розробки додатків під Android. Це було зроблено, так як компанія давно спостерігала за розвитком мови Kotlin і з боку Google вона побачила плюси у вигляді лаконічності та зручності коду, а також сумісність з Java, який є передовим в області мобільної розробки і вже давно є офіційним в розробці під Android [33].

Переваги Kotlin:

– код є інтуїтивно зрозумілим, тому доволі легким у вивченні;

– висока сумісність з Java;

– відкритий вихідний код;

– швидка загрузка додатків, так як оновлення йде лише необхідного вмісту.

Недоліки:

– нешвидка компіляція програмних кодів;

– мова поки не має широкого загалу прихильників, що ускладнює пошук необхідного робочого складу для розробки ПП;

– хоча розробники докладають зусиль для розповсюдження мови, тим не менш, співтовариство ще не є широким, що ускладнює програмування, так як при пошуку інформації для вирішення завдань, її може не бути в мережі.

Підсумовуючи вищесказане, було обрано мову програмування Java, так як вона володіє рядом переваг, що відповідає поставленій задачі КРБ. При розробці на Java під Android використовуються не тільки Java-класи, що містять код, але також файли маніфесту на мові XML, що надають системі основну інформацію про програму, і системи автоматичного складання Gradle, Maven або Ant, команди в яких пишуться на мовах Groovy, POM і XML відповідно.

2.3.2 Вибір СКБД

Для розробки автоматизованого електронного органайзера роботи викладачів навчальних курсів потрібне відповідне програмне забезпечення для забезпечення зв'язку з БД, тому необхідно проаналізувати існуючі СКБД, порівняти їх функції і вибрати те, яке найкраще підходить для реалізації ІС.

Серед існуючих ПЗ виділені MS Access, MySQL, PostegreSQL, MS SQL Server, Oracle Database, так як вони є найбільш поширені на поточний час.

MS SQL Server – це багатофункціональне рішення для структуризації БД, формування запитів SQL, який підходить для бізнес-аналітики та зберігання даних. Microsoft SQL Server забезпечує інтегровані у БД функції для обробки інформації в ОП при навантаженнях системи, швидкі результати аналізу і систематизації з використанням аналітики, а також інструменти для обробки великої кількості даних. SQL Server також включає можливість налаштування БД до хмарних сервісів[34] (рисунок 2.8).

Переваги:

– швидка обробка запитів і аналіз даних;

– платформа для гібридного хмарного сховища;

– можливість резервного копіювання і зберігання в ситуації випадкового виходу з системи;

– підтримує кластеризацію БД;

– легко взаємодіє з іншими продуктами Windows, зокрема з платформою Framework .Net;

– є пошук по ключовим значенням, фільтри.

Недоліки:

– як продукт компанії Microsoft, при інсталяції, налаштовує програми, розбиваючись на підпрограми, надлишковість функцій (так як далеко не всі функції потрібно реалізовувати);

– доволі дорогий (має обмежений безкоштовний функціонал).

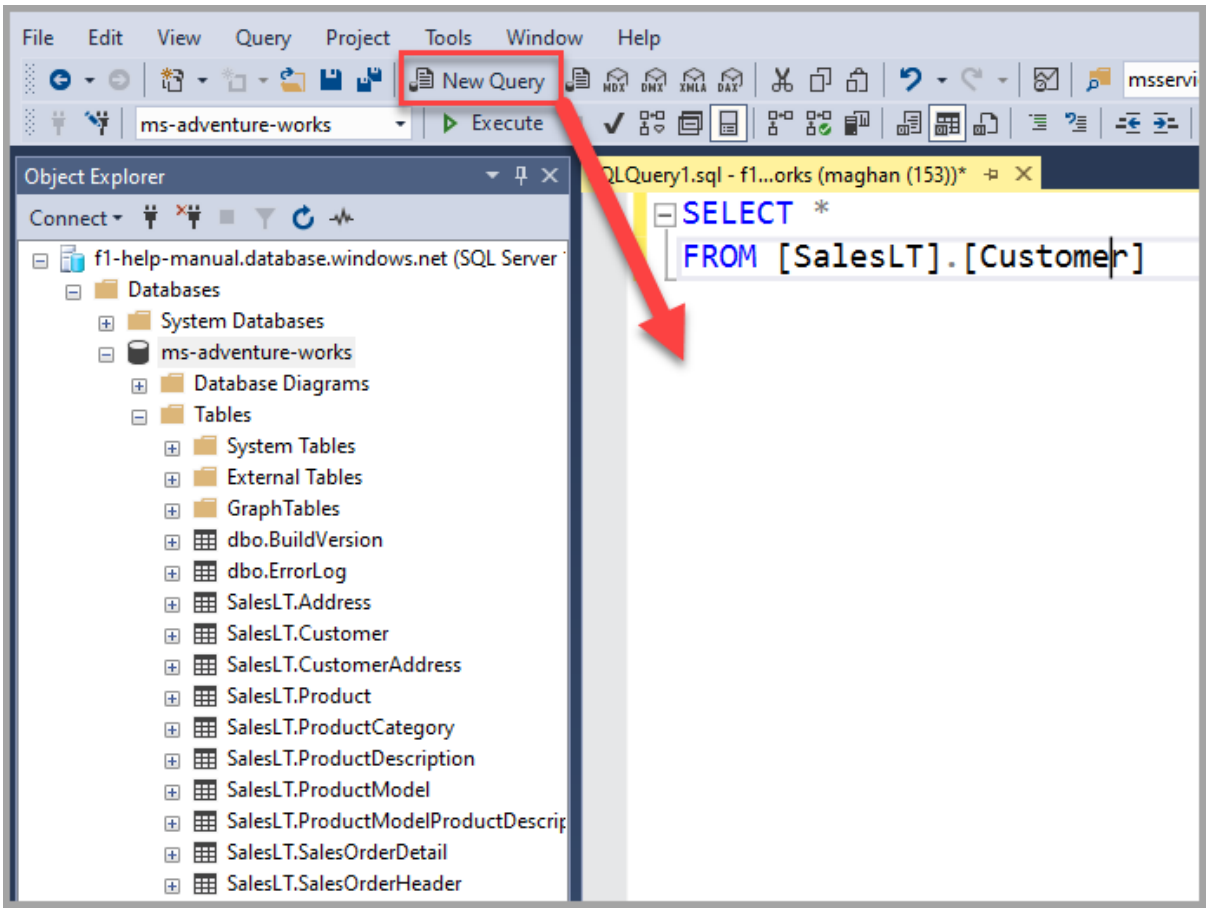


Рисунок 2.8 – Робота СКБД Microsoft SQL Server [35]

Microsoft Access – є офісним продуктом компанії Microsoft, підходить під реалізацію невеликих систем, часто використовується для навчальних цілей або для невеликих бізнесів. Для розробки ПЗ не зовсім підходить, так як функціонал дуже обмежений і спрощений. При цьому має свої позитивні сторони такі, як: велика кількість шаблонів та інструментів, легке перемикання між різними вкладками, можливість формування SQL-запитів, легкий дизайн, швидкість обробки запитів і можливість інтегрування з таблиць Excel [36].

Як було сказано вище, повністю весь фунціонал з роботою з БД Microsoft Access не може надати, але може працювати в парі з іншим продуктом компанії: MS SQL, беручи на себе роль одного з інструментів в роботі з нею, що й рекомендуються самою компанією (рисунок 2.9).

Переваги Microsoft Access:

– легкий в роботі, швидкий, інтуїтивно зрозумілий інтерфейс;

– підходить для людей різного кваліфікаційного рівня, так як простий у вивчення;

– володіє інструментами для легкого впровадження в інші ПП;

– широкий спектр візуальних інструментів для роботи [37].

Мінуси:

– рідко в користуванні працює як самостійний продукт, швидше як движок для роботи інших ПП;

– невеликий функціонал, не підходить для всіх типів задач;

– не є веб-сервісом;

– не є багатопотоковим;

– є платним, безкоштовний він тільки для власників платної версії Windows.

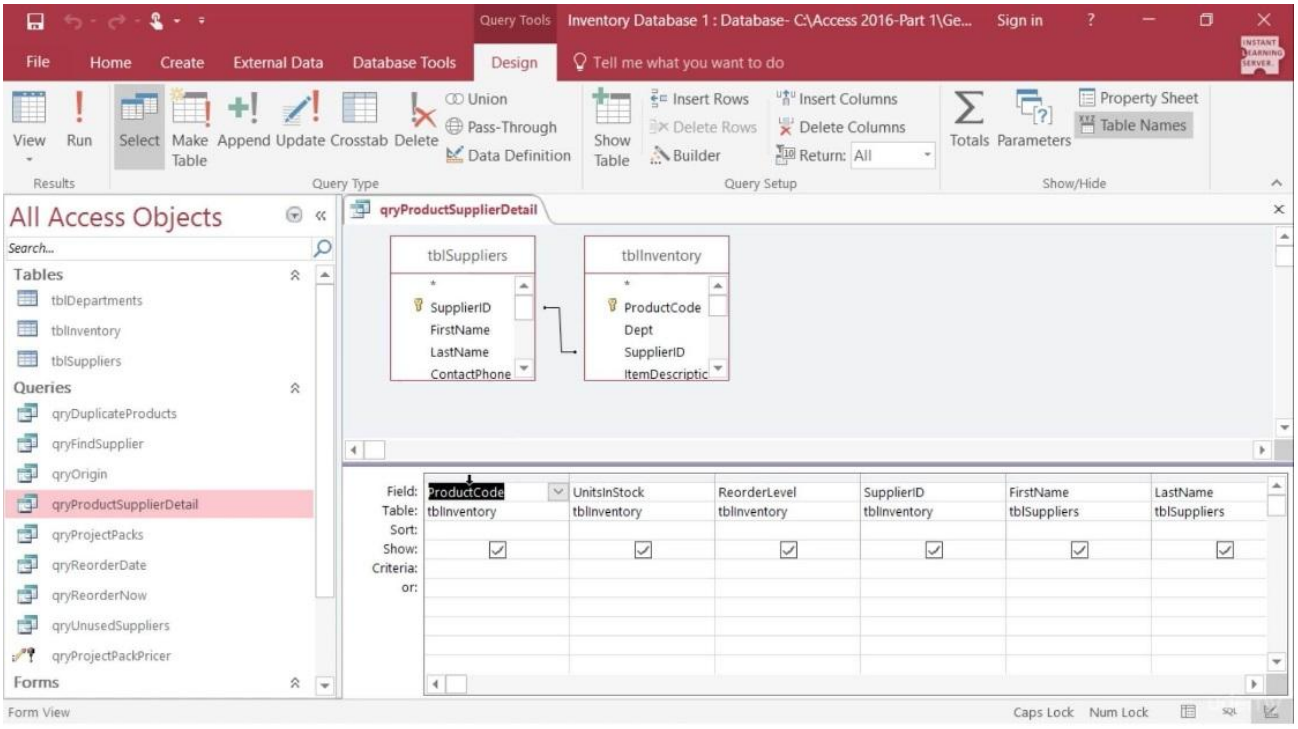


Рисунок 2.9 – Робота СКБД MS Access [37]

Для побудови таблиць і взаємозв’язків між ними необхідна вільна система для управління реляційними БД. Розробку і підтримку MySQL здійснює компанія Oracle. З найперших версій виник механізм реплікації.

MySQL виникла як продукт компанії для того щоб застосувати mSQL для власної інтеграції таблиць, БД для яких використовувались програми низького рівня [38]. У фіналі був створений інтерфейс SQL, який легко взаємодіяв і реалізовував принципи mSQL.

Входить у склад серверів WAMP, AppServ, LAMP і в портативні збірки серверів Денвер, XAMPP, VertrigoServ. Зазвичай MySQL використовується у якості сервера, до якого звертаються локальні або віддалені клієнти, хоча в дистрибутив входить бібліотека внутрішнього серверу, що дозволяє включати MySQL в автономні програми.

Гнучкість СКБД MySQL забезпечується підтримкою великої кількості типів таблиць: користувачі можуть вибрати як таблиці типу MyISAM, що підтримують повнотекстовий пошук, так і таблиці InnoDB, що підтримують транзакції на рівні окремих записів. Більш того, сервіси СКБД MySQL включають спеціальний тип таблиць EXAMPLE, що виконують демонстративні цілі. Завдяки відкритій архітектурі і GPL-ліцензуванню, в СКБД MySQL постійно з'являються нові типи таблиць [38].

MySQL використовується у великій кількості CMS і на даний момент більшість вебсайтів використовують MySQL як мінімум провідну, а деякі підприємства – у своїй роботи застосовують тільки її без інакших аналогів. Це досягнуто за рахунок швидкості роботи, а також підтримкою великої кількості видів таблиць. Окрім того MySQL проста в користування, легко взаємодії і з БД, що робить її провідною на ринку IT

MySQL може містити до 50 мільйонів рядків коду, до 4 Гб ємності файлу, підтримує секціонування, Xpath і збережені процедури, тригери та візуалізацію.

Переваги MySQL:

– масштабованість, багато де застосовується;

– підтримка кластеризації, секціонування, реплікації;

– швидкість обробки запитів;

– має найзручніший функціонал для роботи вебсервісів;

– підтримку на багатьох платформах, універсальність [39].

Недоліки:

– невисока ефективність процесів розробки;

– лише частково відповідає стандартам SQL;

– можуть виникати проблеми з багатопотоковістю [40];.

Для спрощення роботи з сервером MySQL в базовий комплект установки входить такий інструмент як MySQL Workbench. Це графічний клієнт для роботи з сервером, через який можна створювати БД та керувати ними [41] (рисунок 2.10).

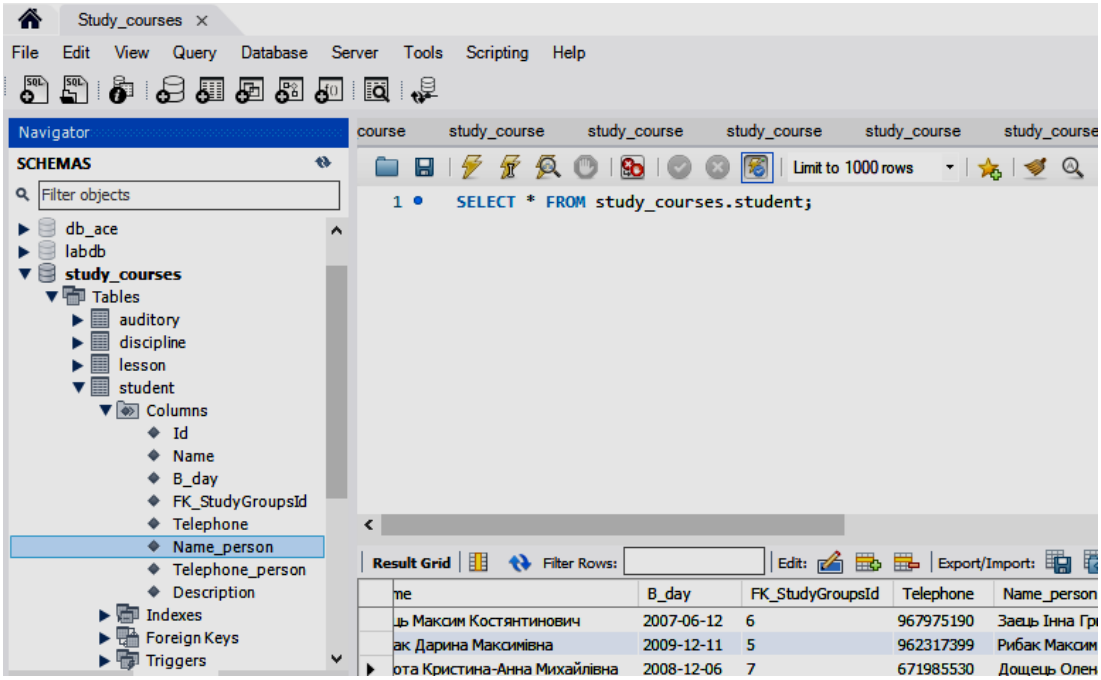


Рисунок 2.10 – Робота MySQL Workbench [42]

Функції, що надає MySQL Workbench:

– візуальне подання таблиць;

– функціональний і наочний механізм створення взаємозв’язків між таблицями;

– відновлення структури таблиць із вже існуючої на сервері БД;

– зручний відкритий редактор коду, що надає можливість оновлення БД за допомогою запитів в поточний момент часу написання коду;

– можливість редагування таблиць в режимі їх візуального відображення.

Отже, для розробки БД автоматизованого електронного органайзера роботи викладачів навчальних курсів в подальшому використовуються: СКБД MySQL і графічний клієнт для роботи з сервером MySQL Workbench через популярність і ряд переваг, що дають можливість створювати і керувати БД.

Розділ 3

Програмна реалізація інформаційної системи

3.1 Структура та функціональне призначення модулів системи

Сервер програмного продукту реалізовується в програмному середовищі Eclipse за допомогою бібліотеки HttpServer, що дозволяє підключитись до бази даних і в асинхронних фонових потоках задавати запити до неї, отримуючи при цьому зворотну відповідь. Також в Eclipse є моделі даних, які представлені у вигляді простих класів з переліком змінних, що уявляють собою метадані в БД відповідно до таблиць. Архітектуру інформаційної системи зображено на рисунку 3.1.

Рисунок 3.1 – Архітектура інформаційної системи

Подібна система реалізована і в Android-Studio – є користувацький інтерфейс, кнопки, надписи, поля для вводу і користувач може переходити по формам відповідно при введенні певних даних, тобто. Зі спільного з сервером:

– наявність окремих класів, що включають в себе подання даних;

– наявність статичних класів, що працюють в фоновому режимі;

– методи, створені на базі (у випадку Eclipse) цих класів або для кращого відображення виведеної інформації.

З відмінного:

– AndroidStudio включає в себе файли для відображення інформації, тобто саму клієнтську частину;

– сервер напряму робить запити до БД і в основному методі класу має перелік виконуваних запитів, а в AndroidStudio відбуваються Http-запити у вигляді посилань до серверу.

3.2 Розробка програмних модулів

Розробка програмних модулів починалась зі створення бази даних, а тобто – відповідних таблиць та взаємозв'язків між ними. Скрипт, з якого було створено базу даних продемонстровано на рисунку 3.2. А на рисунку 3.3 можна побачити заповнення даної БД.

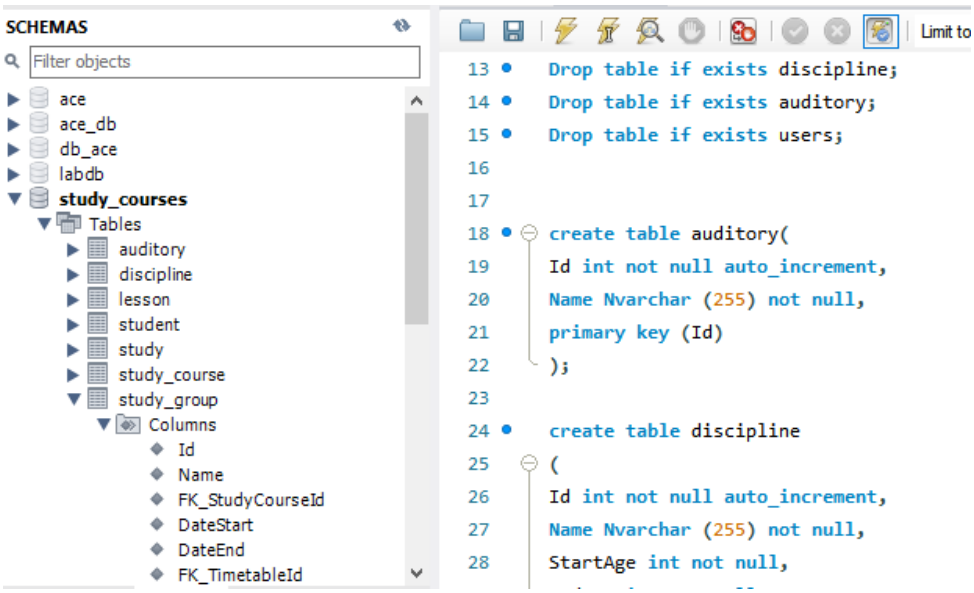


Рисунок 3.2 – Скрипт СКБД по створенню бази даних

Далі задачею виступало створення серверної частини, що зможе забезпечувати запити до БД і зчитування звідти інформації. В попередньому розділі пояснювалось принцип їх роботи, але ще вищесказану методику можна побачити по лістингам кодів. Продовжимо з того місця, де зупинились: в

серверній частині функції забезпечувались створенням інтерфейсу, який необхідний для перевірки правильності Http-запиту:

```
static class DisciplineHandler implements HttpHandler
{
	@Override
	public void handle(HttpExchange arg0) throws IOException {
		Discipline[] disciplines;
		try {
			disciplines=getDisciplines();
			if(disciplines==null)
			{
				String response = "Disciplines not found";
				arg0.sendResponseHeaders(401, response.length());
				OutputStream os = arg0.getResponseBody();
				os.write(response.getBytes());
				os.close();
			}
			else
			{
				Gson gson = new Gson();
				String response = gson.toJson(disciplines);
				Headers responseHeaders = arg0.getResponseHeaders();
				responseHeaders.set("Content-Type", "application/json;
charset=UTF-8");
arg0.sendResponseHeaders(200, 0);
				Writer os = new OutputStreamWriter(arg0.getResponseBody(),
"UTF-8");
				os.write(response);
				os.close();
			} }
		catch (ClassNotFoundException | SQLException e) {
			String response = e.toString();
			arg0.sendResponseHeaders(500, response.length());
			OutputStream os = arg0.getResponseBody();
			os.write(response.getBytes());
			os.close();

		} }
```

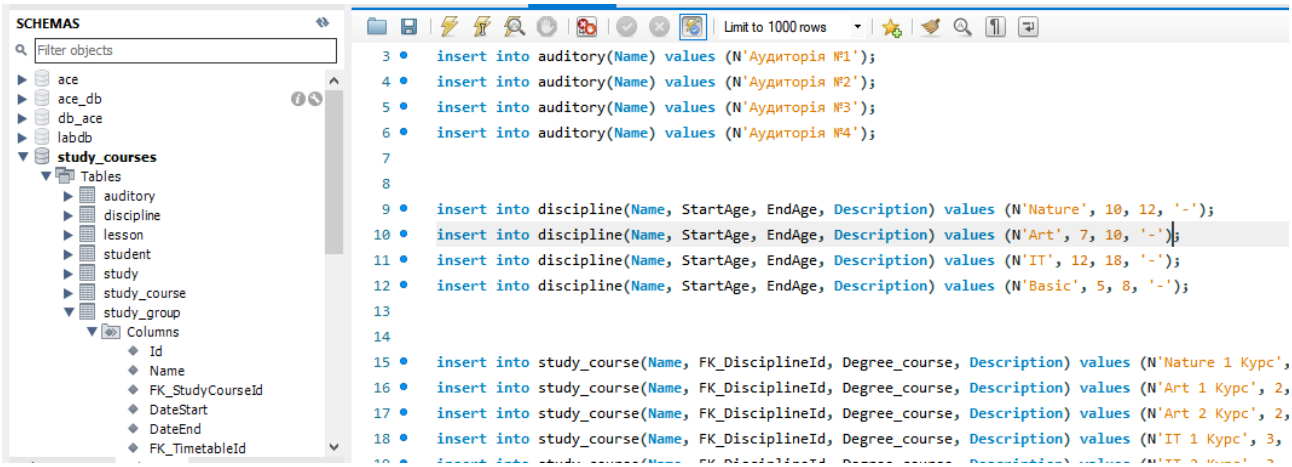


Рисунок 3.3 – Скрипт по заповненню бази даних

Вище можна побачити, що інтерфейс викликає метод, який представлений нижче:

```
private Discipline[] getDisciplines() throws ClassNotFoundException, SQLException
	{
		ArrayList<Discipline>disciplines=new ArrayList<Discipline>();
		Connection connection = createConnection();
		String query;
		query= "SELECT * FROM discipline;";
		Statement stmt = connection.createStatement();
		ResultSet rs = stmt.executeQuery(query);

		while (rs.next())
		{
			Discipline dItem = new Discipline();
			dItem.Id = rs.getInt("Id");
			dItem.Name = rs.getString("Name");
			dItem.StartAge = rs.getInt("StartAge");
			dItem.EndAge = rs.getInt("EndAge");
			dItem.Desctiption= rs.getString("Description");
			disciplines.add(dItem);
		}
		connection.close();
		return disciplines.toArray(new Discipline[disciplines.size()]);
	}
}
```

Даний фрагмент коду формує запит до БД і повертає дані, які, при правильному прописаному маршруту, будуть оформлені вже у Gson-формат. Для того, щоб запит працював, потрібно вказати URN і назву самого Http-класу в основному методі класу.

Тепер стосовно клієнтської частини: в нас наявний простий функціонал, який відображає дані, надає можливість їх редагувати та видаляти. (рисунок 3.4 і 3.5).

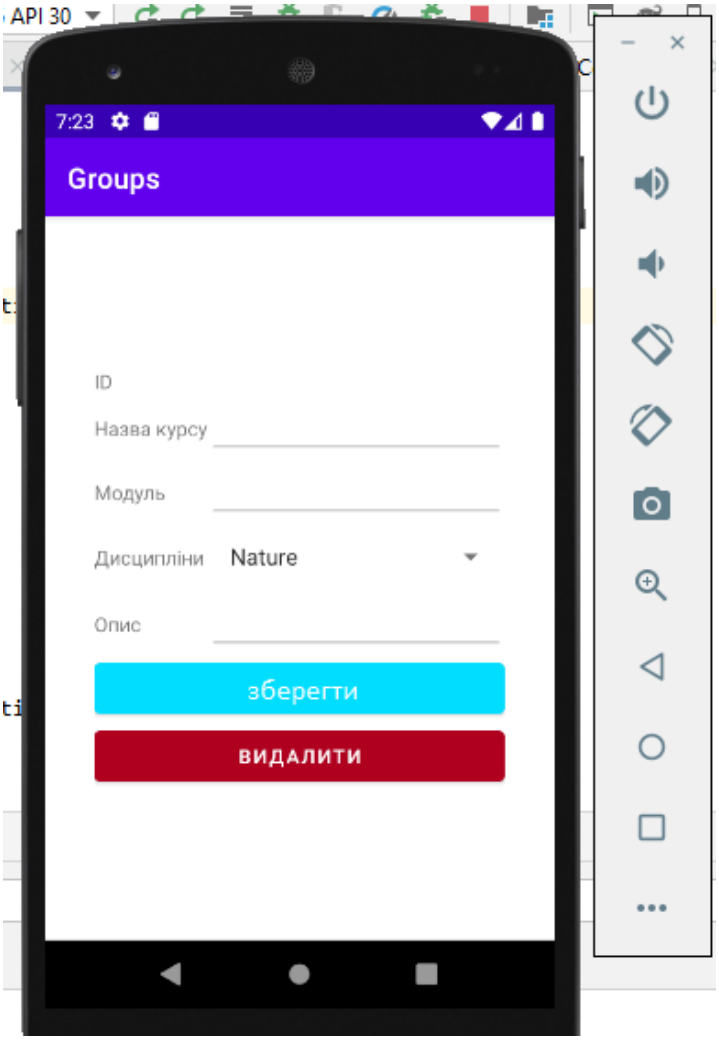


Рисунок 3.4 – Можливість редагування та створення даних

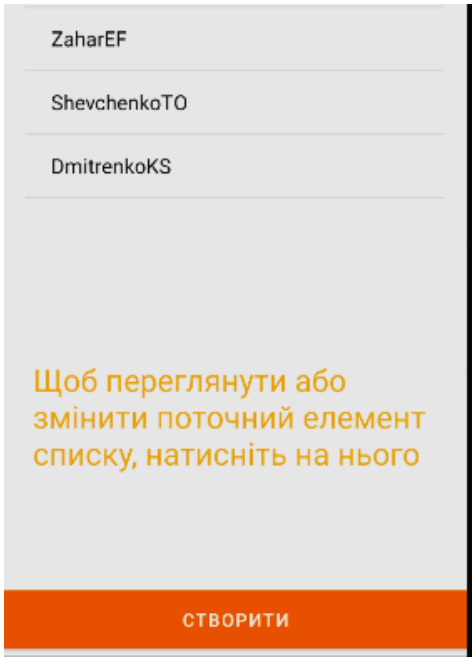


Рисунок 3.5 – Список користувачів автоматизованого електронного органайзера роботи викладачів навчальних курсів

3.3 Тестування інформаційної системи

Для перевірки роботи мобільного додатку автоматизованого електронного органайзера роботи викладачів було вирішено провести Test-case і перевірити, чи відповідає програмний продукт заданим вимогам (таблиця 3.1).

Таблиця 3.1 – Тестування функцій на критерії якості вимог

Критерії якості вимог	**Результат**
Завершеність Програмний продукт має містити повністю закінчений функціонал, щоб кожна функція уявляла з себе повноцінну реалізацію.	Відповідає частково, є перевірка на тип введених даних, але немає шифрування пароля, тобто він в БД лежить у відкритому доступі. Є деякі функції, які можна було б доробити При тому є функція для запобігання ситуації випадкового виходу. (рисунок 3.6)
Атомарність Розбиття програми на відповідні блоки, безпомилковість їх виводу. Кожна активність відповідає тільки за свій функціонал.	Тест пройдений, система не перенавантажена, що не дає можливості умістити зайву функцію там, де їй недоречно бути.
Несуперечливість, послідовність Послідовність функцій, створена система не дає збій через непродуманість. Стратегічно важливі кнопки та інші елементи відповідають логічній задумці з боку оформлення (колір, порядок розташування, місце)	Пройдено (рисунок 3.7)
Недвозначність Наявний текст повинен сприйматись лише у своєму прямому значенні, різні функції не повинні посилатись на 1 і той самий метод, якщо цього не вимагає ситуація.	Відносно порушено. Так як дана система включає в себе активність, яка одночасно є і формою для створення та редагування, що у свою чергу може призвести до збоїв в роботі з Базою даних. Але дана функція перевіряється в кодові по декілька разів на правильність формування запитів, тому збоїв в роботі не виникає. Є ще деякий недолік в тому, що немає підписів до того, що уявляє з себе та чи інакша форма.

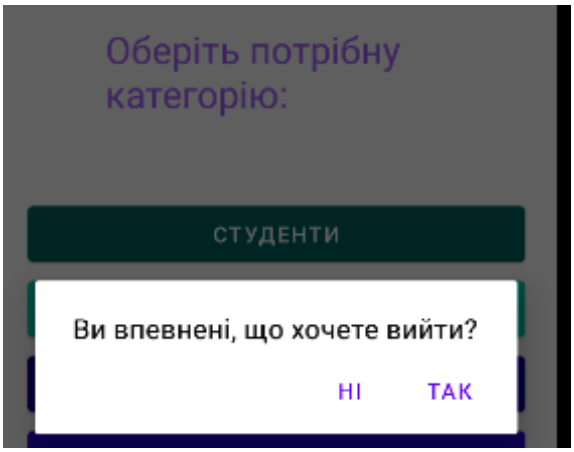


Рисунок 3.6 – Діалогове вікно автоматизованого електронного органайзера роботи викладачів навчальних курсів з вибором, якщо натиснути системну кнопку «Назад»

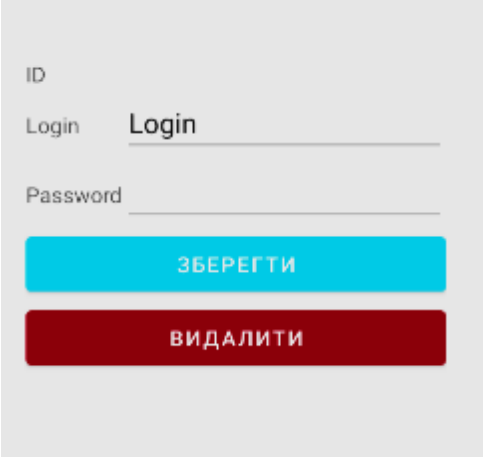


Рисунок 3.7. – Тест на послідовність та несуперечливість

У висновку по даному пункту можна підсумувати і сказати, що система ще вимагає додаткової налаштованості, але при цьому по пройденим тестам, система є робочою і в більшу сторону відповідає заданим вимогам.

3.4 Інструкція користувача до роботи з серверною та клієнтською частиною

Перш за все, користувач потрапляє на форму для входу в систему, де потрібно ввести свій логін та пароль (рисунок 3.8). Відповідно до введених

даних, він потрапить на форму з представленим для його ролі, функціоналом (рисунок 3.9).

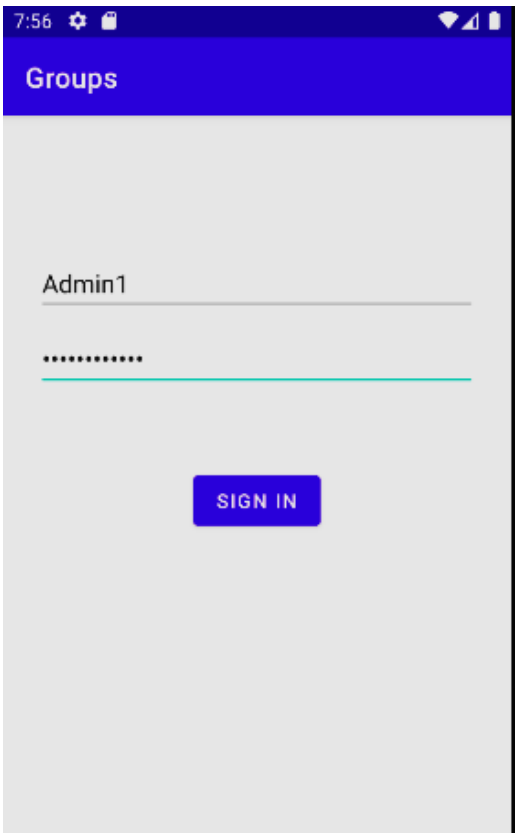


Рисунок 3.8 – Форма входу

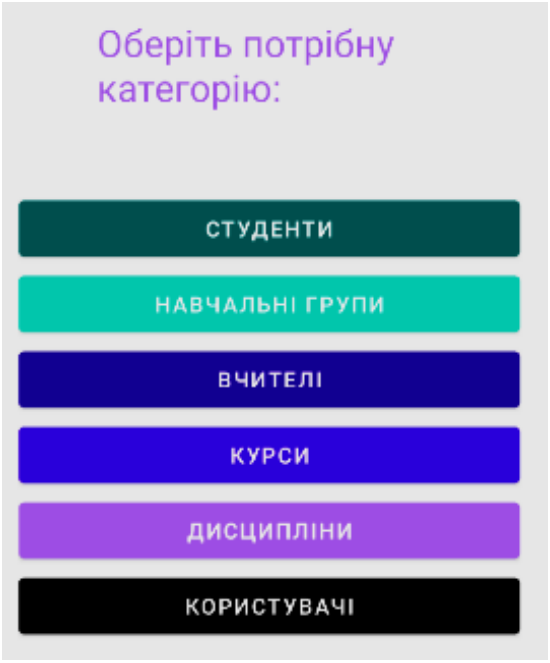


Рисунок 3.9 – Основний функціонал користувача

Відповідно до обраної категорії, користувач потрапляє на форму зі списком, звідки він може далі приступити до створення нового запису в БД, до редагування, видалення наявного або ж зробити звіт по певним атрибутам. (рисунки 3.10, 3.11).

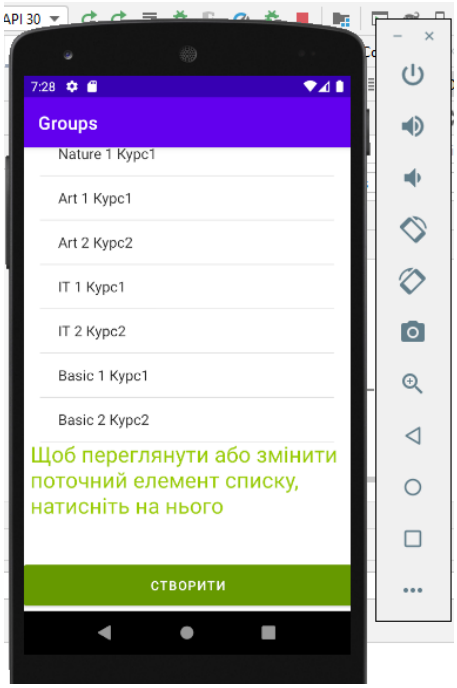


Рисунок 3.10 – Представлення вмісту таблиці у вигляді списку

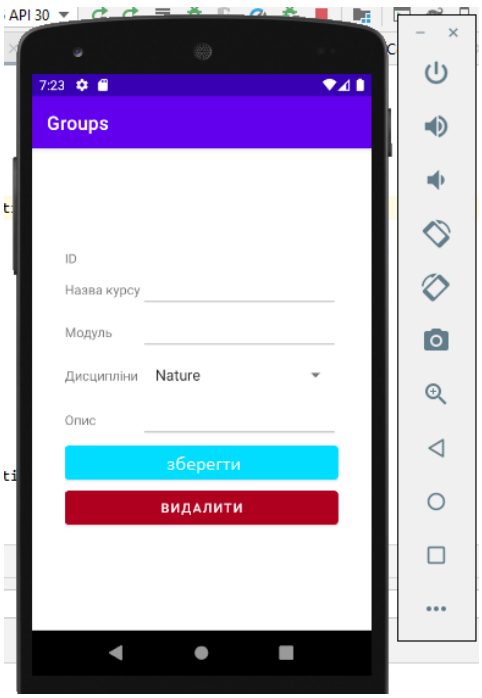


Рисунок 3.11 – Можливість редагувати, видаляти елементи списку

В залежності від поставлених задач, користувач обирає модуль, з яким буде працювати на даний момент.

У висновку по даному розділу можна підсумувати, що система ще вимагає додаткової налаштованості, але при цьому по пройденим тестам, система є робочою і відповідає заданим вимогам.

3.5 Вимоги до розгортання системи

Вимоги до апаратних засобів:

– мобільний пристрій повинен бути підключений до Інтернету;

– версія API не старше 21 (система виготовлена під 30 версію);

– розмір дисплею – не менше 5 дюймів по діагоналі;

– ємність оперативної пам’яті – 2Гб і більше;

– ємність вбудованої пам’яті – 16 Гб і більше;

– версія Android – не старше 6.

До програмних засобів:

– AndroidStudio версії 4;

– зовнішні бібліотеки для Екліпсу – Json і для Android Studio – Json і Okhttp;

– 64розрядна система ПК, кількість ядер: 4 і більше (для емулятора);

– Eclipse версії 4.17 і молодше;

– Java – 14 версія;

– відкритий порт 8000 на Сервері і доступний порт 3306 для СКБД.

Висновок

В ході виконання кваліфікаційної роботи бакалавра було отримано програмний продукт – автоматизований електронний органайзер роботи викладачів навчальних курсів на платформі Android. Для розробки автоматизованого електронного органайзера роботи викладачів навчальних курсів було використано програмне середовище AndroidStudio, мову програмування Java, систему керування базами даних MySQL і графічний клієнт для роботи з сервером MySQL Workbench.

Дана система може використовуватись у закладах з додатковою освітою, такі як: клуби, навчальні курси, тренінги. ПП за своїм функціоналом розрізняє 2 ролі користувачів: адміністратор і викладач. Відштовхуючись від ролі, під яким знаходиться особа, що увійшла в систему, реалізовано такі системи: робота з даними викладачів, учнів; створення, редагування, видалення, перегляд інформації навчальних курсів, груп, аудиторій, розкладів, тощо; долучення студентів до їх навчальних курсів; побудова планів навчання і відповідність їх реальному стану речей; ведення звітів відповідно відвідуваності учнів на заняттях.

Серед різних варіантів, де така система була б найзручнішою у користуванні, було обрано ОС Android. Адже в останні роки існує тенденція частого використання смартфону в усіх сферах. Ця система забезпечує постійний зв’язок, мобільність та доступ до Інтернету. Провідними лідерами мобільної техніки є MacOS і Android. Було обрано останній варіант через доступність і популярність. Отож, це говорить про те, що в працівників, швидше за все, буде на смартфоні встановлена саме ця ОС.

Розроблена система призначена для адміністраторів навчальних центрів, де проводиться надання інформаційних послуг клієнтам. Реалізований органайзер дає змогу адміністраторам контролювати роботу викладачів у проведенні занять, а також відмічати присутність учнів на заняттях.

Напрямками практичного використання даної програми є контроль виконання викладачами службових обов’язків: проведення занять, заміна інших викладачів, введення даних в систему та функціонал адміністратора, який може продивлятися наявну інформацію, формувати навчальні групи, реєструвати нових студентів, працівників, робочі розклади.

Перелік посилань

1. Wikipedia.Курси.URL: https://uk.wikipedia.org/wiki/Курси
2. Національне агентство кваліфікацій. Види освіти. .URL: https://nqa.gov.ua/vidi-osviti/
3. Педагогічний органайзер. Самоосвіта – складова професійного навчання. URL: http://metod-portfolio.blogspot.com/2020/11/blog-post_27.html
4. Академік. Органайзер, тлумачення слова. URL: https://dic.academic.ru/dic.nsf/ruwiki/1077129
5. Wikipedia.Навчальний_заклад.URL: https://uk.wikipedia.org/wiki/Навчальний_заклад
6. Wikipedia.Освіта.URL: https://uk.wikipedia.org/wiki/Освіта
7. Wikipedia.Ринок_освітніх_послуг.URL: https://uk.wikipedia.org/wiki/Ринок_освітніх_послуг
8. Wikipedia.Навчання.URL: https://uk.wikipedia.org/wiki/Навчання
9. Moyaosvita. Чим відрізняється вчитель від викладача. URL: https://moyaosvita.com.ua/osvita-2/chim-vidriznyayetsya-vchitel-vid-vikladacha/
10. Wikipedia.Педагогічний процес.URL: https://uk.wikipedia.org/wiki/Педагогічний_процес
11. Stud.com.ua.Додаткова освіта.URL:
12. https://stud.com.ua/46385/pedagogika/dodatkova_osvita
13. CRM «Мій клас». URL: https://moyklass.com/crm
14. CRM «Hollihop». URL: https://holyhope.ru/
15. Tproger. Java: коротке керівництво для початківців.URL: https://tproger.ru/translations/java-intro-for-beginners/
16. Лайв Тайпінг. На чому пишуть додатки під Android. URL: https://livetyping.com/ru/blog/na-chem-pishut-prilozhenija-pod-android
17. Dan.IT.Education. Розробка мобільних додатків від А до Я. URL: https://dan-it.com.ua/uk/rozrobka-mobilnih-dodatkiv-vid-a-do-ja-povnij-gajd/

18. Slaidik. Програми для розробки додатків: як зробити додаток для iOS і Android самостійно. URL: http://slaidik.com.ua/programi-dlya-rozrobki-dodatkiv-yak-zrobiti-dodatok-dlya-ios-i-android-samostijno/

19. PPT online. Мобільні операційні системи. URL : https://ppt-online.org/171133

20. PPT online. Інструменти і середовища розробки мобільних додатків.URL: https://ppt-online.org/341525

21. AndroidStudio. URL: https://developer.android.com/

22. Arduino+. Android Studio: середовище розробки мобільних додатків. URL: https://arduinoplus.ru/android-studio/

23. Android App. Топ-3 середовища розробки для Android. URL: https://app-android.ru/blog/environment-develop-android

24. Proglib. IDE Eclipse: за і проти від провідних програмістів. URL: https://proglib.io/p/ide-eclipse-za-i-protiv-ot-vedushchih-programmistov-2019-11-0

25. Geertjan's Blog. Офіційна підтримка PhoneGap в NetBeans IDE. URL: https://blogs.oracle.com/geertjan/official-support-for-phonegap-in-netbeans-ide

26. JAVARUSH. Eclipse, NetBeans чи IntelliJ IDEA? Обираємо IDE для Java-розробки. URL: https://javarush.ru/groups/posts/1642-eclipse-netbeans-ili-intellij-idea-vihbiraem-ide-dlja-java-razrabotki

27. Tiobe. Індекс Tiobe в 2021 році. URL: https://www.tiobe.com/tiobe-index/

28. Wikipedia. Java.URL: https://uk.wikipedia.org/wiki/Java

29. Плюси і мінуси. Мова Java – плюси і мінуси. URL: https://plusiminusi.ru/yazyk-java-plyusy-i-minusy/

30. Metanit. Введення в мову Kotlin. URL:https://metanit.com/kotlin/tutorial/1.1.php

31. MDM. Kotlin для додатків на Android : плюси і мінуси мови. URL: https://mdm.ooo/ru/kotlin-for-android-apps-pros-and-cons-of-programming-language/

32. Itmo.news. Чому Kotlin так полюбили в Google і кому потрібні 2000 мов програмування. URL: https://news.itmo.ru/ru/science/it/news/6683/

33. Хабр. Kotlin: мова програмування як продукт. URL: https://habr.com/ru/company/productsense/blog/526238/

34. Версія. Microsoft SQL Server. URL:http://www.versiya.com/ua/soft/microsoft/sql-server.html

35. Microsoft. Редактор запитів в SQL Server Management Studio (SSMS). .URL:https://docs.microsoft.com/ru-ru/sql/ssms/f1-help/database-engine-query-editor-sql-server-management-studio?view=sql-server-ver15

36. Microsoft office. Microsoft Access. URL: https://microffice.net/access.html#:~:text=Microsoft Access

37. BizzApps. Mecrosoft Access. URL: https://bizzapps.ru/p/microsoft-access/

38. Wikipedia. MySQL.URL: https://ru.wikipedia.org/wiki/MySQL

39. Proglib.Топ-10 систем управління базами даних в 2019 році. URL:https://proglib.io/p/databases-2019

40. Depix. Система управління базами даних MySQL. URL: https://depix.ru/articles/sistema_upravleniya_bazami_dannyh_mysql

41. Metanit.com. Графічний клієнт MySQL Workbench.URL: https://metanit.com/sql/mysql/1.3.php

42. MySQL.MySQL Workbench. URL: https://www.mysql.com/products/workbench/

ДОДАТКИ

Додаток А

Програмний код основних модулів автоматизованого електронного органайзера роботи викладачів навчальних курсів

Серверна частина:

```
package server_sg;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.io.OutputStreamWriter;
import java.io.Writer;
import java.net.InetSocketAddress;
import java.security.NoSuchAlgorithmException;
import java.sql.Connection;
import java.sql.Date;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;

import com.google.gson.Gson;
import com.google.gson.GsonBuilder;
import com.mysql.cj.ParseInfo;
import com.sun.net.httpserver.Headers;
import com.sun.net.httpserver.HttpExchange;
import com.sun.net.httpserver.HttpHandler;
import com.sun.net.httpserver.HttpServer;

public class server {
public static void main(String[] args) throws Exception {
	HttpServer server = HttpServer.create(new InetSocketAddress(8000),0);
	server.createContext("/login", new LoginHandler());
	server.createContext("/");
	server.createContext("/users", new UsersHandler());
	server.createContext("/groups", new StudyGroupHandler());
	server.createContext("/disciplines", new DisciplineHandler());
	server.createContext("/courses", new StudyCourseHandler());
	server.createContext("/saveCourse", new CreateCourseHandler());
	server.createContext("/updateCourse", new UpdateCourseHandler());
	server.createContext("/deleteCourse", new DeleteCourseHandler());
	server.createContext("/students", new StudentsHandler());
	server.setExecutor(null);
	server.start();
}

static class LoginHandler implements HttpHandler
{
	@Override
	public void handle(HttpExchange arg0) throws IOException {
		users user;
		Map<String, String> params = queryToMap(arg0.getRequestURI().getQuery());
```

```
			try
			{

				user=getUser(params.get("login"), params.get("password"));
				if(user==null)
				{
					String response = "User not found";
					arg0.sendResponseHeaders(401, response.length());
					OutputStream os = arg0.getResponseBody();
					os.write(response.getBytes());
					os.close();
				}
				else
				{
					Gson gson = new Gson();
					String response = gson.toJson(user);
					Headers responseHeaders = arg0.getResponseHeaders();
					responseHeaders.set("Content-Type", "application/json");
					arg0.sendResponseHeaders(200, response.length());
					OutputStream os = arg0.getResponseBody();
					os.write(response.getBytes());
					os.close();
				}
			}
			catch (ClassNotFoundException | NoSuchAlgorithmException | SQLException e) {
				String response = e.toString();
				arg0.sendResponseHeaders(500, response.length());
				OutputStream os = arg0.getResponseBody();
				os.write(response.getBytes());
				os.close();
			}

		}
}
static class UsersHandler implements HttpHandler
{
		@Override
		public void handle(HttpExchange arg0) throws IOException {
			users []pers;
		//	Map<String, String> params=queryToMap(arg0.getRequestURI().getQuery());
			try
			{
				pers=getUsers();
				if(pers==null)
				{
					String response = "Users not found";
					arg0.sendResponseHeaders(401, response.length());
					OutputStream os = arg0.getResponseBody();
					os.write(response.getBytes());
					os.close();
				}
				else
				{
					Gson gson = new Gson();
					String response = gson.toJson(pers);
					Headers responseHeaders = arg0.getResponseHeaders();
					responseHeaders.set("Content-Type", "application/json;
charset=UTF-8");
					arg0.sendResponseHeaders(200, 0);
					Writer os = new OutputStreamWriter(arg0.getResponseBody(),
"UTF-8");
					os.write(response);
```

```
                        os.close();
                }
            }
            catch (ClassNotFoundException | NoSuchAlgorithmException | SQLException e) {
                String response = e.toString();
                arg0.sendResponseHeaders(500, response.length());
                OutputStream os = arg0.getResponseBody();
                os.write(response.getBytes());
                os.close();
            }

      }
}
static class StudyGroupHandler implements HttpHandler
{
      @Override
      public void handle(HttpExchange arg0) throws IOException {
            StudyGroup[] groups;
            try {
                groups=getGroups();
                if(groups==null)
                {
                    String response = "Groups not found";
                    arg0.sendResponseHeaders(401, response.length());
                    OutputStream os = arg0.getResponseBody();
                    os.write(response.getBytes());
                    os.close();
                }
                else
                {
                //    Gson gson = new GsonBuilder().setDateFormat("dd-MMM-
yyyy").create();
                    Gson gson = new Gson();
                    String response = gson.toJson(groups);
                    Headers responseHeaders = arg0.getResponseHeaders();
                    responseHeaders.set("Content-Type", "application/json;
charset=UTF-8");
                    arg0.sendResponseHeaders(200, 0);
                    Writer os = new OutputStreamWriter(arg0.getResponseBody(),
"UTF-8");
                    os.write(response);
                    os.close();
                }
            }
            catch (ClassNotFoundException | SQLException e) {
                String response = e.toString();
                arg0.sendResponseHeaders(500, response.length());
                OutputStream os = arg0.getResponseBody();
                os.write(response.getBytes());
                os.close();

            }

      }

      private StudyGroup[] getGroups() throws ClassNotFoundException, SQLException
      {
            ArrayList<StudyGroup>group=new ArrayList<StudyGroup>();
            Connection connection = createConnection();
            String query;
            query= "SELECT * FROM study_group;";
            Statement stmt = connection.createStatement();
            ResultSet rs = stmt.executeQuery(query);
```

```
			while (rs.next())
			{
				StudyGroup groupItem = new StudyGroup();
				groupItem.Id = rs.getInt("Id");
				groupItem.Name = rs.getString("Name");
				groupItem.FK_StudyCourseId = rs.getInt("FK_StudyCourseId");
				groupItem.DateStart = rs.getDate("DateStart");
				groupItem.DateEnd = rs.getDate("DateEnd");
				groupItem.FK_TimetableId = rs.getInt("FK_TimetableId");
				groupItem.FK_TeachersgId = rs.getInt("FK_TeachersgId");
				groupItem.FK_AuditoryId = rs.getInt("FK_AuditoryId");
				group.add(groupItem);
			}
			connection.close();
			return group.toArray(new StudyGroup[group.size()]);
		}
}
static class DisciplineHandler implements HttpHandler
{
		@Override
		public void handle(HttpExchange arg0) throws IOException {
			Discipline[] disciplines;
			try {
				disciplines=getDisciplines();
				if(disciplines==null)
				{
					String response = "Disciplines not found";
					arg0.sendResponseHeaders(401, response.length());
					OutputStream os = arg0.getResponseBody();
					os.write(response.getBytes());
					os.close();
				}
				else
				{
					Gson gson = new Gson();
					String response = gson.toJson(disciplines);
					Headers responseHeaders = arg0.getResponseHeaders();
					responseHeaders.set("Content-Type", "application/json;
charset=UTF-8");
					arg0.sendResponseHeaders(200, 0);
					Writer os = new OutputStreamWriter(arg0.getResponseBody(),
"UTF-8");
					os.write(response);
					os.close();
				}
			}
			catch (ClassNotFoundException | SQLException e) {
				String response = e.toString();
				arg0.sendResponseHeaders(500, response.length());
				OutputStream os = arg0.getResponseBody();
				os.write(response.getBytes());
				os.close();

			}

		}

		private Discipline[] getDisciplines() throws ClassNotFoundException, SQLException
		{
			ArrayList<Discipline>disciplines=new ArrayList<Discipline>();
			Connection connection = createConnection();
			String query;
```

```
			query= "SELECT * FROM discipline;";
			Statement stmt = connection.createStatement();
			ResultSet rs = stmt.executeQuery(query);

			while (rs.next())
			{
				Discipline dItem = new Discipline();
				dItem.Id = rs.getInt("Id");
				dItem.Name = rs.getString("Name");
				dItem.StartAge = rs.getInt("StartAge");
				dItem.EndAge = rs.getInt("EndAge");
				dItem.Desctiption= rs.getString("Description");
				disciplines.add(dItem);
			}
			connection.close();
			return disciplines.toArray(new Discipline[disciplines.size()]);
		}
}
static class StudyCourseHandler implements HttpHandler
{
		@Override
		public void handle(HttpExchange arg0) throws IOException {
			StudyCourse[] sc;
			Map<String, String> params = queryToMap(arg0.getRequestURI().getQuery());
			try {
				sc=getStudyCourses();
				if(sc==null)
				{
					String response = "Study course not found";
					arg0.sendResponseHeaders(401, response.length());
					OutputStream os = arg0.getResponseBody();
					os.write(response.getBytes());
					os.close();
				}
				else
				{
					Gson gson = new Gson();
					String response = gson.toJson(sc);
					Headers responseHeaders = arg0.getResponseHeaders();
					responseHeaders.set("Content-Type", "application/json;
charset=UTF-8");
					arg0.sendResponseHeaders(200, 0);
					Writer os = new OutputStreamWriter(arg0.getResponseBody(),
"UTF-8");
					os.write(response);
					os.close();
				}
			}
			catch (ClassNotFoundException | SQLException e) {
				String response = e.toString();
				arg0.sendResponseHeaders(500, response.length());
				OutputStream os = arg0.getResponseBody();
				os.write(response.getBytes());
				os.close();

			}

		}

		private StudyCourse[] getStudyCourses() throws ClassNotFoundException,
SQLException
		{
			ArrayList<StudyCourse>sc=new ArrayList<StudyCourse>();
```

```
            Connection connection = createConnection();
            String query;
            query= "SELECT * FROM study_course;";
            Statement stmt = connection.createStatement();
            ResultSet rs = stmt.executeQuery(query);

            while (rs.next())
            {
                  StudyCourse scItem = new StudyCourse();
                  scItem.Id = rs.getInt("Id");
                  scItem.Name = rs.getString("Name");
                  scItem.Degree_course = rs.getInt("Degree_course");
                  scItem.FK_DisciplineId = rs.getInt("FK_DisciplineId");
                  scItem.Description= rs.getString("Description");
                  sc.add(scItem);
            }
            connection.close();
            return sc.toArray(new StudyCourse[sc.size()]);
      }
}
static class CreateCourseHandler implements HttpHandler
{
            @Override
            public void handle(HttpExchange arg0) throws IOException {
                  StudyCourse[] sc;
                  Map<String, String> params =
queryToMap(arg0.getRequestURI().getQuery());
                              try
                              {
                        sc=createStudyCourses(params.get("Name"),
params.get("FK_DisciplineId"),params.get("Degree_course"),params.get("Description")) ;
                        if(sc==null)
                        {
                              String response = "Not created";
                              arg0.sendResponseHeaders(401, response.length());
                              OutputStream os = arg0.getResponseBody();
                              os.write(response.getBytes());
                              os.close();
                        }
                        else
                        {
                              Gson gson = new Gson();
                              String response = gson.toJson(sc);
                              Headers responseHeaders = arg0.getResponseHeaders();
                              responseHeaders.set("Content-Type", "application/json");
                              arg0.sendResponseHeaders(200, response.length());
                              OutputStream os = arg0.getResponseBody();
                              os.write(response.getBytes());
                              os.close();
                        }
                  }

                  catch (ClassNotFoundException | NoSuchAlgorithmException |
SQLException e) {
                        String response = e.toString();
                        arg0.sendResponseHeaders(500, response.length());
                        OutputStream os = arg0.getResponseBody();
                        os.write(response.getBytes());
                        os.close();
                  }

            }
```

```
            private static StudyCourse[] createStudyCourses(String N, String
Fk_id,String Deg, String Desc) throws ClassNotFoundException, SQLException,
NoSuchAlgorithmException
            {
                  Connection connection = createConnection();
                  int id = Integer.parseInt(Fk_id);
                  int D = Integer.parseInt(Deg);
            //    String q1 = "insert into study_course(Name, FK_DisciplineId,
Degree_course, Description) values (N'"+N+"', "+id+", "+D+", N'"+Desc+"');";
            //    Statement stmt = connection.createStatement();
            //    ResultSet rs = stmt.executeQuery(q1);
                  ArrayList<StudyCourse>sc=new ArrayList<StudyCourse>();
                  StudyCourse scitem=new StudyCourse();
                  PreparedStatement pr = connection.prepareStatement
            ("insert into study_course(Name, FK_DisciplineId, Degree_course,
Description) values(?,?,?,?)");
                   pr.setString(1, N);
                 pr.setInt(2, id);
                 pr.setInt(3, D);
                 pr.setString(4, Desc);
                 pr.executeUpdate();

            String q1 = "SELECT * FROM study_course;";
            Statement stmt = connection.createStatement();
            ResultSet rs = stmt.executeQuery(q1);
                        while(rs.next())
                        {
                        scitem.Id=rs.getInt("Id");
                        scitem.Name=rs.getString("Name");
                        scitem.FK_DisciplineId=rs.getInt("FK_DisciplineId");
                        scitem.Degree_course = rs.getInt("Degree_course");
                        scitem.Description=rs.getString("Description");
                        }
                  connection.close();
                  return sc.toArray(new StudyCourse[sc.size()]);
            }
      }
static class UpdateCourseHandler implements HttpHandler
{
            @Override
            public void handle(HttpExchange arg0) throws IOException {
                  StudyCourse[] sc;
                  Map<String, String> params =
queryToMap(arg0.getRequestURI().getQuery());
                              try
                              {
                        sc=updateStudyCourses(params.get("Id"),params.get("Name"),
params.get("FK_DisciplineId"),params.get("Degree_course"),params.get("Description")) ;
                        if(sc==null)
                        {
                              String response = "Not updated";
                              arg0.sendResponseHeaders(401, response.length());
                              OutputStream os = arg0.getResponseBody();
                              os.write(response.getBytes());
                              os.close();
                        }
                        else
                        {
                              Gson gson = new Gson();
                              String response = gson.toJson(sc);
                              Headers responseHeaders = arg0.getResponseHeaders();
                              responseHeaders.set("Content-Type", "application/json");
                              arg0.sendResponseHeaders(200, response.length());
```

```
                                OutputStream os = arg0.getResponseBody();
                                os.write(response.getBytes());
                                os.close();
                        }
                }

                catch (ClassNotFoundException | NoSuchAlgorithmException |
SQLException e) {
                        String response = e.toString();
                        arg0.sendResponseHeaders(500, response.length());
                        OutputStream os = arg0.getResponseBody();
                        os.write(response.getBytes());
                        os.close();
                }

        }
        private static StudyCourse[] updateStudyCourses(String ID, String N, String
Fk_id,String Deg, String Desc) throws ClassNotFoundException, SQLException,
NoSuchAlgorithmException
        {
                Connection connection = createConnection();
                int Id = Integer.parseInt(ID);
                int id = Integer.parseInt(Fk_id);
                int D = Integer.parseInt(Deg);

                ArrayList<StudyCourse>sc=new ArrayList<StudyCourse>();
                StudyCourse scitem=new StudyCourse();
                String query = "update study_course set Name = '"+N+"',
FK_DisciplineId = "+id+", Degree_course = "+D+", Description = '"+Desc+"'  where Id =
"+Id+";";
                PreparedStatement preparedStmt = connection.prepareStatement(query);

                preparedStmt.executeUpdate();


        String q1 = "SELECT * FROM study_course;";
        Statement stmt = connection.createStatement();
        ResultSet rs = stmt.executeQuery(q1);
                        while(rs.next())
                        {
                        scitem.Id=rs.getInt("Id");
                        scitem.Name=rs.getString("Name");
                        scitem.FK_DisciplineId=rs.getInt("FK_DisciplineId");
                        scitem.Degree_course = rs.getInt("Degree_course");
                        scitem.Description=rs.getString("Description");
                        }
                connection.close();
                return sc.toArray(new StudyCourse[sc.size()]);
        }
    }




static class DeleteCourseHandler implements HttpHandler
{
    @Override
    public void handle(HttpExchange arg0) throws IOException {
        StudyCourse[] sc;
        Map<String, String> params = queryToMap(arg0.getRequestURI().getQuery());
                        try
```

```
                        {
                    sc=deleteCourse(params.get("Id"));
                    if(sc==null)
                    {
                          String response = "Not updated";
                          arg0.sendResponseHeaders(401, response.length());
                          OutputStream os = arg0.getResponseBody();
                          os.write(response.getBytes());
                          os.close();
                    }
                    else
                    {
                          Gson gson = new Gson();
                          String response = gson.toJson(sc);
                          Headers responseHeaders = arg0.getResponseHeaders();
                          responseHeaders.set("Content-Type", "application/json");
                          arg0.sendResponseHeaders(200, response.length());
                          OutputStream os = arg0.getResponseBody();
                          os.write(response.getBytes());
                          os.close();
                    }
              }

              catch (ClassNotFoundException | NoSuchAlgorithmException| SQLException e) {
                    String response = e.toString();
                    arg0.sendResponseHeaders(500, response.length());
                    OutputStream os = arg0.getResponseBody();
                    os.write(response.getBytes());
                    os.close();
              }
        }

        private StudyCourse[] deleteCourse(String Id)throws ClassNotFoundException,
SQLException, NoSuchAlgorithmException
        {
              Connection connection = createConnection();
              int id = Integer.parseInt(Id);

              ArrayList<StudyCourse>sc=new ArrayList<StudyCourse>();

              String query="DELETE FROM study_course WHERE Id="+id+";";
              PreparedStatement preparedStatement = connection.prepareStatement(query);

              preparedStatement.executeUpdate();


              connection.close();
              return sc.toArray(new StudyCourse[sc.size()]);
        }

}

static class StudentsHandler implements HttpHandler
{
        @Override
        public void handle(HttpExchange arg0) throws IOException {
              Students[] sc;
              Map<String, String> params = queryToMap(arg0.getRequestURI().getQuery());
              try {
                    sc=getStudent();
                    if(sc==null)
                    {
                          String response = "student not found";
```

```
                        arg0.sendResponseHeaders(401, response.length());
                        OutputStream os = arg0.getResponseBody();
                        os.write(response.getBytes());
                        os.close();
                }
                else
                {
                        Gson gson = new Gson();
                        String response = gson.toJson(sc);
                        Headers responseHeaders = arg0.getResponseHeaders();
                        responseHeaders.set("Content-Type", "application/json;
charset=UTF-8");
                        arg0.sendResponseHeaders(200, 0);
                        Writer os = new OutputStreamWriter(arg0.getResponseBody(),
"UTF-8");
                        os.write(response);
                        os.close();
                }
            }
            catch (ClassNotFoundException | SQLException e) {
                  String response = e.toString();
                  arg0.sendResponseHeaders(500, response.length());
                  OutputStream os = arg0.getResponseBody();
                  os.write(response.getBytes());
                  os.close();

            }

      }

      private Students[] getStudent() throws ClassNotFoundException, SQLException
      {
            ArrayList<Students>sc=new ArrayList<Students>();
            Connection connection = createConnection();
            String query;
            query= "SELECT * FROM student;";
            Statement stmt = connection.createStatement();
            ResultSet rs = stmt.executeQuery(query);

            while (rs.next())
            {
                  Students sItem = new Students();
                  sItem.Id = rs.getInt("Id");
                  sItem.Name = rs.getString("Name");
                  sItem.b_day = rs.getDate("B_day");
                  sItem.FK_StudyGroupId= rs.getInt("FK_StudyGroupsId");
                  sItem.phone_stud= rs.getInt("Telephone");
                  sItem.Name_parent= rs.getString("Name_person");
                  sItem.phone_par= rs.getInt("Telephone_person");
                  sItem.Description = rs.getString("Description");
                  sc.add(sItem);
            }
            connection.close();
            return sc.toArray(new Students[sc.size()]);
      }
}

public static users[] getUsers() throws SQLException, ClassNotFoundException,
NoSuchAlgorithmException
{
      ArrayList<users> pers = new ArrayList<users>();
      Connection connection = createConnection();
      String query =  "SELECT * FROM users;";
```

```
	Statement stmt = connection.createStatement();
	ResultSet rs = stmt.executeQuery(query);

	while (rs.next())
	{
		users userItem = new users();
		userItem.Id = rs.getInt("Id");
		userItem.Login = rs.getString("Login");
		userItem.Password = rs.getString("Password");

		pers.add(userItem);
	}
	connection.close();
	return pers.toArray(new users[pers.size()]);
}


public static users getUser(String login, String password) throws SQLException,
ClassNotFoundException, NoSuchAlgorithmException
{
	Connection connection = createConnection();
	String q1 = "select * from users where Login='"+login+"' and
Password='"+password+"';";
	Statement stmt = connection.createStatement();
	ResultSet rs = stmt.executeQuery(q1);

	users user=null;

	while(rs.next())
	{
		user=new users();
		user.Id=rs.getInt("Id");
		user.Login=rs.getString("Login");

	}

	connection.close();
	return user;
}
public static Connection createConnection() throws ClassNotFoundException, SQLException
{
	Connection connect;

	String driverName = "com.mysql.cj.jdbc.Driver";
	Class.forName(driverName);
	String serverName = "localhost";
	String mybase = "study_courses";
	String url_="jdbc:mysql://"+serverName+"/"+mybase;
	String userName = "root";
	String password = "1qazxsw20";
	connect = DriverManager.getConnection(url_, userName, password);

	return connect;
}
public static Map<String, String> queryToMap(String query)
{
	Map<String, String> result = new HashMap<String, String>();

	for(String param : query.split("&"))
	{
		String pair[] = param.split("=");
		if(pair.length>1)
		{
```

```
                result.put(pair[0], pair[1]);
            }
            else
            {
                result.put(pair[0], "");
            }
        }
        return result;
}

}
class users{
        int Id;
        String Login;
        String Password;
        public users()
        {
            Id = -1;
            Login = "";
            Password = "";
        }
}
class StudyGroup{
        int Id;
        String Name;
        int FK_StudyCourseId;
        Date DateStart;
        Date DateEnd;
        int FK_TimetableId;
        int FK_AuditoryId;
        int FK_TeachersgId;
}
class Discipline{
        int Id;
        String Name;
        int StartAge;
        int EndAge;
        String Desctiption;
}
class StudyCourse{
        int Id;
        String Name;
        int FK_DisciplineId;
        int Degree_course;
        String Description;
}
class Students
{
        int Id;
        String Name;
        Date b_day;
        int FK_StudyGroupId;
        int phone_stud;
        String Name_parent;
        int phone_par;
        String Description;
}
```

Клієнтська частина:

ActivityUsers.java

```
package com.example.groups;

import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListView;

import androidx.appcompat.app.AppCompatActivity;

import com.google.gson.Gson;

public class ActivityUsers extends AppCompatActivity {
    ListView listView;
    users[] person;
    static String[] userString;
    static String result = "";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_users);
        listView = (ListView)findViewById(R.id.listUsers);
        new UsersAsyncTask().execute("http://192.168.31.92:8000/users");
    }


    public void onCreateUser(View view) {
        Intent createUserIntent = new Intent(this, ChangeUser.class);
        createUserIntent.putExtra("edit", false);
        startActivity(createUserIntent);
    }

    private class UsersAsyncTask extends AsyncTask<String, Void, String> {
        @Override
        protected String doInBackground(String... params) {
            return MainActivity.GET(params[0]);
        }

        @Override
        protected void onPostExecute(String s) {
            Gson gson = new Gson();
            person = gson.fromJson(s, users[].class);
            userString = new String[person.length];
            for (int i = 0; i < person.length; i++) {
                userString[i] = " " + person[i].Login;
            }
            ArrayAdapter<String> adapter = new ArrayAdapter<String>(ActivityUsers.this,
android.R.layout.simple_list_item_1, userString);
            listView.setAdapter(adapter);
            listView.setOnItemClickListener(new AdapterView.OnItemClickListener()
            {
                @Override
                public void onItemClick(AdapterView<?> parent, View view, int position,
long id) {
                    Intent userChange = new Intent(ActivityUsers.this,
ChangeUser.class);

                    userChange.putExtra("edit", true);
                    userChange.putExtra("id", person[position].Id);
                    userChange.putExtra("login", person[position].Login.toString());
```

```
                    startActivity(userChange);
                }
            });

        }
    }
}
```

activity_users.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".ActivityUsers">

    <TextView
        android:id="@+id/textView"
        android:layout_width="312dp"
        android:layout_height="137dp"
        android:layout_marginBottom="36dp"
        android:text="@string/text_for_lists"
        android:textColor="@android:color/holo_orange_light"
        android:textSize="24sp"
        app:layout_constraintBottom_toTopOf="@+id/btnForCreate"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent" />

    <ListView
        android:id="@+id/listUsers"
        android:layout_width="324dp"
        android:layout_height="350dp"
        android:layout_marginBottom="8dp"
        app:layout_constraintBottom_toTopOf="@+id/textView"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <Button
        android:id="@+id/btnForCreate"
        android:layout_width="364dp"
        android:layout_height="57dp"
        android:onClick="onCreateUser"
        android:text="@string/new_"
        app:backgroundTint="@android:color/holo_orange_dark"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.454"
        app:layout_constraintStart_toStartOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Admin_panel.java

```
package com.example.groups;

import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;

import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;

public class Admin_panel extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_admin_panel);

    }
    @Override
    public void onBackPressed() {
        openQuitDialog();
    }

    private void openQuitDialog()
    {
        AlertDialog.Builder quitDialog = new AlertDialog.Builder(Admin_panel.this);
        quitDialog.setTitle("Ви впевнені, що хочете вийти?");
        quitDialog.setPositiveButton("Так", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                finish();
                Intent i = new Intent(Admin_panel.this, MainActivity.class);
                startActivity(i);
            }
        });
        quitDialog.setNegativeButton("Ні", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {

            }
        });

        quitDialog.show();
    }


    public void onClients(View view) {
        Intent clientIntent = new Intent(Admin_panel.this, ClientActivity.class);
        startActivity(clientIntent);
    }

    public void onGroups(View view) {
        Intent sgIntent = new Intent(Admin_panel.this, StudyGroupActivity.class);
        startActivity(sgIntent);
    }

    public void onTeachers(View view) {
        Intent tIntent = new Intent(Admin_panel.this, TeachersActivity.class);
        startActivity(tIntent);
    }

    public void onCourses(View view) {
        Intent courseIntent = new Intent(Admin_panel.this, CourseActivity.class);
```

```
        startActivity(courseIntent);
    }

    public void onDiscipline(View view) {
        Intent dIntent = new Intent(Admin_panel.this, DisciplineActivity.class);
        startActivity(dIntent);
    }

    public void onUsers(View view) {
        Intent userIntent = new Intent(Admin_panel.this, ActivityUsers.class);
        startActivity(userIntent);
    }
}
```

admin_panel.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.coordinatorlayout.widget.CoordinatorLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".Admin_panel">

    <TableLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="@color/white"
        android:gravity="center_horizontal"
        android:padding="20dp">

        <TableRow
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_weight="1"
                android:padding="50dp"
                android:text="@string/act_for_admin"
                android:textColor="@color/purple_200"
                android:textSize="24sp" />
        </TableRow>

        <TableRow
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <Button
                android:id="@+id/students"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_column="0"
                android:layout_weight="1"
                android:onClick="onClients"
                android:text="@string/clients"
                app:backgroundTint="@color/design_default_color_secondary_variant" />
        </TableRow>
```

```
<TableRow
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <Button
        android:id="@+id/groups"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_column="0"
        android:layout_weight="1"
        android:onClick="onGroups"
        android:text="@string/groups"
        app:backgroundTint="@color/teal_200" />
</TableRow>

<TableRow
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <Button
        android:id="@+id/teachers"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_column="0"
        android:layout_weight="1"
        android:onClick="onTeachers"
        android:text="@string/teachers"
        app:backgroundTint="@color/purple_700" />
</TableRow>

<TableRow
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <Button
        android:id="@+id/courses"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_column="0"
        android:layout_weight="1"
        android:onClick="onCourses"
        android:text="@string/courses"
        app:backgroundTint="@color/purple_500" />
</TableRow>
<!--
tema 7
-->
<TableRow
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <Button
        android:id="@+id/disciplines"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_column="0"
        android:layout_weight="1"
        android:onClick="onDiscipline"
        android:text="@string/disciplines"
        app:backgroundTint="@color/purple_200" />
</TableRow>

<TableRow
```

```
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <Button
                android:id="@+id/users"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_column="0"
                android:layout_weight="1"
                android:onClick="onUsers"
                android:text="@string/users"
                app:backgroundTint="@color/black" />
    </TableRow>

    </TableLayout>

</androidx.coordinatorlayout.widget.CoordinatorLayout>
```

ChangeCourse.java

```
package com.example.groups;

import android.app.Activity;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.TextView;
import android.widget.Toast;

import com.google.gson.Gson;

import java.io.IOException;
import java.security.NoSuchAlgorithmException;

import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.Response;

public class ChangeCourse extends Activity implements
AdapterView.OnItemSelectedListener
{
    //Поля для вводу інформації
    TextView idTV;
    static EditText Name;
    static EditText Degree;
    static EditText Description;
    static Spinner DiciplineC;

    //Значення для збереження
    static boolean flag = false;
    static int id;
    static int degree;
    static String name;
    static String description;
    static int FK_DesceplineId;
```

```
    // Модель даних дисциплін +масив значень Дисциплін
    static Discipline[] disciplines;
    static Courses[] courses;
    String[] disciplinesList;

    Button saveBtn;
    String res;

    static OkHttpClient client = new OkHttpClient();
    static Courses c;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_change_course);
        initializeComponent();
        getKeys();
        saveBtn = (Button) findViewById(R.id.btnSave);
        new CoursesAsyncTask().execute("http://192.168.31.92:8000/courses?");
        new DisciplineAsyncTask().execute("http://192.168.31.92:8000/disciplines?");
      if (flag) {
            idTV.setText("" + id);
            Name.setText(""+ name);

           Degree.setText(""+degree);
            Description.setText(""+description);
            DiciplineC.setEnabled(false);
        } else {
            idTV.setEnabled(false);
            Name.setText("");
            Degree.setText("");
            Description.setText("");
            DiciplineC.setEnabled(true);
        }
      Log.d("Input:", "It works#1");
    }




    @Override
    public void onItemSelected(AdapterView<?> parent, View view, int position, long id)
{
        desciplineItemChange();
    }

    @Override
    public void onNothingSelected(AdapterView<?> parent) {

    }

    // Метод для отримання існуючих дисциплін
    private class DisciplineAsyncTask extends AsyncTask<String, Void, Discipline[]> {
        @Override
        protected Discipline[] doInBackground(String... params) {
            String res = MainActivity.GET(params[0]);
            Gson gson = new Gson();
            disciplines = gson.fromJson(res, Discipline[].class);
            return disciplines;
        }
```

```
        @Override
        protected void onPostExecute(Discipline[] d) {
            disciplinesList = new String[disciplines.length];
            for (int i = 0; i < disciplines.length; i++) {
                disciplinesList[i] = disciplines[i].Name;
            }
            Log.d("Input:", "It works#3");
            ArrayAdapter<String> desceplinesAdapter = new
ArrayAdapter<String>(ChangeCourse.this, android.R.layout.simple_list_item_1,
disciplinesList);

desceplinesAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item
);
            DiciplineC.setAdapter(desceplinesAdapter);
           desciplineItemChange();

        }




    }
    // Метод для вибору дисципліни
    public void desciplineItemChange() {
        if (flag) {
            for (int i = 0; i < disciplines.length; i++) {
                if (FK_DesceplineId == disciplines[i].Id) {
                    DiciplineC.setSelection(i);
                    Log.d("Input:", "It works#4");
                }
            }
        }
        else
            DiciplineC.setSelection(0);
        DiciplineC.setEnabled(true);
        DiciplineC.setVisibility(View.VISIBLE);
    }
    // Метод для ініціалізації компонентів (прикріплення до змінних елементів)
    private void initializeComponent() {
        idTV = (TextView) findViewById(R.id.idShow);
        Name = (EditText) findViewById(R.id.nameET);
        Degree = (EditText) findViewById(R.id.degreeET);
        Description = (EditText) findViewById(R.id.descriptionET);
        DiciplineC = (Spinner) findViewById(R.id.disciplinesSpinner);

    }
    // Метод, для отримання інформації для редагування з попередньої форми
    private void getKeys() {

        flag = getIntent().getExtras().getBoolean("edit");

        if (flag) {
            id = getIntent().getExtras().getInt("id");
           name = getIntent().getExtras().getString("name");
            FK_DesceplineId = getIntent().getExtras().getInt("discipline");
           degree = getIntent().getExtras().getInt("degree");
            description = getIntent().getExtras().getString("description");
        }
    }

    public static boolean isNotEmpty(EditText Name, EditText degree)
```

```
    {
        int n=1;
        if(Name.getText().toString().trim().length()==0)
        {
            Name.setError("Поле Назва не може бути порожнім!");
            n=n*0;
            Log.d("Input:", "It works##5"+n);

        }
        if(degree.getText().toString().trim().length()==0)
        {
            degree.setError("Поле Модуль не може бути порожнім!");
            n=n*0;
            Log.d("Input:", "It works##5"+n);
        }


        if(n==1) {
            Log.d("Input:", "It works##5"+n);
            return true;
        }
        else return false;
    }

    public void btnOnSave(View view) throws NoSuchAlgorithmException {
        Log.d("Input:", "ItWorks#2");
        if(isNotEmpty(Name, Degree))
        {
            dataToRequest(courses, id);

        }
        else {

            Toast.makeText(getBaseContext(), "Заповніть, будь ласка, всі поля!",
Toast.LENGTH_LONG).show();
        }

    }
    public Courses dataToRequest(Courses[]C, int id) throws NoSuchAlgorithmException {
        c = new Courses();
        name = Name.getText().toString().trim();
        degree = Integer.parseInt(Degree.getText().toString().trim());
        description = Description.getText().toString().trim();

        Log.d("Input:", "ItWorks#6:"+name+degree+description+id);
        if (flag) {

            for (int i = 0; i < C.length; i++) {
                if (C[i].Id == id) {
                    c = C[i];
                    c.Name = name;
                    c.Degree_course = degree;
                    c.Description = description;
                    int desciplineIndex = DiciplineC.getSelectedItemPosition();
                    c.FK_DisciplineId = disciplines[desciplineIndex].Id;
                }
            }
            Log.d("Input:",
"ItWorks#7:"+c.Name+c.Degree_course+c.Description+c.FK_DisciplineId);
            Log.d("Input:",
"http://192.168.31.92:8000/updateCourse?Id="+id+"&Name="+name+"&FK_DisciplineId="+FK_Des
ceplineId+"&Degree_course="+degree+"&Description="+description+"");
            new
```

```
PostAsyncTask().execute("http://192.168.31.92:8000/updateCourse?Id="+id+"&Name="+name+"&
FK_DisciplineId="+FK_DesceplineId+"&Degree_course="+degree+"&Description="+description);

        } else {

            c.Name = name;
            c.Degree_course = degree;
            c.Description = description;
            Log.d("Input:",
"ItWorks#7:"+c.Name+c.Degree_course+c.Description+c.FK_DisciplineId);

Log.d("Input:","http://192.168.31.92:8000/saveCourse?Name="+name+"&FK_DisciplineId="+Dic
iplineC.getSelectedItemPosition()+"&Degree_course="+degree+"&Description="+description);

            new
PostAsyncTask().execute("http://192.168.31.92:8000/saveCourse?Name="+name+"&FK_Disciplin
eId="+DiciplineC.getSelectedItemPosition()+"&Degree_course="+degree+"&Description="+desc
ription);
            int desciplineIndex = DiciplineC.getSelectedItemPosition();
            c.FK_DisciplineId = disciplines[desciplineIndex].Id;

        }


        return c;
    }

    private class CoursesAsyncTask extends AsyncTask<String, Void, Courses[]> {
        @Override
        protected Courses[] doInBackground(String... params) {
            res = MainActivity.GET(params[0]);
            Gson gson = new Gson();
            courses = gson.fromJson(res, Courses[].class);
            return courses;
        }
    }
    private class PostAsyncTask extends AsyncTask<String, Void, String> {
        @Override
        protected String doInBackground(String... params) {
            try {
                return postChangeCourse(params[0]);
            } catch (IOException e) {
                e.printStackTrace();
            } catch (NoSuchAlgorithmException e) {
                e.printStackTrace();
            }
            return "";
        }

        @Override
        protected void onPostExecute(String s) {
            if (res.equals("User already exists"))
                Toast.makeText(getBaseContext(), "Користувач з таким логіном уже
існує!", Toast.LENGTH_SHORT).show();
            else
                Toast.makeText(getBaseContext(), "Зміни внесені успішно!",
Toast.LENGTH_LONG).show();
        }
    }

        public String postChangeCourse (String url) throws IOException,
NoSuchAlgorithmException {
            try{
```

```
                Request request = new Request.Builder()
                        .url(url)
                        .build();
                try (Response response = client.newCall(request).execute()) {

                    // Toast.makeText(getBaseContext(),
"Received!"+response.body().toString(), Toast.LENGTH_LONG).show();

                    Log.d("Input:", response.body().string());
                    return response.body().string();
                }
            } catch (IOException e)
            {

                return ""+e;
            }



        }

    public void btnOnDelete(View view) {
        finish();
        new
DeleteUserAsyncTask().execute("http://192.168.31.92:8000/deleteCourse?Id="+id);
    }


        private class DeleteUserAsyncTask extends AsyncTask<String, Void, String> {
        @Override
        protected String doInBackground(String... params) {
            try {
                return postChangeCourse(params[0]);
            } catch (IOException e) {
                e.printStackTrace();
            } catch (NoSuchAlgorithmException e) {
                e.printStackTrace();
            }
            return "";
        }
        @Override
        protected void onPostExecute(String s) {
            Toast.makeText(getBaseContext(), "Обраного користувача видалено!",
Toast.LENGTH_LONG).show();
        }
    }

}
```

activity_change_course.xml

```
   <?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".ChangeCourse">
    <TableLayout
        android:layout_width="wrap_content"
```

```
android:layout_height="wrap_content"
android:id="@+id/tableLayout2"
app:layout_constraintBottom_toBottomOf="parent"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toTopOf="parent">
<TableRow>
    <TextView
        android:text="@string/Id"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/idTV"/>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/idShow"/>
</TableRow>

<TableRow>
    <TextView
        android:text="@string/nameC"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/firstNameTV"/>
    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/nameET"
        android:inputType="text"
        android:text="Name"
        android:ems="10"/>
</TableRow>

<TableRow>
    <TextView
        android:text="@string/degree"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/degreeTV"/>
    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/degreeET"
        android:inputType="number"
        android:text="0"
        android:ems="10"/>
</TableRow>

<TableRow>
    <TextView
        android:text="@string/disciplines"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/disciplinesTV"/>
    <Spinner
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/disciplinesSpinner"
        android:layout_column="1"
        />
</TableRow>

<TableRow>
```

```
            <TextView
                android:id="@+id/descriptionTV"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:text="@string/description" />

            <EditText
                android:id="@+id/descriptionET"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:ems="10"
                android:gravity="start|top"
                android:inputType="textMultiLine" />

        </TableRow>

        <Button
            android:id="@+id/ButtonCreateOrChangeUser"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:onClick="btnOnSave"
            android:text="@string/change_"
            app:backgroundTint="@android:color/holo_blue_bright" />

        <Button
            android:id="@+id/buttonDeletUser"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:onClick="btnOnDelete"
            android:text="@string/delete"
            app:backgroundTint="@color/design_default_color_error" />

    </TableLayout>

</androidx.constraintlayout.widget.ConstraintLayout>
```

MainActivity.java

```
package com.example.groups;

import android.content.Intent;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import com.google.gson.Gson;

import java.io.IOException;

import okhttp3.OkHttpClient;
import okhttp3.Request;
```

```
import okhttp3.Response;

public class MainActivity extends AppCompatActivity {
    EditText login, password;
    String log, pass;
    int userID;
    static String result = "";
    static OkHttpClient client = new OkHttpClient();
    TextView text;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        initializeComponent();
    }
    private void initializeComponent()
    {
        login = (EditText)findViewById(R.id.username);
        password = (EditText)findViewById(R.id.password);
    }
    public boolean isConnected()
    {
        ConnectivityManager connMgr =
(ConnectivityManager)getSystemService(this.CONNECTIVITY_SERVICE);
        NetworkInfo networkInfo = connMgr.getActiveNetworkInfo();
        if(networkInfo!=null&&networkInfo.isConnected())
        {
            return  true;
        }
        else
        {
            return false;
        }
    }



    public static  String GET(String url){
        try{
            Request request = new Request.Builder()
                    .url(url)
                    .build();
            try (Response response = client.newCall(request).execute()) {
            //    Log.d("Input:", response.body().string());
                // Toast.makeText(getBaseContext(),
"Received!"+response.body().toString(), Toast.LENGTH_LONG).show();


                return response.body().string();
            }
        } catch (IOException e)
            {

                return ""+e;
            }


    }

    private class HttpAsyncTask extends AsyncTask<String, Void, String>
    {
        @Override
```

```
        protected String doInBackground(String... params) {
          //  Toast.makeText(getBaseContext(), "Received!"+params[0],
Toast.LENGTH_LONG).show();
        //    Log.d("Input:", params[0]);
                return GET(params[0]);

            }
        @Override
        protected void onPostExecute(String result) {

            Log.d("Input:", result);

            Gson gson = new Gson();

            users user = gson.fromJson(result, users.class);

            Log.d("Input:", user.Login);
            userID= user.Id;
            if(userID==1) {

                Log.d("Input:", result);
                Intent adminIntent = new Intent(MainActivity.this, Admin_panel.class);
                startActivity(adminIntent);
            }
            else
            {
                Toast.makeText(getBaseContext(), "Student", Toast.LENGTH_LONG).show();
            }
        }
    }
    public void onBtnAutorization(View view) throws InterruptedException {
        log = login.getText().toString().trim();
        pass=password.getText().toString().trim();
        if(isConnected())
        {
            String url ="http://192.168.31.92:8000/login?login="+log+"&password="+pass;
            Log.d("InputStream:",url);
            new HttpAsyncTask().execute(url);

        }
         else
       Toast.makeText(getBaseContext(), "NO connection", Toast.LENGTH_LONG).show();

    }
}
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <EditText
        android:id="@+id/username"
```

```
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="24dp"
        android:layout_marginTop="96dp"
        android:layout_marginEnd="24dp"
        android:hint="@string/prompt_login"
        android:inputType="text"
        android:selectAllOnFocus="true"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <EditText
        android:id="@+id/password"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="24dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="24dp"
        android:hint="@string/prompt_password"
        android:imeActionLabel="@string/action_sign_in_short"
        android:imeOptions="actionDone"
        android:inputType="textPassword"
        android:selectAllOnFocus="true"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/username" />

    <Button
        android:id="@+id/login"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="start"
        android:layout_marginStart="48dp"
        android:layout_marginTop="16dp"
        android:layout_marginEnd="48dp"
        android:layout_marginBottom="64dp"
        android:onClick="onBtnAutorization"
        android:text="@string/action_sign_in"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/password"
        app:layout_constraintVertical_bias="0.2" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Android_Manifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.groups">

    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />

    <uses-permission android:name="android.permission.INTERNET" />

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
```

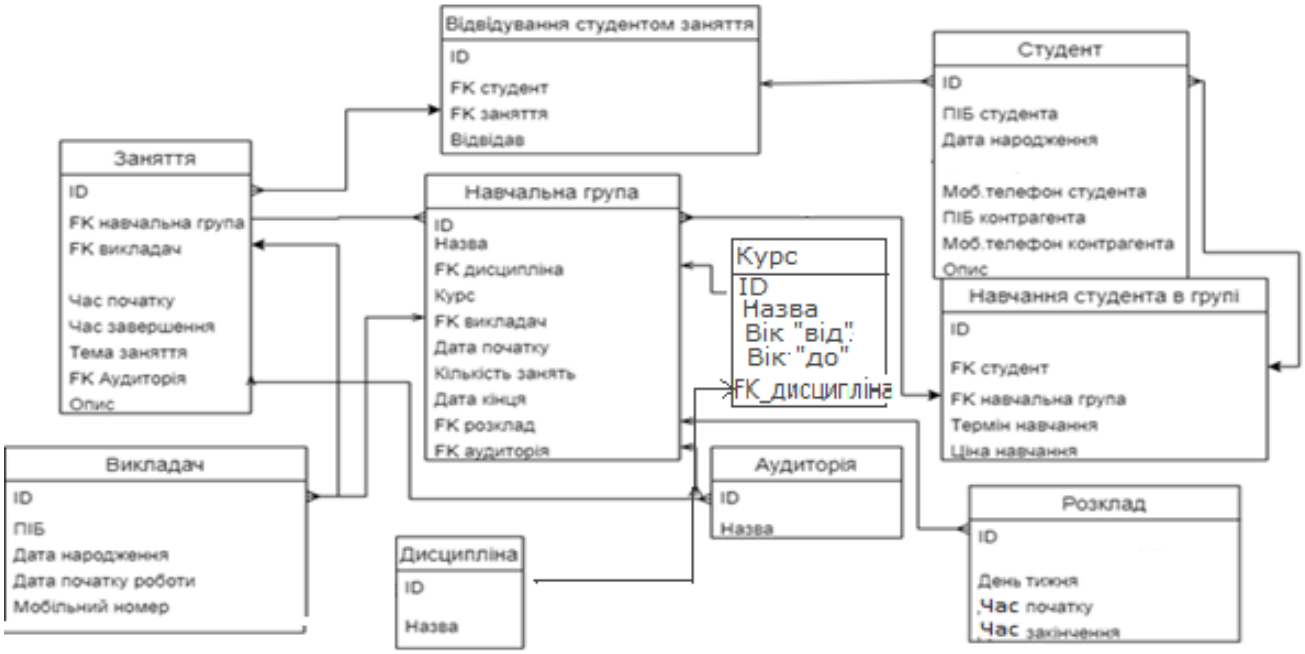
```
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Groups"
        android:usesCleartextTraffic="true">
        <activity android:name=".StudyGroupActivity"></activity>
        <activity android:name=".ClientActivity" />
        <activity android:name=".ChangeCourse" />
        <activity android:name=".CourseActivity" />
        <activity android:name=".ChangeUser" />
        <activity android:name=".ActivityUsers" />
        <activity
            android:name=".Admin_panel"
            android:label="@string/title_activity_admin_panel"
            android:theme="@style/Theme.Groups.NoActionBar" />
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

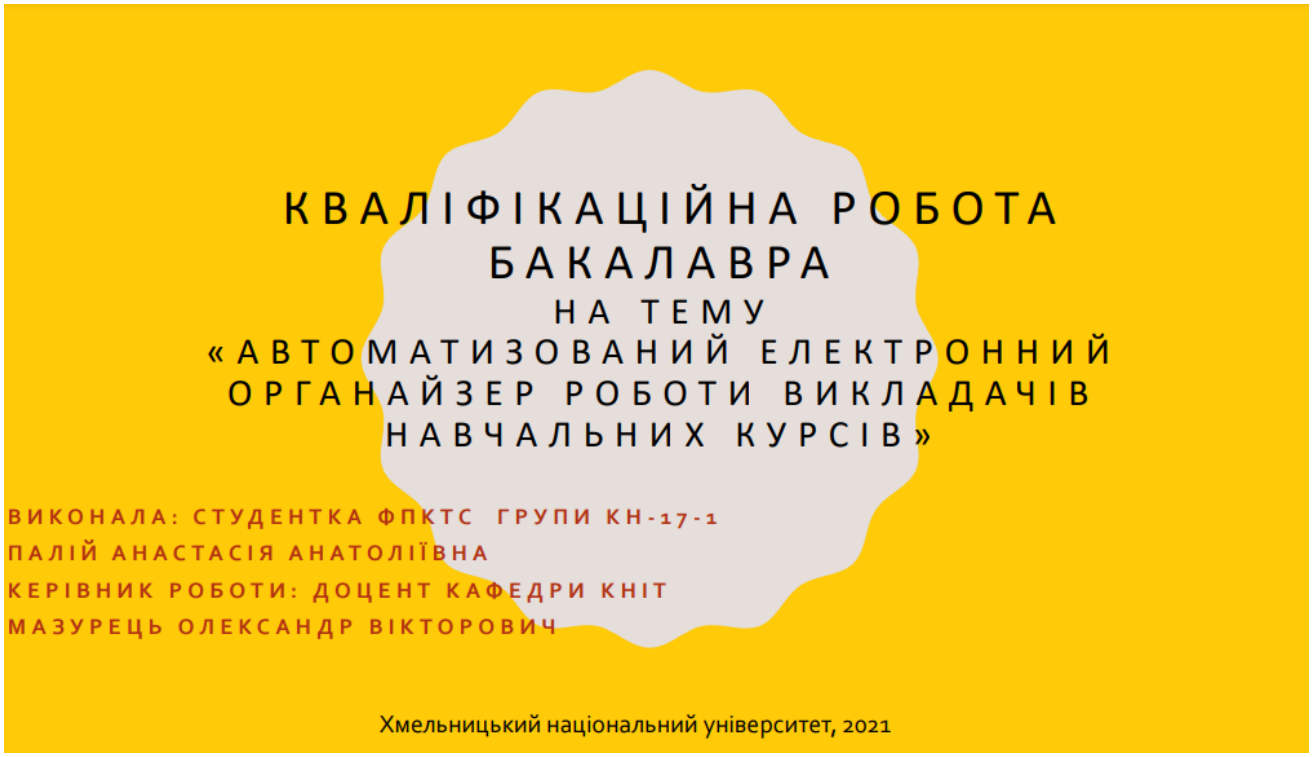
Додаток Б

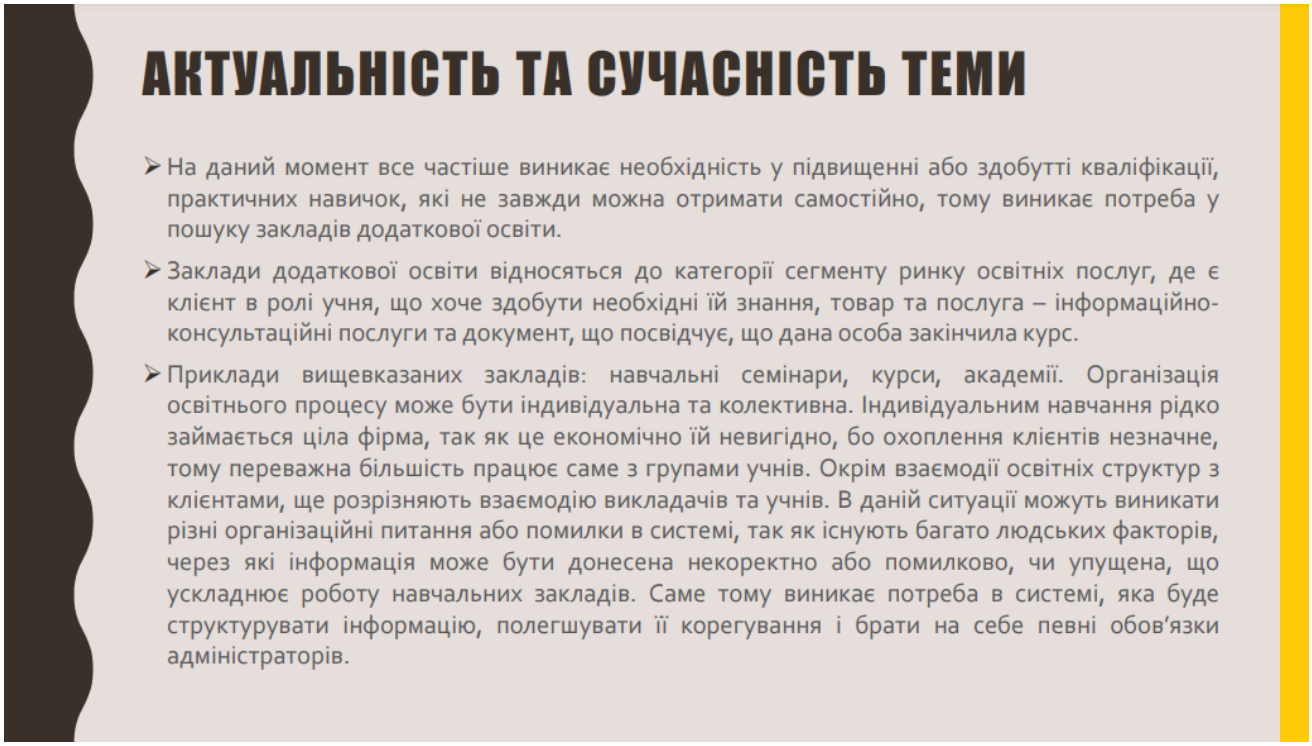
Структура бази даних автоматизованого електронного органайзеру роботи викладачів навчальних курсів



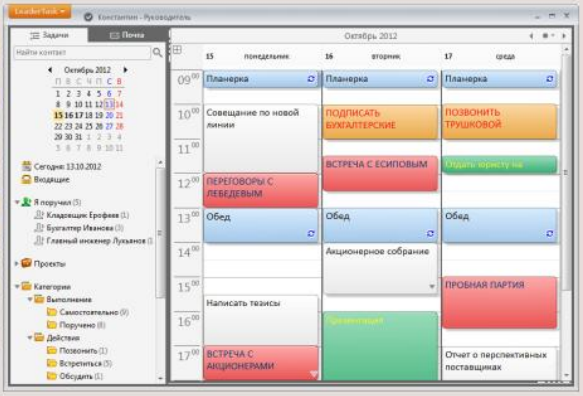
Додаток В

Презентаційний матеріал





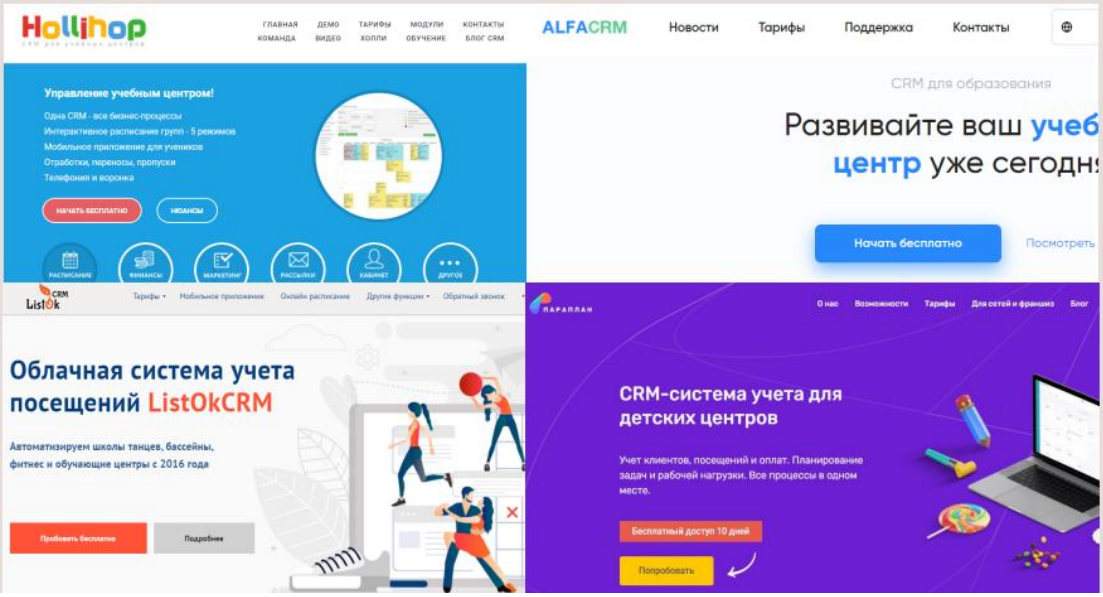
СУТЬ «ОРГАНАЙЗЕРУ» ТА НЕОБХІДНІСТЬ АВТОМАТИЗАЦІЇ СИСТЕМИ



Паперовий органайзер призначений для заповнення щоденних справ, планування часу

Електронний органайзер зі схожими функціями, який може розміщений на будь-якому електронному пристрої

ІСНУЮЧІ ІНФОРМАЦІЙНІ СИСТЕМИ



МЕТА ВИКОНАННЯ РОБОТИ

Завдання кваліфікаційної роботи: розробити автоматизований електронний органайзер для роботи викладачів навчальних курсів.

Система підходить для: закладів додаткової освіти

Користувачі: адміністратори навчальних центрів, старші викладачів, керівники вчительського складу, викладачі

Функції для адміністраторів: перегляд існуючої інформації, ввід необхідних даних, редагування, видалення, долучення студентів та викладачів до навчальних груп

Функції для викладачів: перегляд важливих даних, у тому числі: розклади, групи, що прив'язані до певного викладача, введення проведених занять, відмічання присутності учнів на заняттях

ОБРАНЕ ПЗ ДЛЯ СТВОРЕННЯ ПП

- Для розробки ПП в ході аналізу було обрано в якості СКБД **MySQL**, так як ця система є доволі проста і безкоштовна. Також є такі аспекти, як швидкість виконання запитів, популярність (а отже і велику кількість додактової інформації в мережі);
- Допоміжним інструментом для створення БД було обрано **MySQL Workbench** для візуалізації таблиць, наочне зображення зв'язків між таблицями.
- В якості мови програмування було обрано мову **Java**, яка володіє таким рядом переваг: стабільна популярність, кросплатформеність, є об'єктно-орієнтованою мовою програмування, обробляє асинхронні потоки, включає автоматичний збирач сміття та рішення для виключних ситуацій.
- ПЗ для розробки серверної частини проекту є **Eclipse**. Рідною мовою для нього є **Java**. Включає багато функцій, утиліт, автодоповнення та підсвітку помилок (до компіляції проекту), підтримує майже всі мови програмування, покрокове тестування, корисна документація.
- ПЗ для розробки клієтської частини є **Android Studio**. Незважаючи на вимогливість до апаратного забезпечення, зручний в користуванні, з можливістю покрокового тестування, з широким асортиментом інструментів, великою підтримкою товариства.

АРХІТЕКТУРА ІНФОРМАЦІЙНОЇ СИСТЕМИ

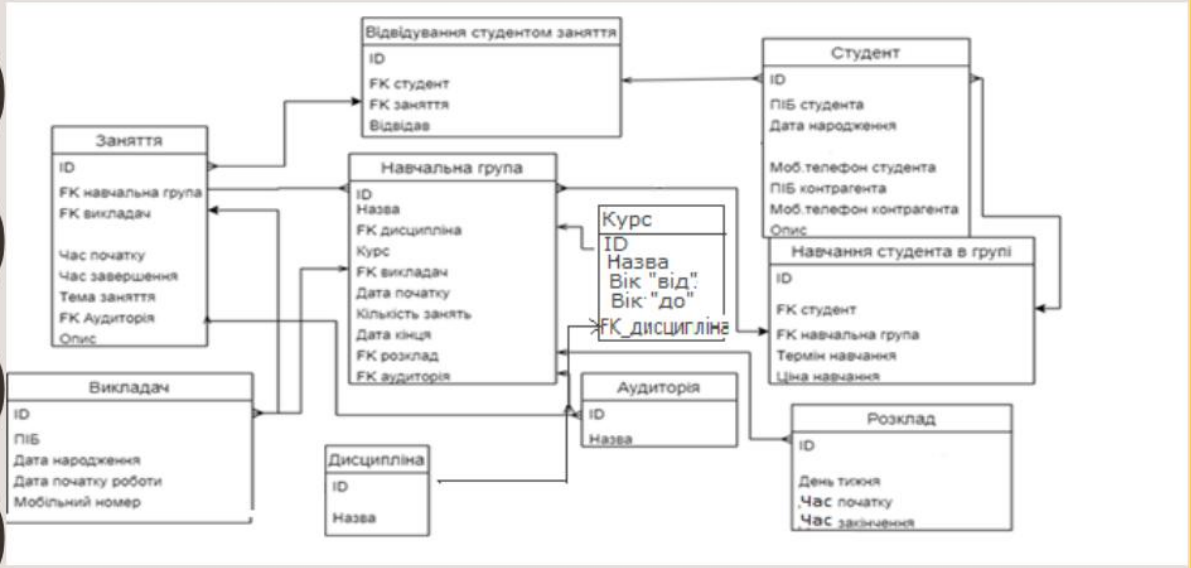
Автоматизований електронний органайзер роботи викладачів навчальних курсів

- робота з основними даними: викладачі, учні, навчальні групи, заняття;
- робота з додатковими даними: аудиторії, спеціальності, розклади
- ведення присутності учнів на заняттях;
- включення студентів у групи, прив'язка пов'язаних сутностей;
- планування зайнятості викладачів, аудиторій.

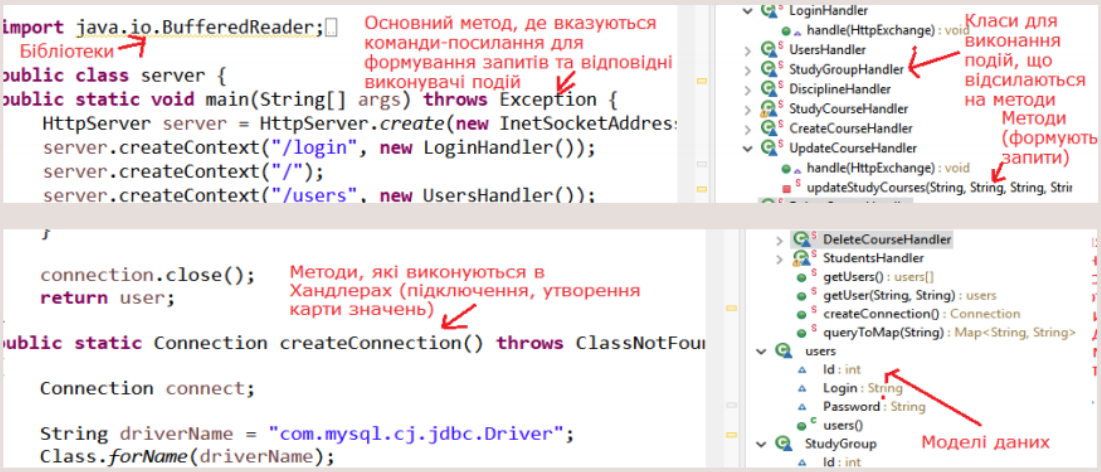
База даних інформаційної системи

- навчальні групи
- учні
- викладачі
- відвідуваність учнів
- аудиторії
- розклади
- заняття

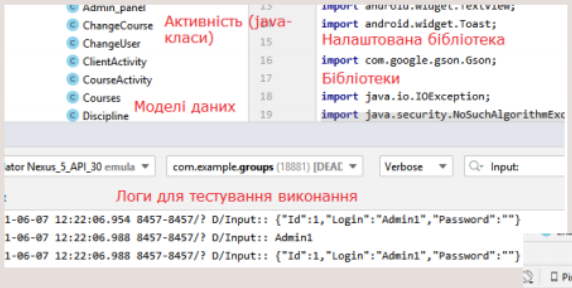
ДАТАЛОГІЧНА МОДЕЛЬ БАЗИ ДАНИХ



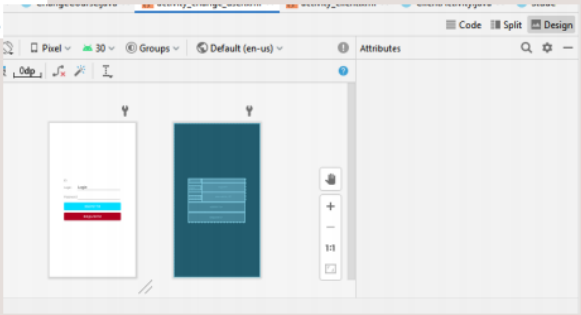
СТРУКТУРА СЕРВЕРНОЇ ЧАСТИНИ



СТРУКТУРА КЛІЄНТСЬКОЇ ЧАСТИНИ

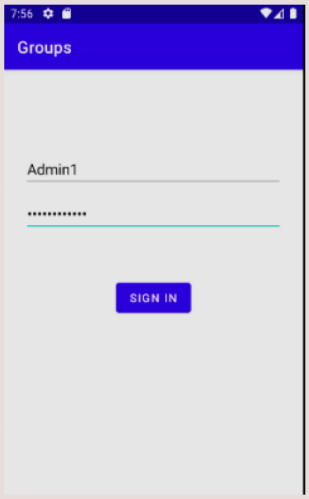


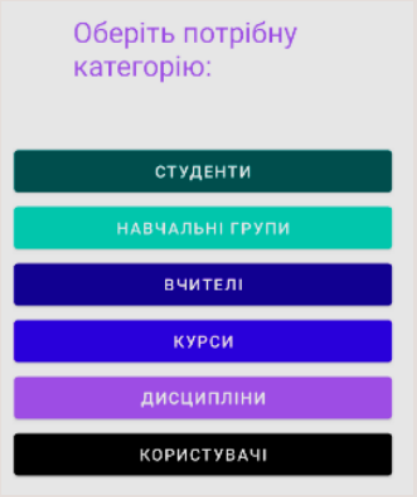
Java-класи (Активності і моделі даних) відсилають запити на Сервер, де обробляється код і повертається результат у вигляді переходів між різними активностями, або вспливаючих повідомлень.



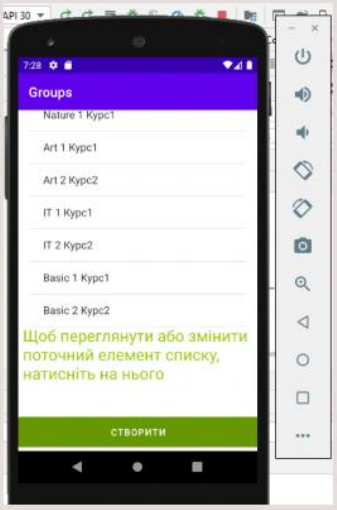
Візуальна частина для користувача у вигляді xml-розмітки, представлена у вигляді дезайнеру та відповідний код.

ГОЛОВНА ФОРМА ТА ФУНКЦІОНАЛ ДЛЯ АДМІНІСТРАТОРА

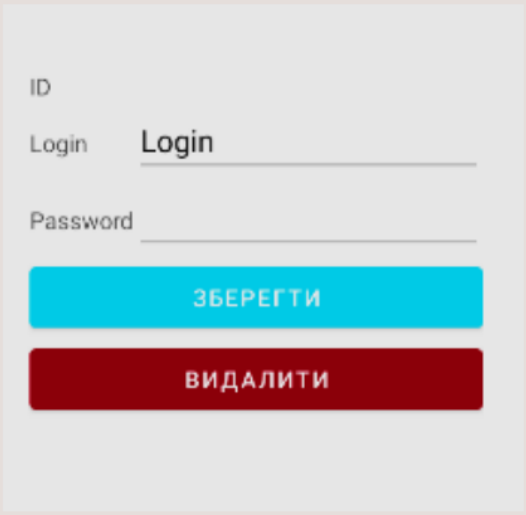


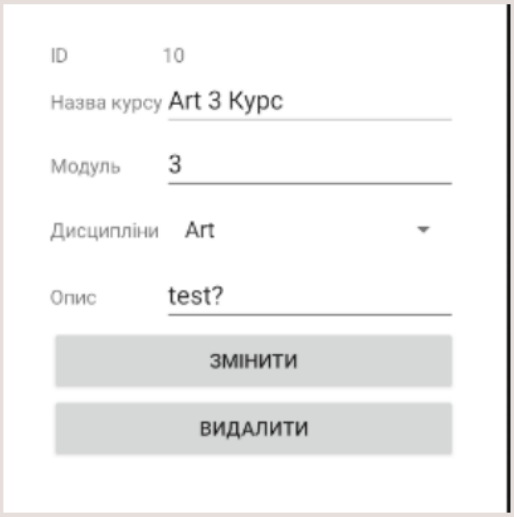


ФУНКЦІЇ ДЛЯ ВИКЛАДАЧА ТА ВИВІД ДАНИХ З БД



СТВОРЕННЯ, РЕДАГУВАННЯ ТА ВИДАЛЕННЯ СТАРИХ ОБ'ЄКТІВ





ВИСНОВОК

- ✓ В ході виконання кваліфікаційної роботи бакалавра було отримано програмний продукт – автоматизований електронний органайзер роботи викладачів навчальних курсів на платформі Android. Для розробки автоматизованого електронного органайзера роботи викладачів навчальних курсів було використано програмне середовище AndroidStudio, мову програмування Java, систему керування базами даних MySQL і графічний клієнт для роботи з сервером MySQL Workbench.
- ✓ Дана система може використовуватись у закладах з додатковою освітою, такі як: клуби, навчальні курси, тренінги. Система за своїм функціоналом розрізняє 2 ролі користувачів: адміністратор і викладач. Відштовхуючись від ролі, під яким знаходиться особа, що увійшла в систему, реалізовано такі системи: робота з даними викладачів, учнів; створення, редагування, видалення, перегляд інформації навчальних курсів, груп, аудиторій, розкладів, тощо; долучення студентів до їх навчальних курсів; побудова планів навчання і відповідність їх реальному стану речей; ведення звітів відповідно відвідуваності учнів на заняттях.
- ✓ Напрямками практичного використання даної програми є контроль **виконання викладачами** службових обов'язків: проведення занять, заміна інших викладачів, введення даних в систему та **функціонал адміністратора**, який може продивлятися наявну інформацію, формувати навчальні групи, реєструвати нових студентів, працівників, робочі розклади.

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ГОЛОВІ ЕКЗАМЕНАЦІЙНОЇ КОМІСІЇ

Направляється студент Палій А. А. на захист дипломного проекту (роботи)
(прізвище, ініціали)

за спеціальністю 122 - Комп'ютерні науки

На тему: Автоматизований електронний органайзер роботи викладачів навчальних курсів

Дипломний проект (робота), рецензія і довідка про перевірку на плагіат додаються.

Декан факультету _____ САВЕНКО О. С.
(підпис) (прізвище та ініціали)

ДОВІДКА УСПІШНОСТІ

Палій А. А. за період навчання на факультеті програмування та компютерних і телекомунікаційних систем з 2017 по 2021 роки. повністю виконав навчальний план спеціальності з такими розподілом оцінок за:
національною шкалою: відмінно 9,38 %, добре 43,75 %, задовільно 46,88 %.
шкалою ЄКТС: A 12,73 %, B 16,36 %, C 29,09 %, D 5,45 %, E 36,36 %.

Методист факультету _____ _____
(підпис) (прізвище та ініціали)

ВИСНОВОК КЕРІВНИКА ДИПЛОМНОГО ПРОЕКТУ (РОБОТИ) ТА ОБГРУНТУВАННЯ ОЦІНКИ

Студент Палій А. А. при виконанні кваліфікаційної роботи проявив себе як підготовлений та дисциплінований фахівець з ІТ, вчасно виконуючи всі завдання. Роботу виконано в актуальному напрямку ІТ, основні функції системи коректно імплементовані, пояснювальна записка в повній мірі відображає результати роботи.

Оцінка дипломного проекту (роботи) добре

Керівник дипломного прокту (роботи) _____ Мануренко О. В.
(підпис) (прізвище та ініціали)

"08" 06 2021 р.

ВИСНОВОК КАФЕДРИ ПРО ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ)

Дипломний проект (роботу) розглянуто. Студент Палій А. А. допускається до захисту цього проекту

Завідувач кафедри КМІТ
(назва)

_____ "08" 06 2021 р.
(підпис, прізвище, ініціали)

Thu Jun 10 18:06:40 EEST 2021, Петровський Сергій Степанович, Хмельницький національний університет, ХНУ

Anti-Plagiarism v-15.257

Максимальное совпадение с одним документом 45.0%

Словари проверки: en_US, ru_RU, ua_UA. **Ошибок в документах: 13%**

	Документ		Суммарное совпадение по Базе Данных	
ID: 93200 Название: Автоматизований електронний органайзер роботи викладачів навчальних курсів Добавлено в БД: 2021-06-10 Авторы: А.А. Палій Руководители: О.В. Мазурець Консультанты: Опоненты:	Символы	Лексемы	Символы	Лексемы
	55175	462	25204 (46%)	210 (45%)

Источник плагиата

ID	Описание	Наличие плагиата в документе	
		Символы	Лексемы
90946	Название: ЗВІТ з професійної практики Добавлено в БД: 2021-05-19 Авторы: Палій А.А. Руководители: Скрипник Т.К. Консультанты: Опоненты:	24596 (45.0%)	199 (43.0%)

UNICHECK

Ім'я користувача:
Кафедра КН

ID перевірки:
1008262551

Дата перевірки:
10.06.2021 18:57:27 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
10.06.2021 18:58:41 EEST

ID користувача:
100005671

Назва документа: **КРБ Палій 20210609 5 фінал Lite**

Кількість сторінок: **56** Кількість слів: **8747** Кількість символів: **63699** Розмір файлу: **2.37 MB** ID файлу: **1008333623**

Виявлено модифікації тексту (можуть впливати на відсоток схожості)

8.83%
Схожість

Найбільша схожість: **2.98%** з джерелом з Бібліотеки (ID файлу: **1008321746**)

6.1% Джерела з Інтернету	122	Сторінка 58
3.52% Джерела з Бібліотеки	54	Сторінка 59

0% Цитат

Вилучення цитат вимкнено

Вилучення списку бібліографічних посилань вимкнено

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Заміжені символи	2
Підозріле форматування	13 сторінок

РІШЕННЯ ЕКСПЕРНОЇ КОМІСІЇ

КАФЕДРИ КОМП'ЮТЕРНИХ НАУК ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатом звіту подібності щодо роботи, генерованого системою виявлення текстових збігів/ідентичності/схожості:

Назва: *Автоматизований органайзер роботи викладачів навчальних курсів*

Автор: *студентка групи КН-17-1 Палій Анастасія Анатоліївна*

Спеціальність: *122 – Комп'ютерні науки*

Освітня програма: *освітньо-професійна*

Науковий керівник: *доцент кафедри КНІТ Мазурець О.В.*

Після аналізу звіту подібності зроблено такий висновок:

№	Висновок	Позначка про відповідність
1	Запозичення, виявлені в роботі, є законними і не є плагіатом. Робота приймається до захисту.	*відповідає*
2	Виявлені запозичення не є плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої та електронної версії роботи	
3	Виявлені запозичення не є плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту (наступного року) після того як буде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат.	
4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	

Підтвердження:

Запозичення, виявлені в роботі, є законними і не є плагіатом, оскільки:

1) запозичення розміщені в розділах аналізу існуючих аналогів та прототипів, які не описують безпосередньо авторське дослідження і не стосуються результатів роботи;

2) усі запозичення фрагментарні;

3) до запозичень входять фрагменти програмного коду, що на мають авторства і містять поширені конструкції;

4) серед запозичень знаходяться загальновідомі терміни, скорочення та визначення.

Сумарний обсяг всіх запозичень, визначений системою виявлення збігів/ ідентичності/схожості, складає 8.83 % і адресується до першоджерел, що, з урахуванням наведених обґрунтувань, відповідає характеру дослідження і свідчить на користь кваліфікаційної роботи.

Керівник роботи _____ О. В. Мазурець

Гарант ОП _____ О. В. Мазурець

Завідувач кафедри КНІТ _____ О. В. Бармак

Хмельницький національний університет
Кафедра комп'ютерних наук та інформаційних технологій

РЕЦЕНЗІЯ

на кваліфікаційну роботу бакалавра

студента *гр. КН-17-1 Палій Анастасії Анатоліївни*

за темою *Автоматизований електронний органайзер для роботи викладачів навчальних курсів*

1. Актуальність і значення теми *Так як інформаційної системи – органайзери для роботи викладачів спрощують та організовують роботу навчальних курсів, вони є ефективним засобом для організації роботи таких закладів і дозволяють автоматизувати й структурувати організаційні процеси.*

2. Оцінка запропонованих моделей, підходів, алгоритмів, інформаційної складової та засобів розробки *Бізнес-процеси, що автоматизовані в даній системі, дозволяють: зчитувати, вводити та обробляти дані відповідно до ролей користувачів; долучати відвідувачів курсів та викладачів; організовувати розклади занять та відмічати присутність студентів на заняттях. При розробці інформаційної системи було використано платформу Java, яка була задіяна у двох середовищах: Eclipse та Android Studio. Створення та редагування баз даних було реалізовано за допомогою СКБД MySQL.*

3. Оцінка розробленої інформаційної системи, її практична цінність та економічна доцільність *Готове програмне рішення повністю відповідає поставленому завданню: забезпечує просту і наочну організацію інформаційної структури закладів додаткової освіти, а саме: зручне введення та корегування даних адміністратором, а для викладачів – перегляд інформації стосовно розкладів, занять, навчальних груп, відмічання занять та присутніх на них учнів.*

4. Загальний висновок та оцінка *Завдання виконано в повній мірі та на професійному рівні. За своєю структурою, поставленій меті та вирішеними завданнями робота відповідає вимогам, що пред'являються до освітньо-кваліфікаційного рівня «бакалавр», а студент, що виконав її, Палій А.А. заслуговує отримання ступеню бакалавра з комп'ютерних наук.*

Робота заслуговує на оцінку «добре».

Рецензент к.т.н., доц. Медведчук Я.К.