МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Тернопільський національний економічний університет
Факультет комп'ютерних інформаційних технологій
Кафедра комп'ютерних наук

КІЛЬЧИЦЬКИЙ Володимир Богданович

Android-додаток для підтримки роботи куратора групи/ Android-application for supporting the work of group's adviser

напрям підготовки: 6.050103 - Програмна інженерія
фахове спрямування - Програмне забезпечення систем

Бакалаврська дипломна робота

Виконав студент групи ПЗС-41
В. Б. Кільчицький

_____

Науковий керівник:
викладач ПОРПЛИЦЯ Н.П.

_____

Бакалаврську дипломну роботу
допущено до захисту:

"___" _____ 20___ р.

Завідувач кафедри
_____ **А. В. Пукас**

ТЕРНОПІЛЬ - 2016

РЕЗЮМЕ

Дипломна робота містить 92 сторінок, 13 таблиць, 47 рисунки, список використаних джерел із 13 найменувань, 2 додатки.

Метою дипломної роботи є розробка Android – додатку підтримки роботи куратора.

Об'єктом дослідження є процес роботи куратора групи у вищому навчальному закладі.

Предметом дослідження є застосування сучасних методів та засобів для розробки Android – додатку підтримки роботи куратора групи у вищому навчальному закладі.

Методи розробки базуються на технології Android Studio на мові програмування java, сервер бази даних MSSQL і Web-сервер Apache.

Одержані результати полягають в розробці Android – додатку допомоги роботи куратора.

Ключові слова: Android-додаток, робота куратора, стимулювати студентів, ТНЕУ.

SUMMARY

Thesis contains 92 pages, 13 tables, 47 figures, list of sources with 13 titles, 2 application.

The aim of the thesis is the development of Android - application to support the work supervisor.

Object of research a process of group supervisor in higher education.

The subject of research is the use of modern methods and tools for developing Android - application to support the work of the curator in higher education.

Methods based on Android technology Studio in the programming language java, MSSQL database server and the Web server Apache.

The resulting is creating Android-application assistance work of the curator.

Keywords: Android application, the work of the curator, stimulate students, TNEU.

ЗМІСТ

ВСТУП

Актуальність теми

Викладач який перебуває на посаді куратора академічної групи повинен стимулювати студентів до саморозвитку та самонавчання. Доступ до системи де знаходяться всі оцінки, кожного з його студентів дозволить краще йому орієнтуватися хто як навчається, а підтримка зв'язку з батьками студентів допоможе краще реалізовувати свої функції.

Мета і задачі розробки

Мета роботи – проаналізувати існуючі веб-сайти навчальних закладів, виділити їх основні переваги та недоліки, а також створити, на основі проведеного аналізу, Android додаток, який б включав всі плюси розглянутих сайтів, та виключав їх мінуси.

Методи, засоби та технології розробки

Для реалізації додатку була обрана клієнт-серверна архітектура, яка наддасть змогу одночасного доступу до системи багатьох користувачів. Технологією розробки для цього було обрано Android Studio, на мові java і базу даних MSSql Server.

Практичне значення одержаних результатів

Результатом виконання дипломної роботи є розроблений додаток, який забезпечує куратору доступ до рейтингової системи кожного із студентів його академічної групи та за необхідності можливість повідомляти студента та його батьків про його «успіхи» у навчанні, що в свою чергу дозволить контролювати успішність всієї групи.

РОЗДІЛ 1
АНАЛІЗ ПРИЕДМЕТНОЇ ОБЛАСТІ

1.1 Коротка характеристика об'єкту управління

Куратором академічної групи обирається провідний фахівець, досвідчений педагог з урахуванням напрямку підготовки та профілю діяльності факультету, стажу викладацької роботи у вищому навчальному закладі. Обов'язковою умовою його перебування на цій посаді є викладання лекцій або проведення семінарсько-практичних занять в даній академічній групі.

Діяльність куратора здійснюється на підставі Статуту закладу, Концепції виховної роботи з огляду на особливості й традиції закладу. Зміст його діяльності визначається Законом України «Про вищу освіту», Державною національною програмою «Освіта» («Україна XXI століття»), «Концепцією виховання дітей та молоді у національній системі освіти», «Національною доктриною розвитку освіти України у XXI столітті», відповідними інструктивно-методичними документами Міністерства освіти і науки України, а також положеннями, розробленими структурними ланками[1].

У роботі куратора передбачається виховна діяльність академічної групи, створення умов для самовираження кожного студента і розвитку кожної особистості. Доступ куратора до додатку який відображає успішність кожного студента його групи, дозволить йому стимулювати їх до навчання, та контролювати їхню відвідуваність, а також при необхідності взаємодіяти з батьками студента.

Тому для підвищення ефективності роботи куратора, потрібно розробити таке ПЗ, як дозволить куратору бути в курсі успішності та відвідування студентів його академічної групи, та в разі потреби корегувати

його за допомогою його прямого спілкування з студентом окремо, та в разі необхідності і з його батьками.

На сьогоднішній день більше 80% викладачів та студентів користуються смартфонами, і приблизно 95% з них на базі операційної системи Android, тому було обрано саме цю платформу для розробки програмної системи. Також додаток має ряд переваг над веб-сайтами чи веб-додатками на ПК, а саме:

- Завжди доступний(смартфон, на відміну від ПК, ви завжди носите з собою);
- Швидкий доступ до системи (не потрібно переходити на сайт системи в браузері, достатньо просто вибрати його на робочому столі вашого смартфону, який завжди з вами);
- Зручний інтерфейс;
- Оптимізоване використання інтернет трафіку, оскільки додаток завантажує тільки данні які потрібні йому для обробки (нічого лишнього, ніяких скриптів, плагінів і тд.)

1.2 Опис предметної області

Оскільки кожен куратор має одну окрему групу, то у системі будуть відбуваються такі бізнес-процеси:

- реєстрація
  - верифікація заповнених полів;
  - додання в БД;
  - відображення звіту про успішність реєстрації;
- авторизація
  - ідентифікація користувача;
  - формування списку студентів групи;
- робота в системі,
  - ідентифікація студента;

- відображення успішності студента;

- відправка повідомлення,
  - вибір критерію з переліку;
  - вибір тексту повідомлення;
  - вибір адресатів;
  - відправка повідомлення.

Для того, щоб розпочати роботу з додатком, користувач (куратор) повинен авторизуватися. Після успішної авторизації, система ідентифікує користувача та виведе на екран список тих студентів, які навчаються в його академічній групі. Далі куратору потрібно вибрати якого-небудь студента із списку і система покаже йому повну інформацію про успішність студента (кількість пропущених занять, та оцінка за кожен з навчальних предметів). Після цього куратор може вибрати будь-який з предметів, щоб переглянути детальну інформацію про нього, достатньо просто вибрати його із списку. Крім детальної інформації йому буде запропонована форма через яку можна відправити повідомлення батькам студента чи самому студенту про його успіхи в даному напрямку. Достатньо просто відзначити адресатів напроти їхніх імен, та натиснути кнопку «Send». Після цього система виведе на екран звіт про успішність надсилання повідомлення.

На рисунку 1.1 зображено діаграму бізнес-процесів системи.

Рисунок 1.1 – Діаграма ієрархії бізнес-процесів предметної області

На рисунку 1.2 зображено діаграму структури функції бізнес-процесу «Авторизація».

Рисунок 1.2 - Діаграма дерева функцій бізнес-процесу «Авторизація»

Характеристика бізнес-процесу «Авторизація» представлена у таблиці 1.1.

Таблиця 1.1

Характеристика бізнес-процесу «Авторизація»

Назва характеристики	Значення характеристики
Ім'я бізнес-процесу	Авторизація
Основні учасники	Куратор, Студент
Вхідна подія	Запуск додатку
Вхідні документ	Логін, пароль
Вихідна подія	Відображення списку групи
Вихідні документи	Список групи
Клієнт бізнес	Куратор

На рисунку 1.3 зображено діаграму структури функції бізнес-процесу «Робота в системі».

Рисунок 1.3 - Діаграма дерева функцій бізнес-процесу «Робота в системі»

Характеристика бізнес-процесу «Робота в системі» представлена у таблиці 1.2.

Таблиця 1.2

Характеристика бізнес-процесу «Робота в системі»

Назва характеристики	Значення характеристики
Ім'я бізнес-процесу	Робота в системі
Основні учасники	Куратор
Вхідна подія	Успішна авторизація
Вхідні документ	Список групи, id вибраного студента
Вихідна подія	Відображення успішності вибраного студента
Вихідні документи	Таблиця успішності студента
Клієнт бізнес	Куратор

На рисунку 1.4 зображено діаграму структури функції бізнес-процесу «Відправка повідомлення».

Рисунок 1.4 – Діаграма дерева функцій бізнес-процесу «Відправка повідомлення»

Характеристика бізнес-процесу «Відправка повідомлення» представлена у таблиці 1.3.

Таблиця 1.3

Характеристика бізнес-процесу «Відправка повідомлення»

Назва характеристики	Значення характеристики
Ім'я бізнес-процесу	Відправка повідомлення
Основні учасники	Куратор
Вхідна подія	Вибір конкретного студента
Вхідні документ	Таблиця успішності конкретного студента, перелік контактних даних батьків обраного студента
Вихідна подія	Відправка повідомлення
Вихідні документи	Текст повідомлення
Клієнт бізнес	Студент

На рисунку 1.5 зображено діаграму структури функції бізнес-процесу «Реєстрація».

Рисунок 1.5 - Структура функції бізнес-процесу «Реєстрація»

Характеристика бізнес-процесу «Реєстрація» представлена у таблиці 1.4.

Таблиця 1.4

Характеристика бізнес-процесу «Реєстрація»

Назва характеристики	Значення характеристики
Ім'я бізнес-процесу	Реєстрація
Основні учасники	Куратор, Студент
Вхідна подія	Запуск додатку
Вхідні документ	Логін, пароль, ім’я, прізвище, назва групи, тип користувача, емейли та імена батьків(при виборі типу студент)
Вихідна подія	Звіту про успішність реєстрації
Вихідні документи	Текст повідомлення
Клієнт бізнес	Куратор, Студент

1.3 Огляд та аналіз існуючих аналогів

Перш ніж почати розробляти певну систему, потрібно спочатку здійснити пошук аналогів, дослідити, проаналізувати їх переваги та недоліки, щоб зрозуміти, яких помилок потрібно уникнути та які їх переваги доцільно реалізувати.

У межах цієї дипломної роботи проаналізовано чотири програмні системи-аналоги, які реалізують схожі функції предметної області. Першим аналогом є: Всеукраїнська безкоштовна освітня мережа «Щоденник.ua» (режим доступу: http://shodennik.ua). Крім того, проаналізовано такі системи: для перегляду модульних оцінок з предметів що викладались в поточному семестрі ТНЕУ (режим доступу: http://mod.tanet.edu.te.ua/),систему призначена перегляду кількості пропущених занять, та розкладу ТНЕУ (режим доступу: http://teach.tneu.org/), систему MyAT (режим доступу: https://www.myattendancetracker.com/), де вчителі та викладачі можуть повідомляти про відвідуваність онлайн, відслідковувати студентів, надсилати повідомлення їх батькам і багато іншого в масштабі реального часу.

Тепер детально розглянемо всі ці системи-аналоги.

На рисунку 1.6 зображено головну сторінку Всеукраїнської освітньої мережі. На ній відображаються різні вкладки, можна переглянути щоденник, журнали вчителів; особисті сторінки учнів, вчителів та батьків; авторизації та реєстрації; також відображені новини освіти у світі; які надають послуги.

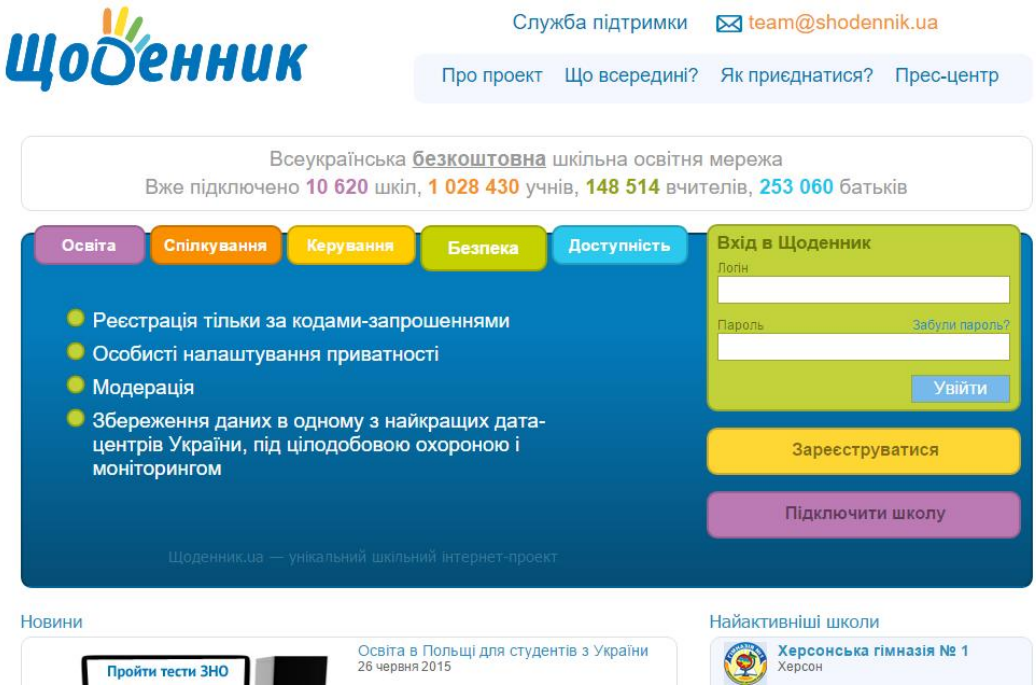
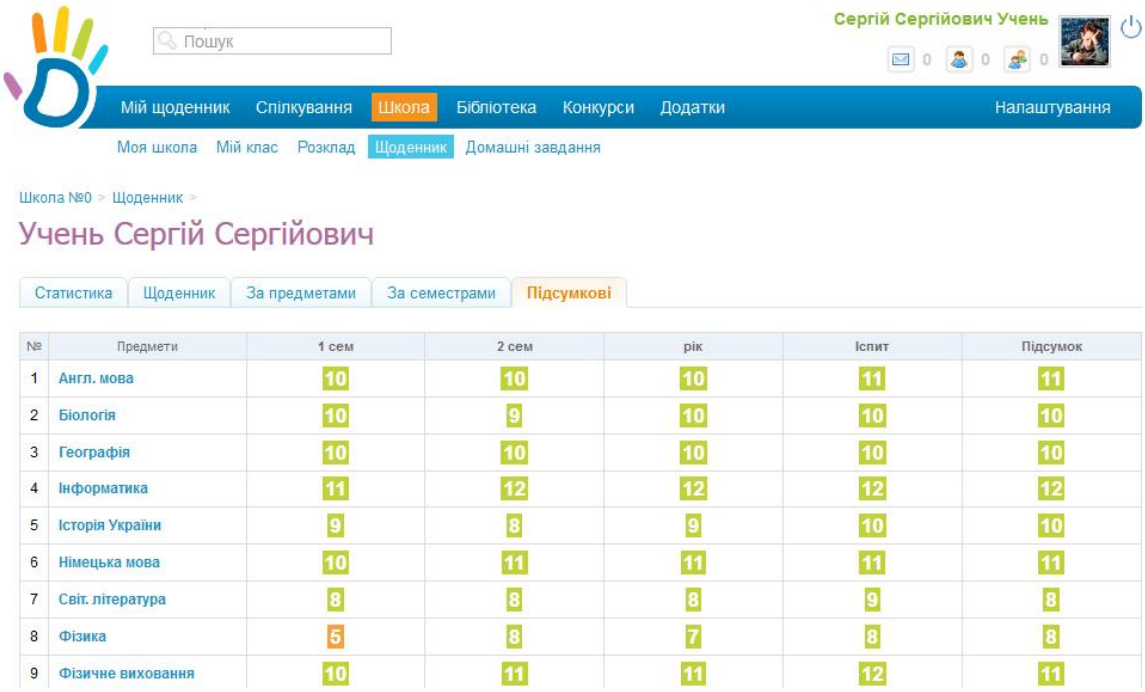


Рисунок 1.6 – Головне вікно сайту

Загальний вигляд вкладки «Щоденник» подано на рисунку 1.7. Кожному учневі в «Щоденнику» доступні всі його оцінки з різних предметів. Крім того, доступна можливість перегляду оцінок з предметів і за певний період (тиждень, семестр).



№	Предмети	1 сем	2 сем	рік	Іспит	Підсумок
1	Англ. мова	10	10	10	11	11
2	Біологія	10	9	10	10	10
3	Географія	10	10	10	10	10
4	Інформатика	11	12	12	12	12
5	Історія України	9	8	9	10	10
6	Німецька мова	10	11	11	11	11
7	Світ. література	8	8	8	9	8
8	Фізика	5	8	7	8	8
9	Фізичне виховання	10	11	11	12	11

Рисунок 1.7 – Вкладка «Щоденник»

Учителі в «Електронному журналі» можуть виставляти оцінки в тих класах, в яких вони викладають, а за наявності адміністративних прав, - і в будь-якому класі школи. На рис.1.8 показано загальний вигляд вкладки «Електронний журнал».

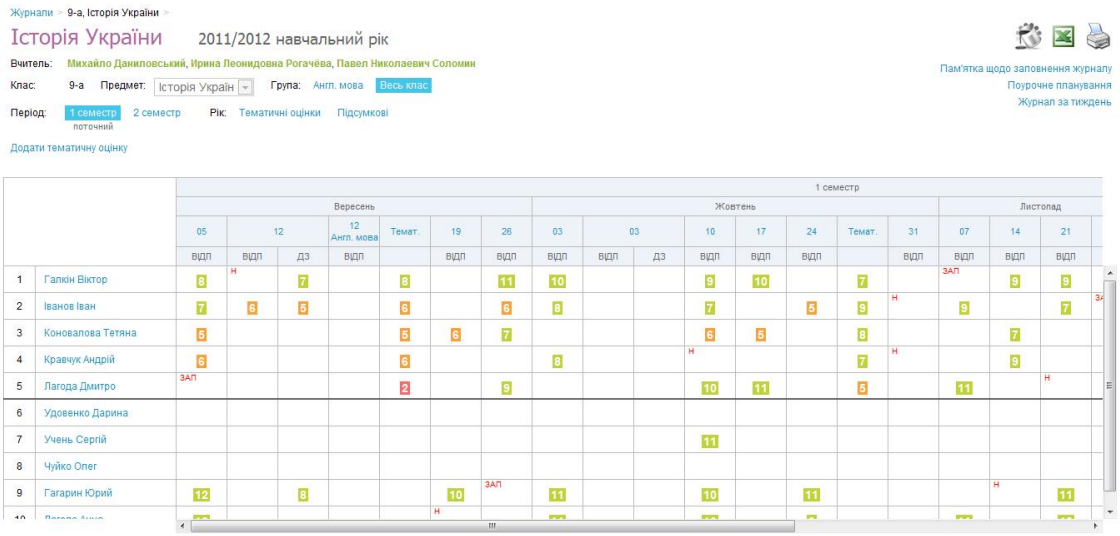


Рисунок 1.8 – Вкладка «Електронний журнал»

На сторінці конкретної школи міститься коротка інформація про школу, список керівництва та адміністраторів, останні новини, зміни в форумі, останні завантажені файли(див. рис. 1.9).

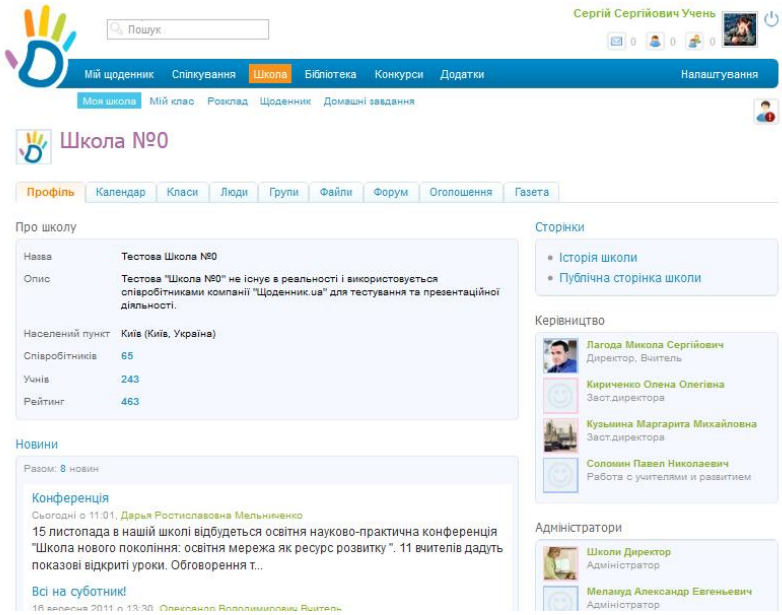


Рисунок 1.9 – Сторінка конкретної школи у системі «Щоденник»

Для кожного користувача «Щоденник» веде особистий календар, в якому відображується інформація про розклад уроків, дні народження та інші події(див. рис. 1.10).

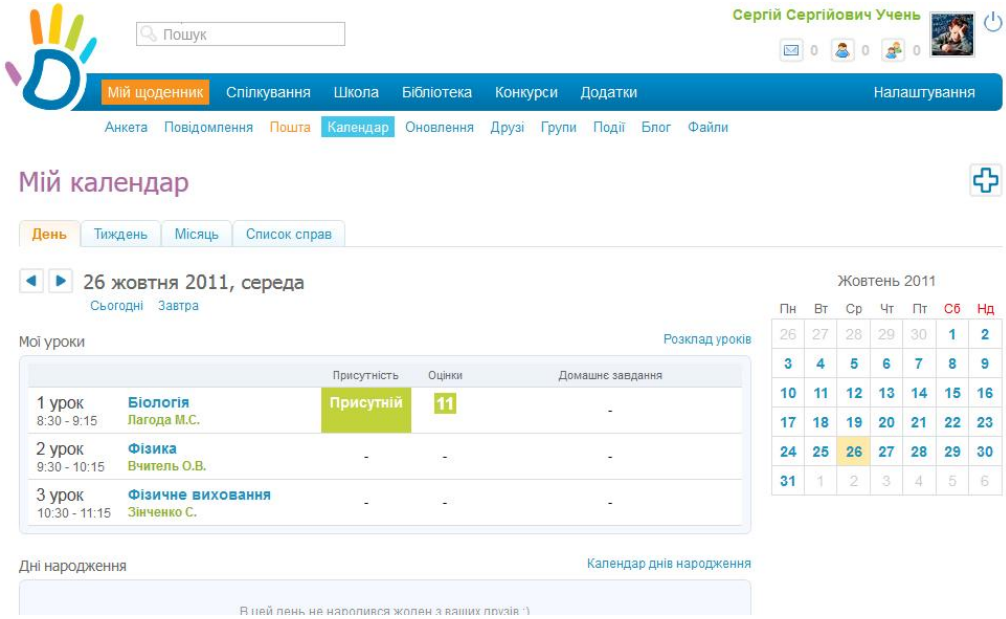


Рисунок 1.10 – Сторінка особистого календаря користувача

Крім того, кожен користувач системи «Щоденник» має свою особисту сторінку, де може розповісти про себе, про свої інтереси, завантажити фотографії, аудіо, відео, вести свій блог (див. рис. 1.11).

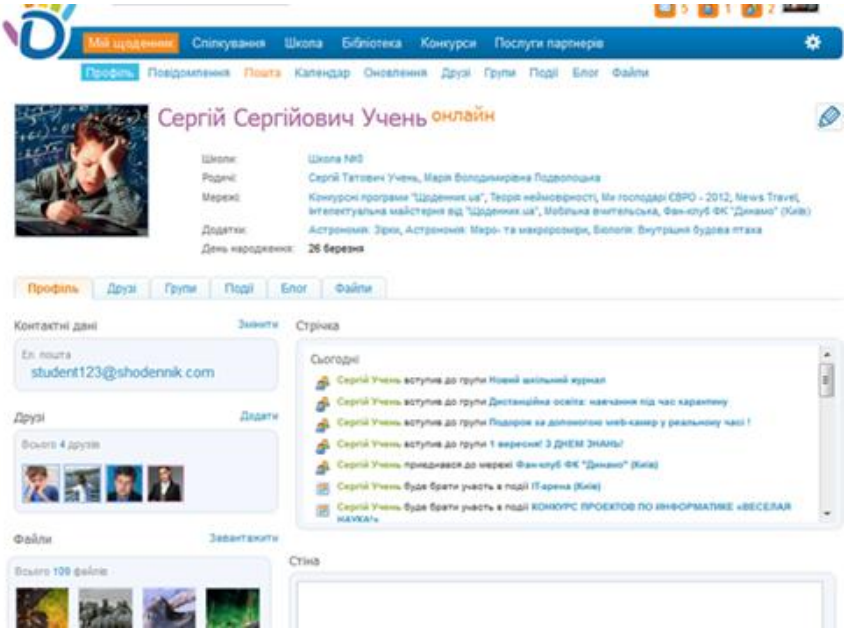


Рисунок 1.11 – Особиста сторінка користувача сайту

Далі розглянемо систему Rating system. На рисунку 1.12 зображено головне вікно сайту Rating system (ТНЕУ). На даному сайті є вкладки авторизації та реєстрації, також є вкладка з наведеним переліком кроків які необхідно зробити для реєстрації та контактна інформація на випадок втрати даних для авторизації.

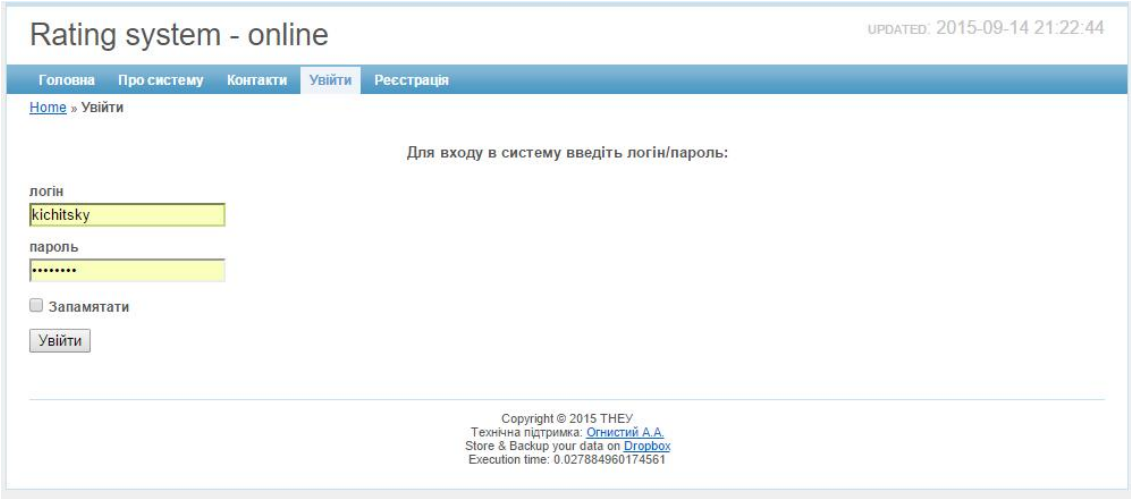


Рисунок 1.12 – Головне вікно сайту

Реєстрація в системі складається з двох частин. 1. На сторінці реєстрації потрібно заповнити всі поля(рисунок 1.13) 2. Зверніться у деканат з проханням активації свого облікового запису в системі. Вікно реєстрації показано на рисунку 1.13.

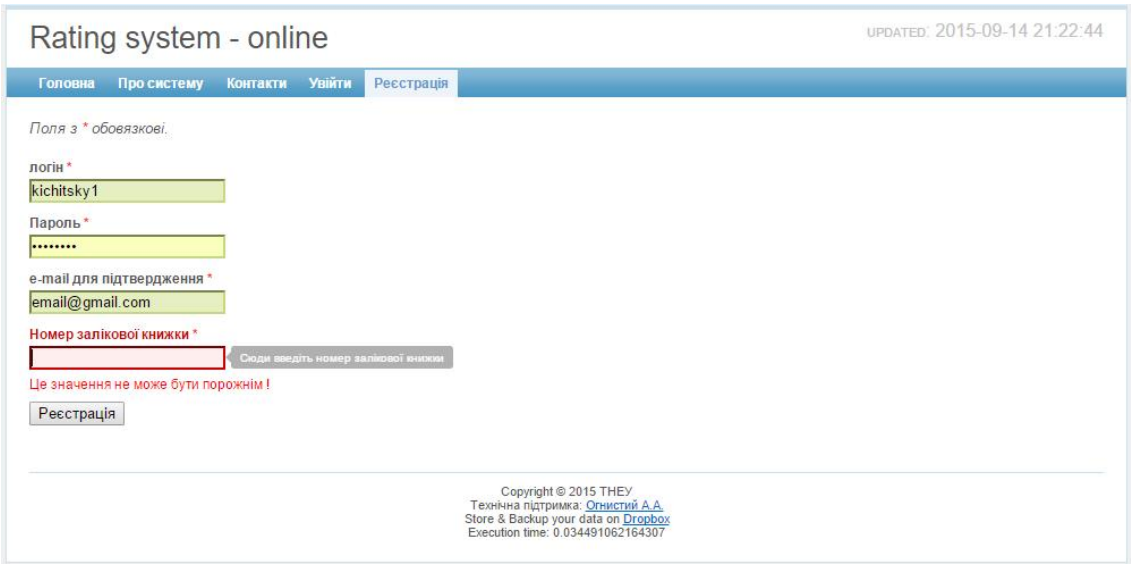
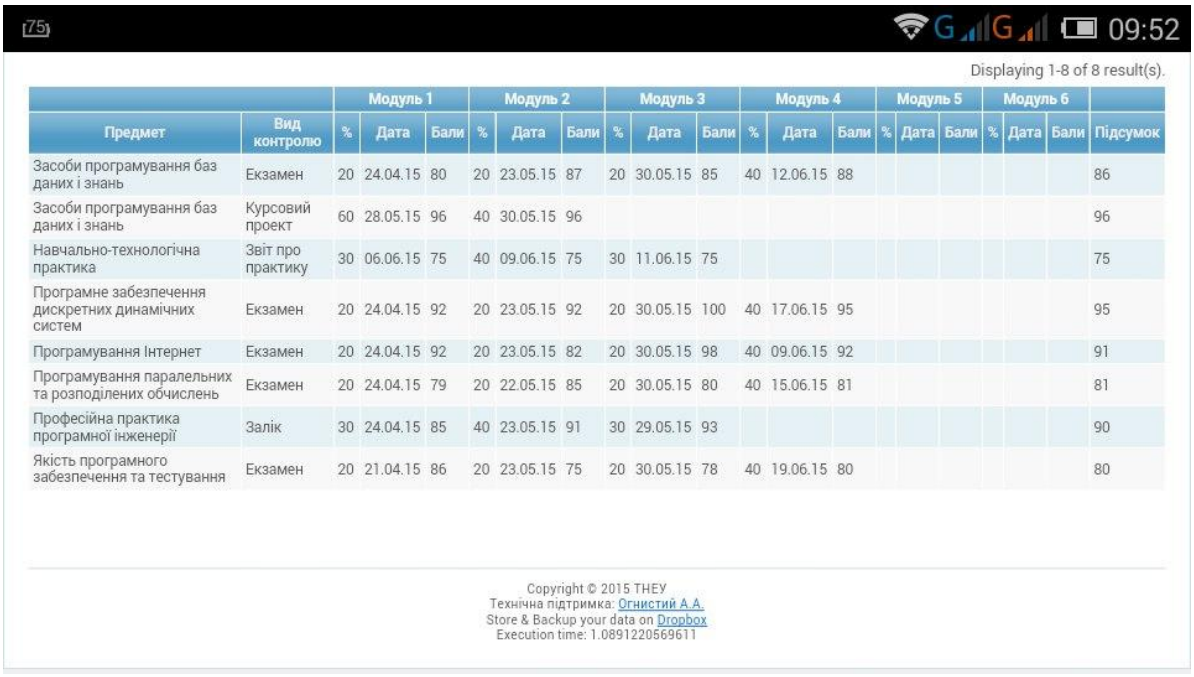


Рисунок 1.13 - Вікно реєстрації в системі

При успішній авторизації студент може переглянути свої модульні оцінки, та загальну оцінку за семестр. Вікно успішності показано на рисунку 1.14.



09:52

Displaying 1-8 of 8 result(s).

		Модуль 1			Модуль 2			Модуль 3			Модуль 4			Модуль 5			Модуль 6			
Предмет	Вид контролю	%	Дата	Бали	%	Дата	Бали	%	Дата	Бали	%	Дата	Бали	%	Дата	Бали	%	Дата	Бали	Підсумок
Засоби програмування баз даних і знань	Екзамен	20	24.04.15	80	20	23.05.15	87	20	30.05.15	85	40	12.06.15	88							86
Засоби програмування баз даних і знань	Курсовий проект	60	28.05.15	96	40	30.05.15	96													96
Навчально-технологічна практика	Звіт про практику	30	06.06.15	75	40	09.06.15	75	30	11.06.15	75										75
Програмне забезпечення дискретних динамічних систем	Екзамен	20	24.04.15	92	20	23.05.15	92	20	30.05.15	100	40	17.06.15	95							95
Програмування Інтернет	Екзамен	20	24.04.15	92	20	23.05.15	82	20	30.05.15	98	40	09.06.15	92							91
Програмування паралельних та розподілених обчислень	Екзамен	20	24.04.15	79	20	22.05.15	85	20	30.05.15	80	40	15.06.15	81							81
Професійна практика програмної інженерії	Залік	30	24.04.15	85	40	23.05.15	91	30	29.05.15	93										90
Якість програмного забезпечення та тестування	Екзамен	20	21.04.15	86	20	23.05.15	75	20	30.05.15	78	40	19.06.15	80							80

Copyright © 2015 ТНЕУ
Технічна підтримка: Огнистий А.А.
Store & Backup your data on Dropbox
Execution time: 1.0891220569611

Рисунок 1.14 – Відображення «Таблиця успішності»

Наступною розглянемо систему FCIT TEACH. На рисунку 1.15 зображено головне вікно сайту FCIT TEACH (ТНЕУ). На головній сторінці сайту є форма авторизації, вкладки з наведеним переліком критеріїв пошуку на сайті (розкладу студента, розкладу викладача, та пошуку вільної аудиторії).

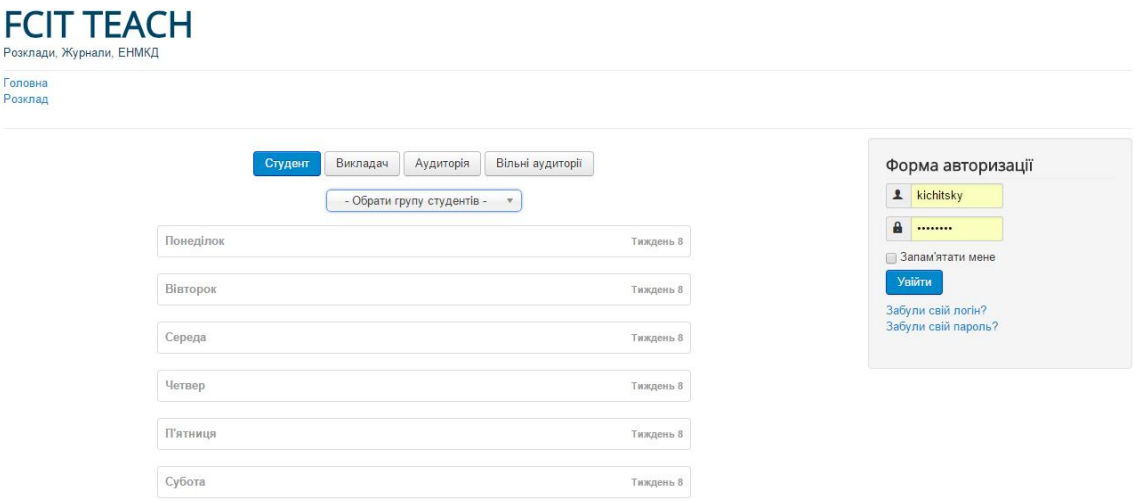


Рисунок 1.15 – Головне вікно сайту

При успішній авторизації студент може переглянути кількість пропущених занять (без ПП, не відпрацьованих),перелік довідок та заяв. Вікно пропущених занять показано на рисунку 1.16.

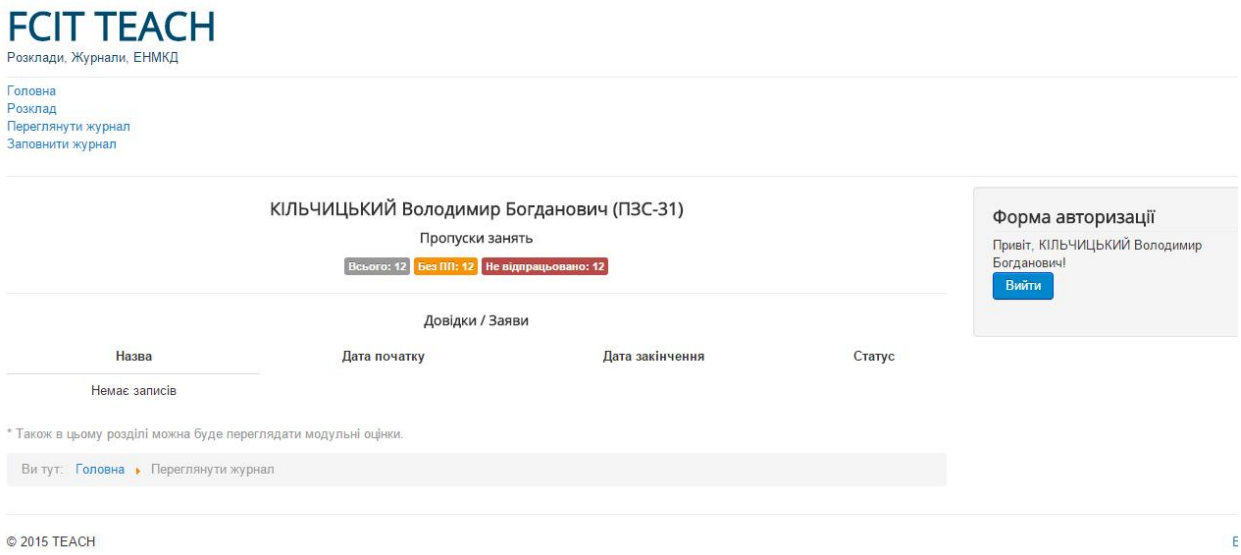


Рисунок 1.16 - Вікно пропущених занять

Також студент може переглянути розклад занять на поточний навчальний тиждень (як своєї групи так і всіх інших груп університету). Вікно розкладу занять показано на рисунку 1.17.

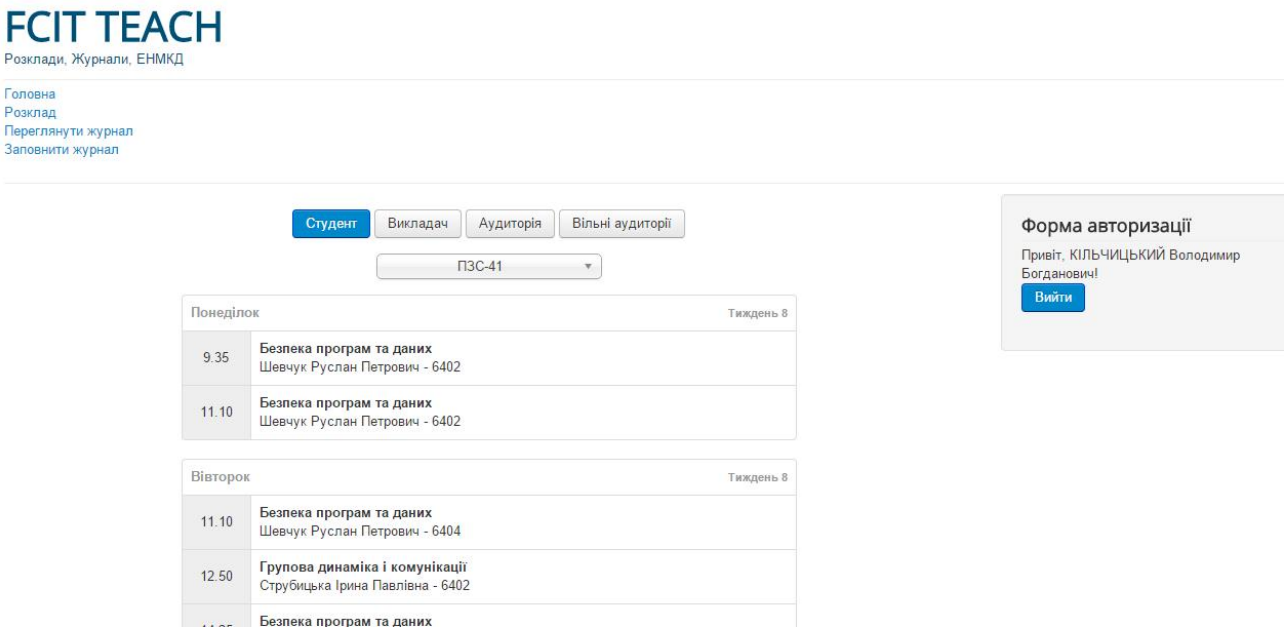


Рисунок 1.17 – Вікно розклад занять

Також ми можемо переглянути список вільних аудиторій на поточний навчальний тиждень, на потрібну годину, в потрібний. Вікно списку вільних аудиторій показано на рисунку 1.18.

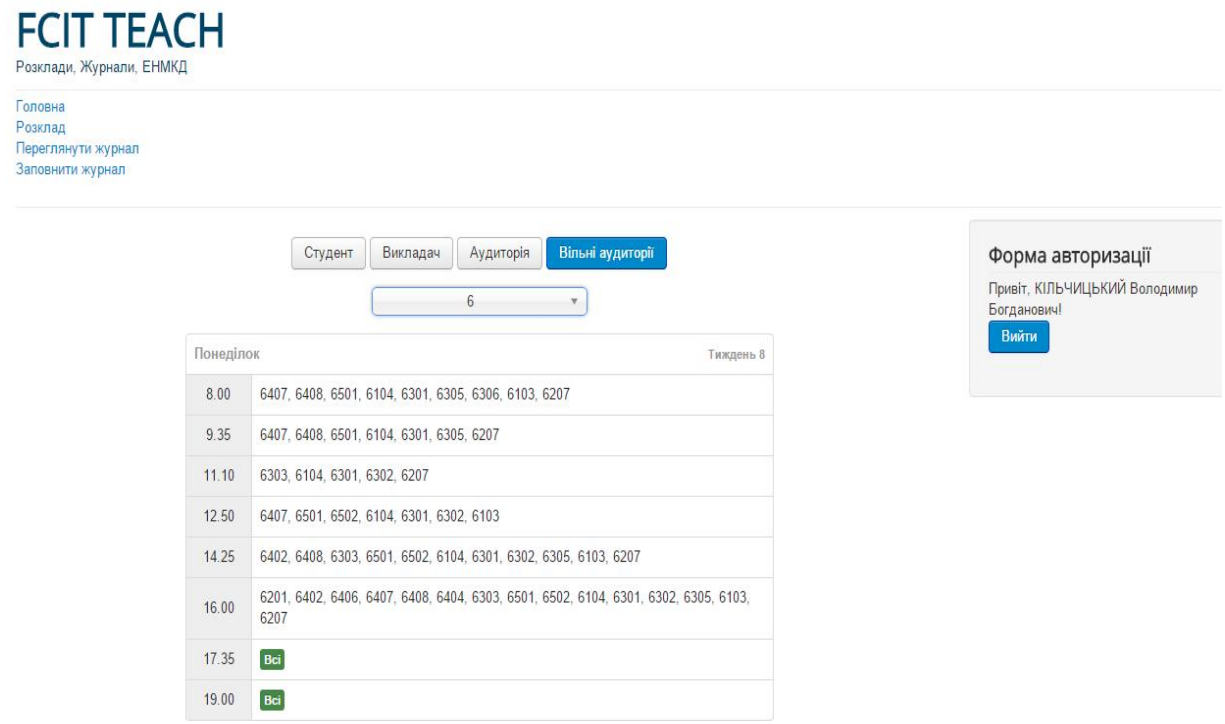


Рисунок 1.18 – Вікно пошуку вільних аудиторій

Наступною розглянемо систему MyAT. На рисунку 1.19 зображено головну сторінку сайту відстежування відвідуваності MyAT (MyAttendanceTracker). На даній сторінці відображаються різні вкладки, можна переглянути інформацію про систему, її авторів, особливості системи; авторизації та реєстрації; також відображені новини освіти у світі.

Рисунок 1.19 – Головне вікно сайту

Також ведеться облік зареєстрованих навчальних закладів, зареєстрованих користувачів, та студентів які відслідковуються системою, як це зображено на рисунку 1.20.

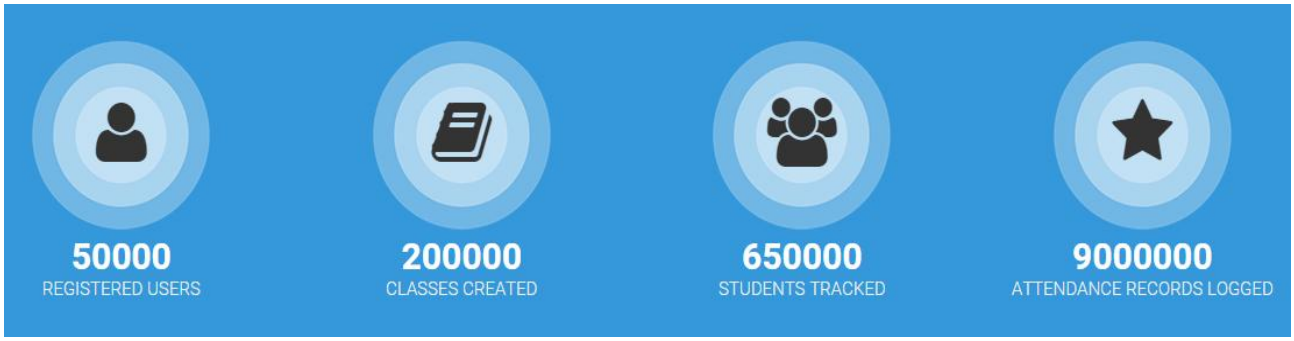


Рисунок 1.20 – Облік користувачів

Кожен користувач MyAT має свою особисту сторінку на котрій відображаються списки його студентів, предметів які він відвідує, його особиста відвідуваність, також тут студент може розповісти про себе, про свої інтереси, завантажити фотографії, як це зображено на рисунку 1.21.

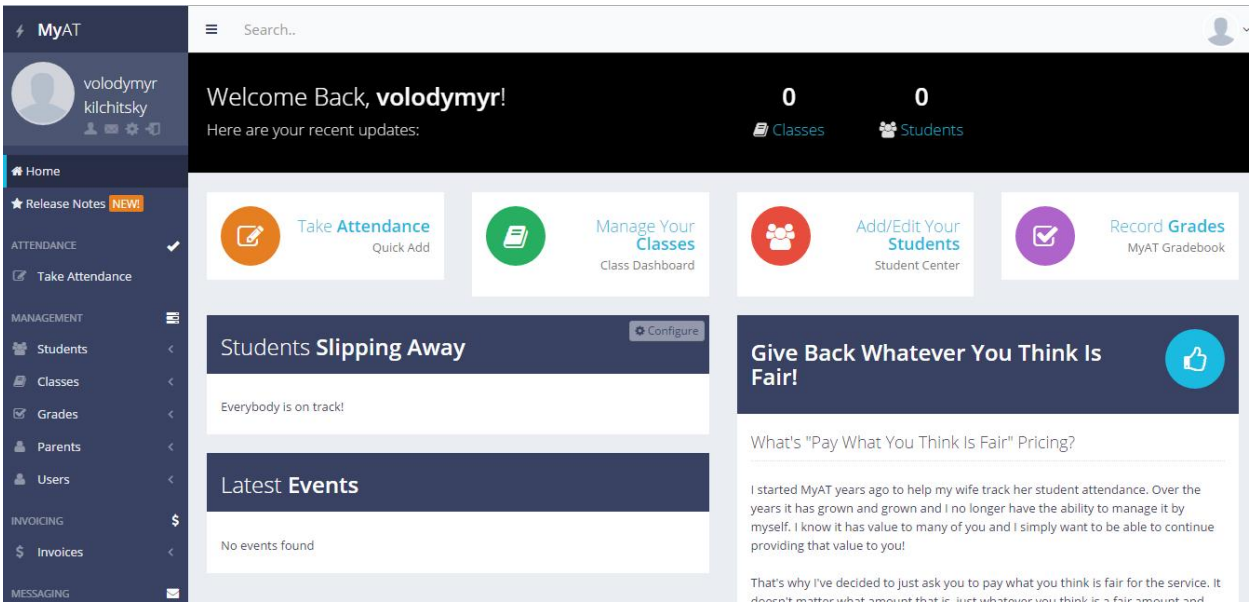


Рисунок 1.21 – Особиста сторінка користувача сайту

Вчителі в Електронному журналі можуть повідомляти про відвідуваність студентів в тих класах, в яких вони викладають, як це зображено на рисунку 1.22.

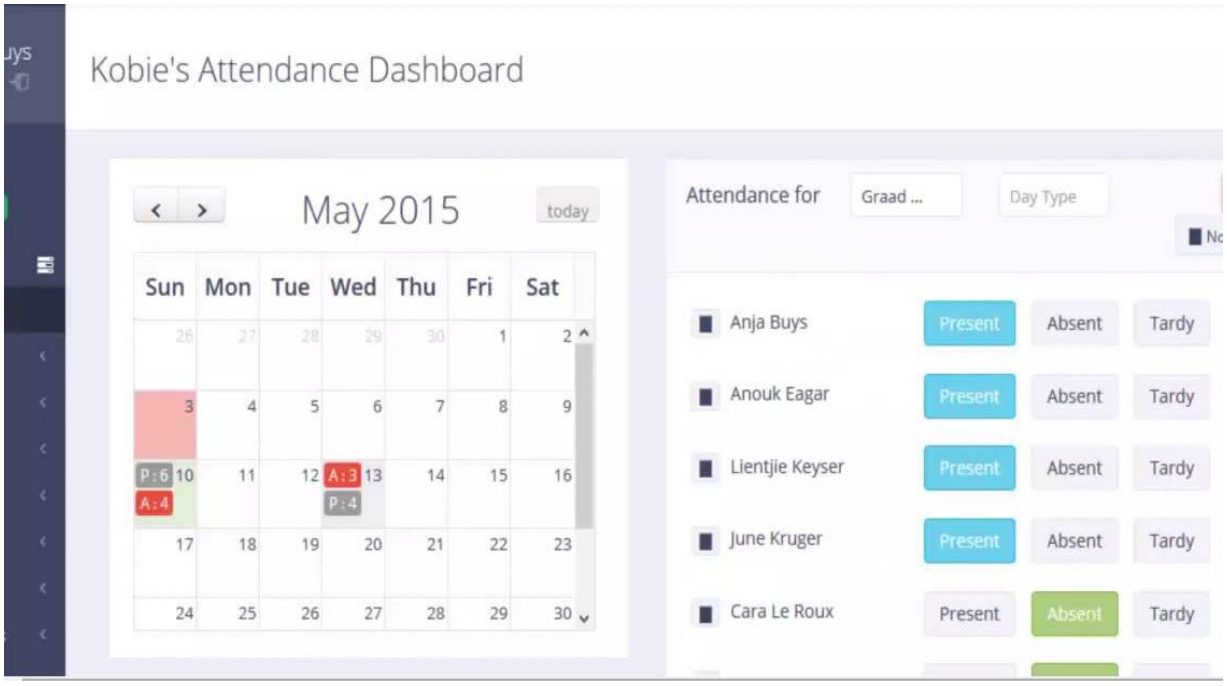


Рисунок 1.22 – Сторінка особистого календаря користувача

Таблиця 1.5

Порівняльна характеристика програмних продуктів

Фірма-розробник	Не вказано	Не вказано	Не вказано	Не вказано
Назва програмного продукту	Щоденник	Rating system	FCIT TEACH	MyAT
Версії продукту	1.0	1.0	1.0	1.0
Функціональність	- Профіль користувача/ школи - Перегляд користувачів - Перегляд шкіл - Щоденник - Журнал	- Профіль - Про систему - Допомога	- Профіль - Журнал - Вільні аудиторії - Розклад студента/ викладача	- Профіль користувача/ школи - Перегляд користувачів - Перегляд шкіл - Журнал відвідувності
Інтерфейс користувача	WEB-система	WEB-система	WEB-система	WEB-система
Допомога користувачеві	Відсутня	Вкладка «Про систему»	Відсутня	Вкладка «Про систему»
Читабельність	Присутня	Присутня	Присутня	Присутня

Детальний аналіз систем «Щоденник», «Rating system», «FCIT TEACH», «MyAT», дозволив прийняти рішення про те які функціональні та не функціональні вимоги повинна виконувати розроблювана у межах цієї дипломної роботи система «Android-додаток допомоги роботи куратора».

1.4 Специфікація вимог до додатку

Після огляду аналогів, детального аналізу їх функціонування та визначення основних переваг і недоліків кожного з них перейдемо до етапу опису специфікації вимог до програмного продукту – документу, котрий містить чіткий та повний опис розроблюваного продукту. Специфікація вимог є необхідним документом для зворотного зв’язку з замовником системи на початку проектування, тому повинна бути написана в простій та зрозумілій формі для сприйняття. До основних етапів формування специфікації можна віднести:

- створення глосарію розроблюваного продукту;
- опис варіантів використання системи, як зі сторони користувача, так і з керуючої сторони;
- опис основних функціональних та не функціональних вимог.

Отже, першочергово при розробці специфікації вимог створюють словник вузьконаправлених термінів, які будуть використані в процесі проектування, реалізації та експлуатації програмного продукту. Глосарій системи наведено в таблиці 1.6.

Таблиця 1.6

Глосарій

Термін	Опис терміну
Веб-сайт	Об’єднання веб-сторінок, доступних у мережі Інтернет, які об'єднані як за змістом, так і за навігацією. Фізично сайт може розміщуватися як на одному, так і на кількох серверах.
Авторизований користувач	Має можливість використовувати усі функції, які надаються сайтом.
Android - додаток	Веб-додаток – на платформі Android.
Адресат	той, кому адресується лист.

Продовження таблиці 1.6

Авторизація	Це надане будь-якій особі право на вчинення певних дій в конкретній системі - на сайті або терміналі, наприклад, при використанні онлайн-банкінгу. Авторизація дозволяє виключити доступ до інформації небажаних лиць та надання певних повноважень на виконання деяких дій у системі.
Реєстрація	Процес при якому користувач вказує ПІБ (логін), яким будуть підписуватись його пропозиції, також підтверджується поштова адреса та встановлюється пароль на акаунт.
Користувач	Це та особа, яка без авторизації має можливість переглядати сайт, але немає доступу до всіх його функцій.
Електронна пошта	це технологія та сервіс для надсилання й отримання електронних повідомлень (під назвою "листи" або "Електронна пошта") по комп'ютерній мережі (в тому числі глобальній).
Веб-додаток	клієнт-серверний додаток, в якому клієнтом виступає браузер, а сервером - веб-сервер. Логіка веб-додатку розподілена між сервером і клієнтом, зберігання даних здійснюється, переважно, на сервері, обмін інформацією відбувається по мережі.
Android	операційна система для комунікаторів, планшетних комп'ютерів, цифрових програвачів, наручних годинників, нетбуків і смартфонів, заснована на ядрі Linux.

Діаграма варіантів використання зображена на рисунку 1.23.

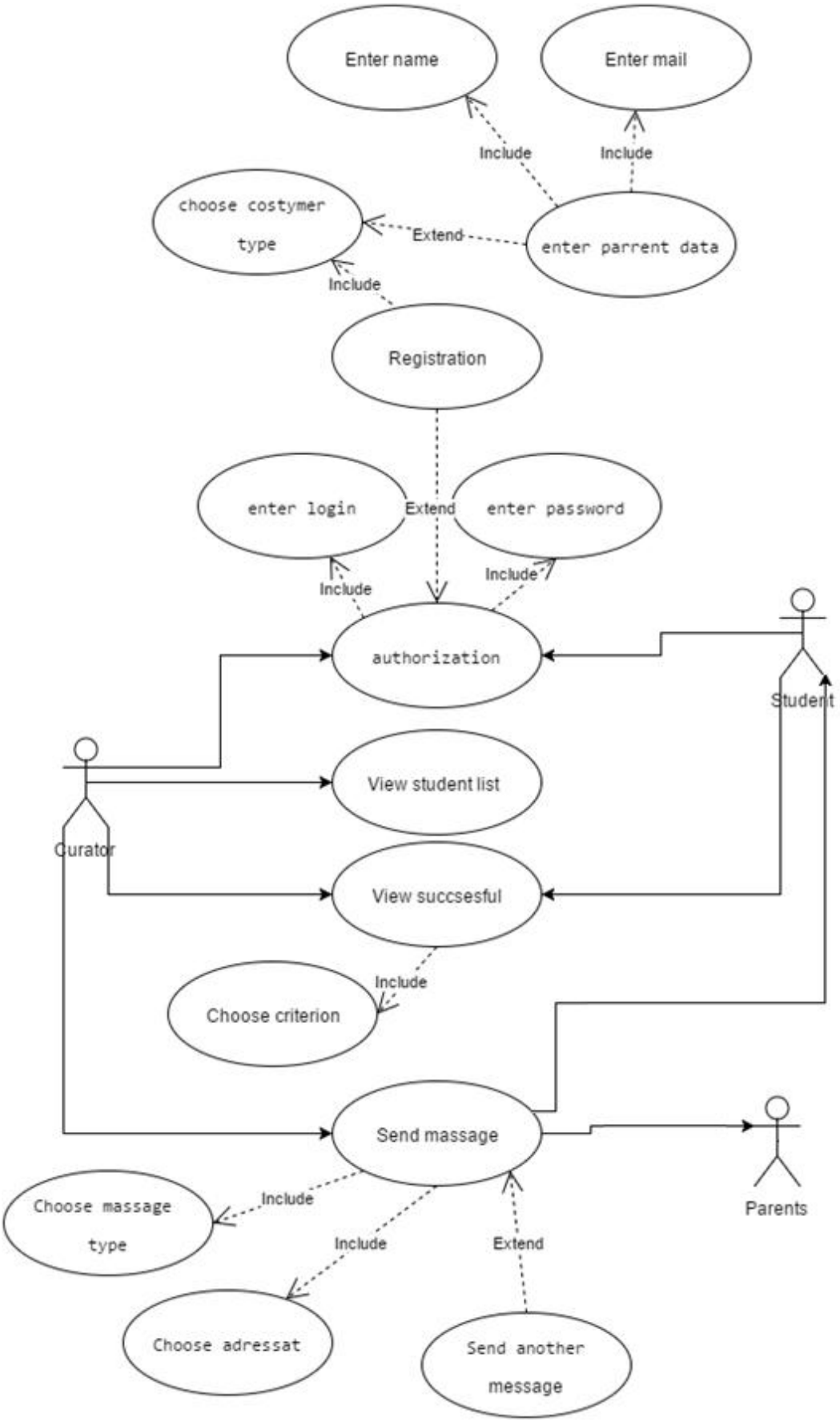


Рисунок 1.23 – Діаграма варіантів використання

Варіанти використання представленні у таблицях 1.7 - 1.12

Таблиця 1.7

Варіант використання реєстрація

Контекст використання	Реєстрація
Дійові особи	Користувач(куратор/студент)
Передумова	Доступ до мережі інтернет, користувач перейшов на сторінку «реєстрація»
Тригер	Відкрита сторінка реєстрації
Сценарій	1. Заповнюємо поле “Login” 2. Заповнюємо поле “Password” 3. Заповнюємо поле “Firs name” 4. Заповнюємо поле “Last name” 5. Заповнюємо поле “Group” 6. Заповнюємо поле “mail” 7. Вибрати тип реєстрованого користувача(куратор/студент) 8. Заповнюємо поле “Mather name” 9. Заповнюємо поле “Father name” 10. Заповнюємо поле “Mather mail” 11. Заповнюємо поле “Father mail” 12. Натиснути клавішу «Реєстрація»
Пост-умова	Користувач зареєстрований

Для того, щоб розпочати роботу з додатком, користувач(куратор) повинен авторизуватися, варіант використання процесу «Авторизація» представлений у таблиці 1.8.

Таблиця 1.8

Варіант використання «Авторизація»

Контекст використання	Авторизація
Дійові особи	Користувач (куратор)
Передумова	Повинен бути зареєстрований
Тригер	Відкрита сторінка авторизації
Сценарій	1. Заповнюємо поле “Login” 2. Заповнюємо поле “Password”
Пост-умова	Користувач авторизований

Розкадровка варіанту використання «Авторизація» зображена на рисунку 1.23.

Рисунок 1.24 - Розкадровка авторизації в системі

Після успішної авторизації, система ідентифікує користувача, та виведе на екран список тих студентів, які навчаються в академічній групі під його

кураторством. Далі куратору потрібно вибрати якого-небудь студента із списку і система покаже йому повну інформацію про успішність студента(кількість пропусків, та оцінка за кожен з навчальних предметів), варіант використання процесу «Перегляд успішності» представлений у таблиці 1.8.

Таблиця 1.9

Варіант використання «Перегляд успішності»

Контекст використання	Перегляд успішності
Дійові особи	Користувач(куратор/студент)
Передумова	Користувач авторизований
Тригер	Відкритий профіль користувача
Сценарій	При авторизації студента таблиця успішності відображається автоматично, для куратора: потрібно вибрати потрібного нам студента із представленого йому списку студентів групи.
Пост-умова	Показано таблицю успішності(власну для студента, вибраного студента – для куратора)

Розкадровка варіанту використання «Перегляд успішності» зображена на рисунку 1.25.

Рисунок 1.25 - Розкадровка сторінки перегляду успішності

Після цього куратор може вибрати будь-який з предметів, щоб переглянути детальну інформацію про нього, достатньо просто вибрати його із списку. Крім детальної інформації йому буде запропонована форма через яку можна відправити повідомлення батькам студента чи самому студенту про його успіхи в даному напрямку. Достатньо просто відзначити адресатів напроти їхніх імен, та натиснути кнопку «Send», варіант використання процесу «Відправка повідомлення» представлений у таблиці 1.10.

Таблиця 1.10

Варіант використання «Відправка повідомлення»

Контекст використання	Відправка повідомлення
Дійові особи	Користувач (Куратор)
Передумова	Користувач авторизований як куратор, а також вибрано студента для відправки повідомлення
Тригер	Відкрито додаток
Сценарій	1. Натискаємо на предмет з негативними даними 2. Вибираємо один з запропонованих текстів повідомлення, або ж набираємо його самі. 3. Відзначаємо одного або декількох адресатів(студент/мати/батько) 4. Натискаємо клавішу «Send»
Пост-умова	Повідомлення були надіслані всім адресатам.

Розкадровка варіанту використання «Перегляд успішності» зображена на рисунку 1.26.

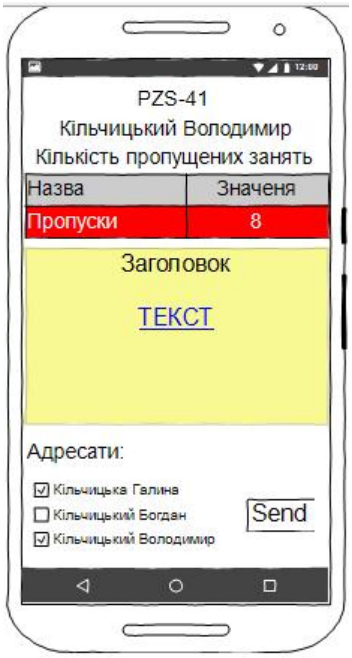


Рисунок 1.26 - Розкадровка сторінки відправки повідомлення

Після цього система виведе на екран звіт про успішність надсилання повідомлень (див. рис. 1.27).

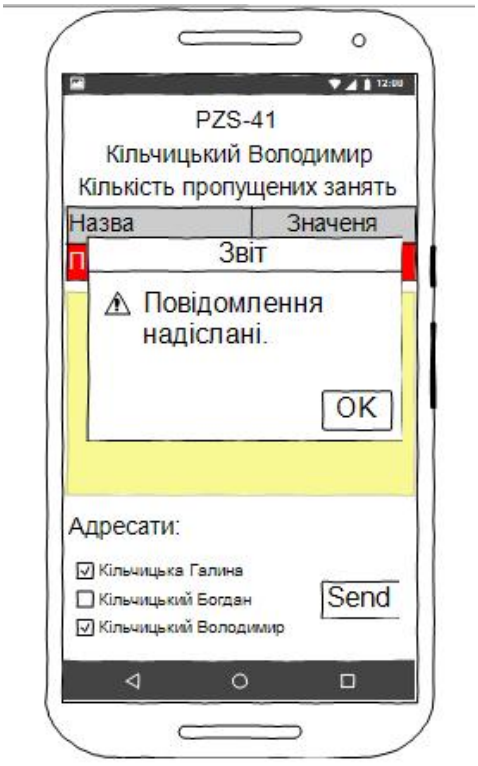


Рисунок 1.27 - Звіт про успішність надсилання повідомлень.

Специфікація функціональних вимог наведена у таблиці 1.11

Таблиця 1.11

Специфікація функціональних вимог

Ідент. вимог	Назва вимоги	Атрибути вимог		
		Пріоритет	Складність	Контакт
1.	Перегляд функції «Реєстрація»	Високий	Середня	Користувач
2.	Перегляд функції «Авторизація»	Високий	Середня	Користувач
3.	Перегляд функції «Перегляд успішності»	Високий	Середня	Користувач
4.	Перегляд функції «Відправка повідомлення»	Високий	Середня	Користувач
5.	Перегляд функції «Перегляд списку студентів»	Високий	Середня	Користувач

Специфікація нефункціональних вимог наведена у таблиці 1.12

Таблиця 1.12

Специфікація нефункціональних вимог

№	Назва вимоги	Характеристики
1.	Застосовність	
1.1	Час для навчання користувачів	Не більше 10 хв.
1.2	Вимірюваний час відгуку для типових завдань	Не повинно відрізнятися від аналогів
2.	Надійність	
2.1	Середній час безвідмовної роботи	10 год.
2.2	Середнє напрацювання до ремонту	3 місяці
3	Проектні обмеження	
3.1	Мова програмування	java

Висновки до розділу 1:

1. Наведено коротку характеристику об’єкту дослідження та визначено склад функцій, що входять до основних бізнес-процесів предметної області та повинні бути автоматизованими.

2. Проведено огляд та аналіз кількох веб-систем аналогів, що реалізують схожі функції предметної області.

3. Проведено етап аналізу вимог до розроблюваного додатку. У процесі аналізу вимог побудовано діаграми варіантів використання, які візуалізують функціональні вимоги до системи та проведено детальний опис функцій розроблюваної програмної системи, також описано не функціональні вимоги.

РОЗДІЛ 2
ПРОЕКТУВАННЯ ДОДАТКУ

2.1 Розробка архітектури додатку

Додаток буде розроблено на основі архітектури «клієнт-сервер». Котра є одним із архітектурних шаблонів програмного забезпечення та має найпоширенішою концепцією у створенні розподілених мережевих застосунків і передбачає взаємодію та обмін даними між ними. Вона передбачає такі основні компоненти[3]:

- набір серверів, які надають інформацію або інші послуги програмам, які звертаються до них;
- набір клієнтів, які використовують сервіси, що надаються серверами;
- мережа, яка забезпечує взаємодію між клієнтами та серверами.

Як сервери так і клієнти функціонують паралельно і незалежно один від одного. Немає жорсткої прив'язки клієнтів до серверів. Більш ніж типовою є ситуація, коли один сервер одночасно обробляє запити від різних клієнтів; з іншого боку, клієнт може звертатися то до одного сервера, то до іншого. Клієнти мають знати про доступні сервери, але можуть не мати жодного уявлення про існування інших клієнтів[3].

На рисунку 2.1 наведено діаграму, яка візуалізує архітектуру розроблюваного андроїд-додатку «підтримки роботи куратора групи».

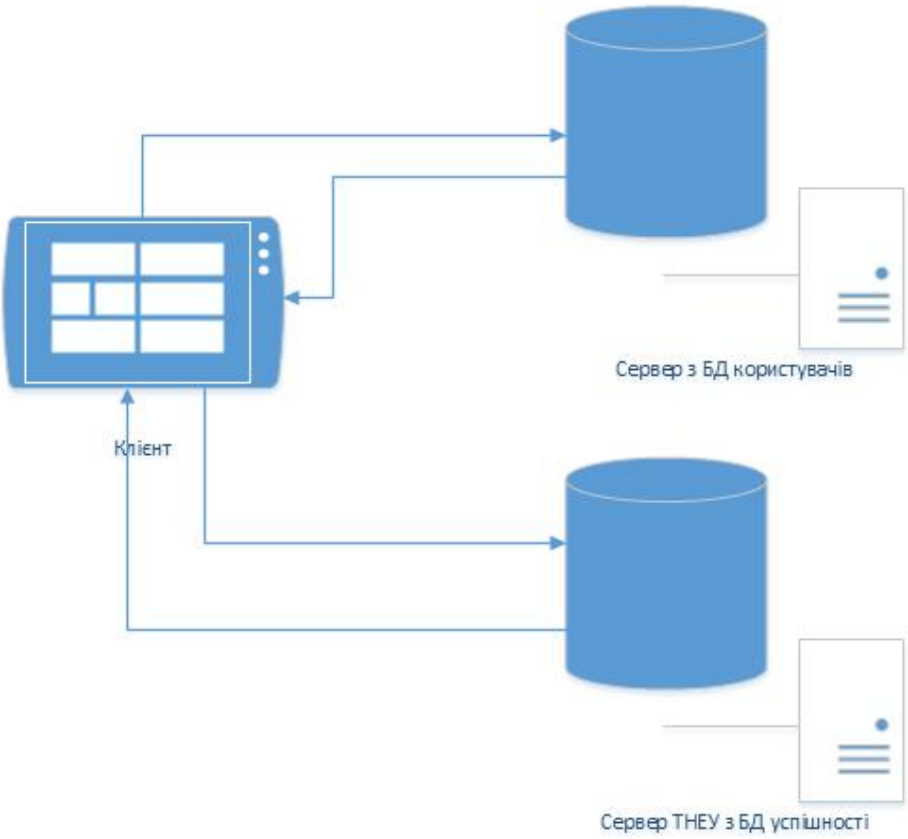


Рисунок 2.1 - Архітектура андроїд-додатку «підтримки роботи куратора групи».

Розроблювальна система використовує підключення до двох різних серверів, на першому знаходиться база даних зареєстрованих користувачів (студентів та кураторів), який використовується додатком при авторизації(повертає звіт чи зареєстрований даний користувач в системі, та якщо так, то який у нього тип доступу(студент чи куратор)) чи реєстрації користувача в системі. Інший ж, це сервер ТНЕУ з базою даних успішності всіх його студентів, на який посилається запит при неуспішній авторизації користувача додатком в БД користувачів(тому що є імовірність що користувач ще не зареєстрований в даному додатку), також при успішній ідентифікації його в БД користувачів як «студент», та при виборі куратором студента чию успішність він хотів би переглянути. Він повертає json з успішністю студента, який потім наша система розпарсить та виведе в зрозуміле нам вікно.

Розглянемо детальніше схему роботи розроблюваного додатку. Для того щоб куратор міг бачити список усіх своїх студентів, їм всім спочатку потрібно зареєструватися в системі вказавши назву групи у якій вони навчаються, інакше список буде не повний. Студенту ж непотрібно робити ніяких додаткових кроків щоб побачити свою успішність, достатньо просто відкрити додаток, ввести логін та пароль від рейтингової системи ТНЕУ та натиснути клавішу «Login».

Для того, щоб розпочати роботу з додатком, користувач (куратор або студент) повинен авторизуватися (на рисунку 2.2 наведено діаграму станів процесу «Авторизація»).

Рисунок 2.2 - Діаграма станів процесу «Авторизація»

Після успішної авторизації система, якщо користувач ідентифікований як куратор, виведе на екран список тих студентів, які навчаються в академічній групі під його кураторством, якщо ж він ідентифікований як студент, то виведе на екран таблицю його успішності (оцінки за кожен з модуль, кожного з навчальних предметів). Далі куратору потрібно вибрати студента із списку, і система покаже йому повну інформацію про успішність студента (таку ж як бачить студент при успішній авторизації). На рисунку 2.3 зображено діаграма станів процесу «Перегляд успішності».

Рисунок 2.3 - Діаграма станів процесу «Перегляд успішності»

Після цього куратор може вибрати будь-який з предметів (користувачеві ідентифікованому як студент ця функція буде недоступною), йому буде запропонована форма через, яку можна відправити повідомлення батькам студента чи самому студенту. Достатньо просто відзначити адресатів напроти їхніх імен, та натиснути кнопку «Send» (на рисунку 2.4 зображено діаграма станів процесу «Надсилання повідомлення»).

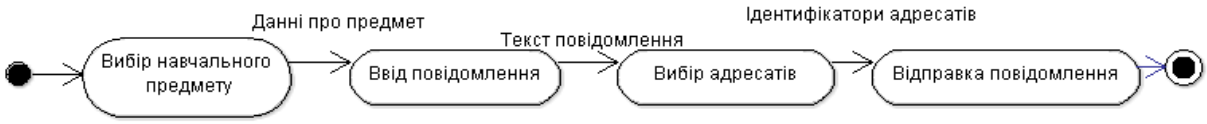


Рисунок 2.4 - Діаграма станів процесу «Надсилання повідомлення»

На рисунку 2.5 представлено діаграму класів.

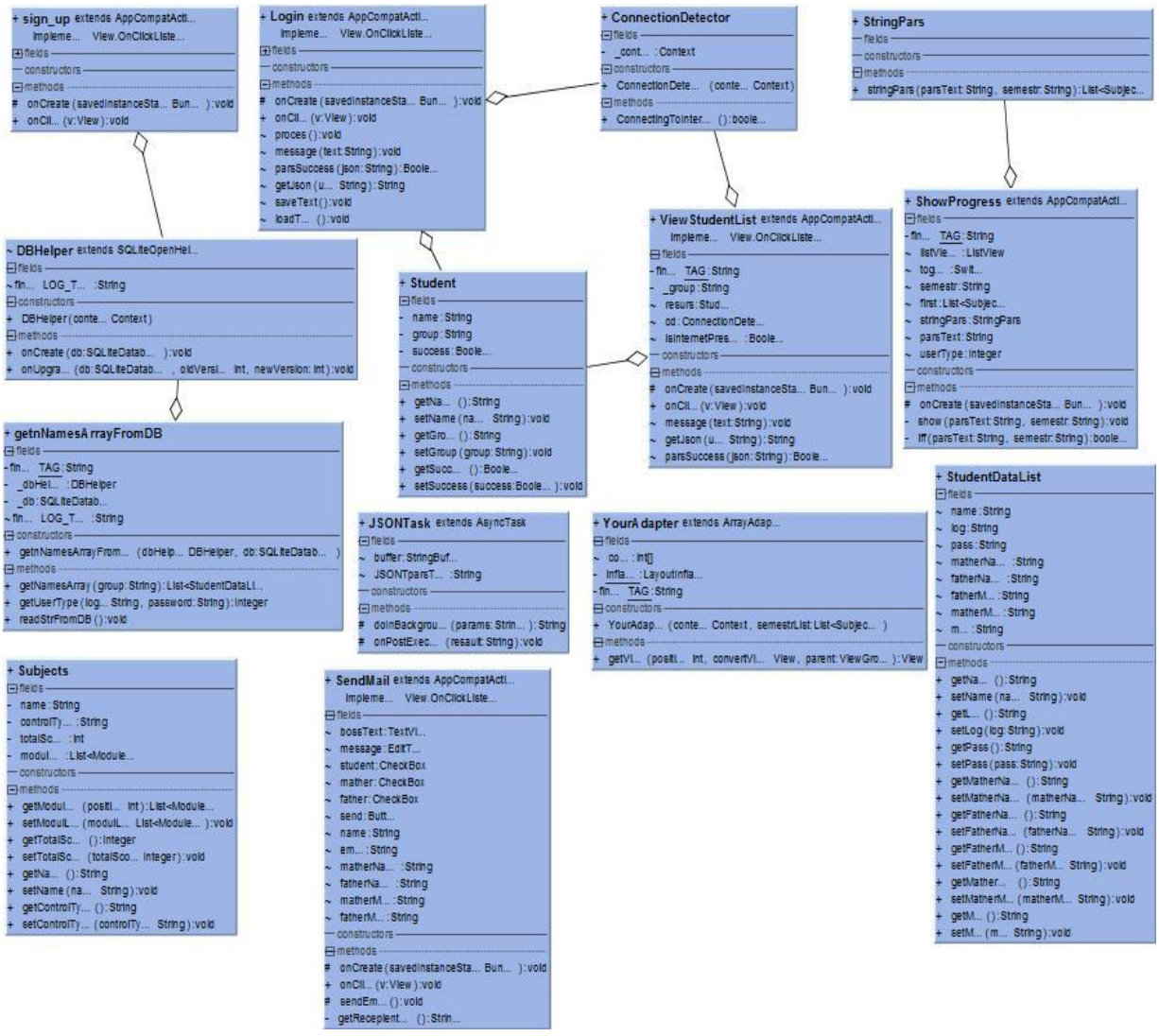


Рисунок 2.5 - Діаграма класів

Клас Modules призначений для зберігання даних про один модуль одного з предметів, має такі поля як: date, weight та score.

Клас Subjects призначений для зберігання даних про предмет, має такі поля як: name, controlType, totalScore та список типу Modules.

Клас Student призначений для зберігання даних про студента, має такі поля як: name, controlType, group та список типу success.

Клас Login котрий містить функції та поля необхідні для авторизації клієнта в системі.

Клас ConnectionDetector котрий містить функції та поля необхідні для перевірки наявності підключення до мережі інтернет.

Клас DBHelper котрий містить функції та поля необхідні для перевірки сумістності системи з БД.

Клас getnNamesArrayFromDB котрий містить функції та поля необхідні для отримання даних з БД.

Клас JSONTask котрий містить функції та поля необхідні для відправки запитів на сервер, та отримання від нього відповіді.

Клас SendMail котрий містить функції та поля необхідні для відправки емейл повідомлення.

Клас ShowProgress котрий містить функції та поля необхідні для відображення успішності студента у вікні додатку.

Клас sign_up котрий містить функції та поля необхідні для реєстрації нового користувача в БД користувачів.

Клас StringPars котрий містить функції та поля необхідні розпаршення json.

Клас ViewStudentList котрий містить функції та поля необхідні для відображення списку студентів певної групи.

Клас YourAdapter котрий містить функції та поля необхідні для заповнення вікна перегляду успішності конкретного користувача.

2.2 Проектування структури бази даних.

Під час розробки системи необхідним етапом є проектування бази даних, яка забезпечить зберігання даних і наддасть змогу проводити операції над ними. На цьому етапі потрібно визначити, які дані будуть зберігатися та які операції необхідно проводити над ними. Відповідно до відношення формувати структуру БД з визначенням основних сутностей. Після того, як основні сутності сформовані, необхідно встановити логічні зв’язки між ними, щоб не виникало помилок при зберіганні або обробці даних.

При авторизації система використовує підключення до двох різних серверів, у першу чергу шукає користувача у базі даних зареєстрованих користувачів (студентів та кураторів), сервер повертає звіт чи зареєстрований даний користувач в системі, якщо так, то який у нього тип доступу(студент чи куратор). При неуспішній авторизації користувача додатком в БД користувачів(тому що є імовірність що користувач ще не зареєстрований в даному додатку), а також при успішній ідентифікації його в БД користувачів як «студент», та при виборі куратором студента чию успішність він хотів би переглянути, додаток віпрравляє запит на сервер ТНЕУ з базою даних успішності всіх його студентів, котрий повертає json з результатом пошуку студента, та успішністю студента, в разі успішної його ідентифікації, який потім наша система розпарсить та відобразить у зручному для перегляду вікні.

Діаграми потоків даних (Data flow diagram, DFD) використовуються для опису документообігу та обробки інформації. Подібну до IDEF0, DFD представляє змодельовану систему як мережу пов'язаних між собою робіт. Їх можна використовувати як доповнення до моделі IDEF0 для більш наочного відображення поточних операцій документообігу в корпоративних системах обробки інформації. Головна мета DFD - показати, як кожна робота перетворює вхідні дані у вихідні, а також виявити відносини між цими

роботами. Для представлення потоків даних в нашій системі використана діаграма DFD, яка представлена на рисунку 2.7.

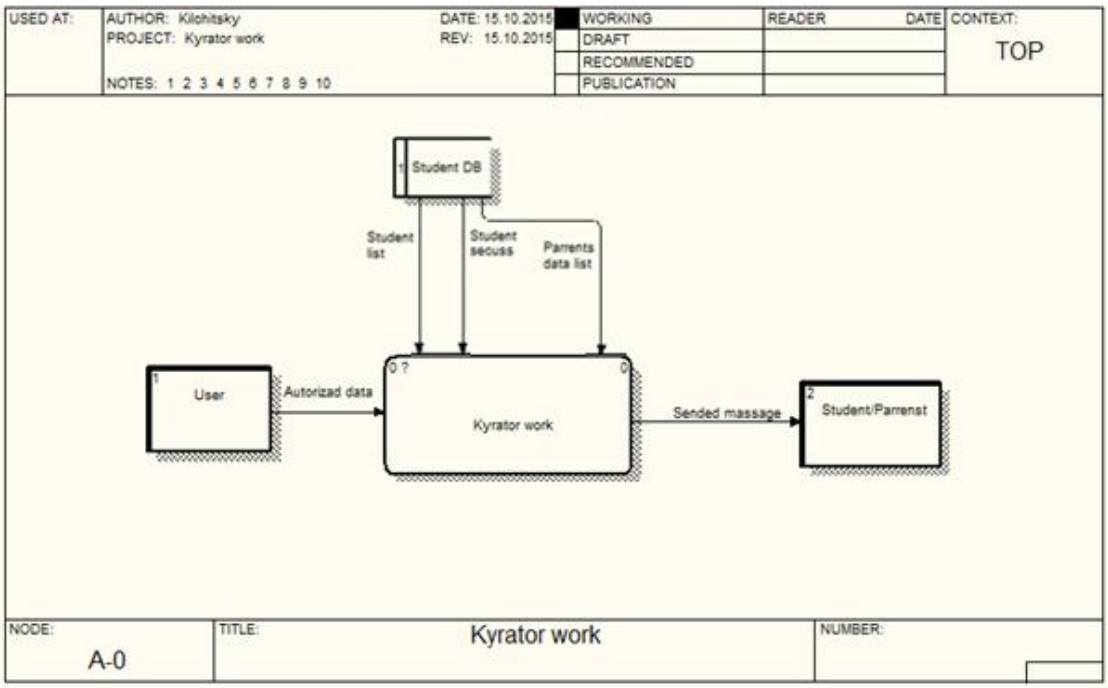


Рисунок 2.7 – Контекстна діаграма DFD розроблюваної системи

Для опису основних процесів та інформаційних потоків проведемо декомпозицію верхнього рівня діаграми потоків даних, яку зображена на рисунку 2.8.

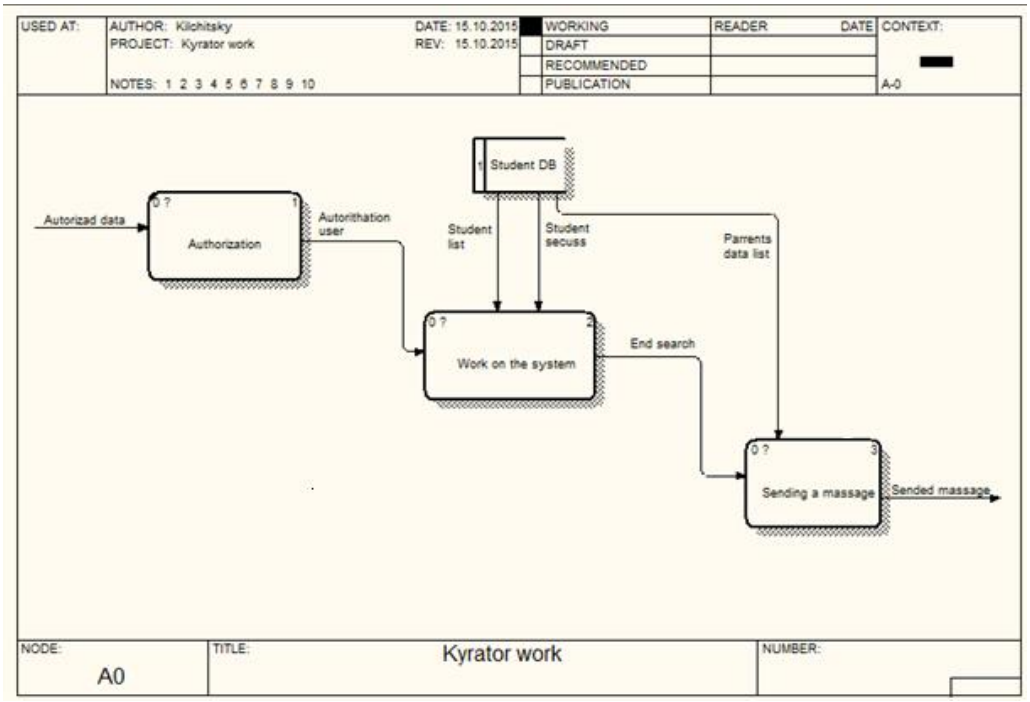


Рисунок 2.8 - Декомпозиція DFD діаграми

Для розробки сховища зберігання даних обрана реляційна модель бази даних. У реляційній моделі база даних являє собою централізоване сховище таблиць, що забезпечує безпечний одночасний доступ до інформації з боку багатьох користувачів. Наступним етапом є розробка діаграми корпоративної моделі даних, для відображення елементів, які будуть в нашій базі даних та в'язків між ними. Діаграма представлена на рисунку 2.9.

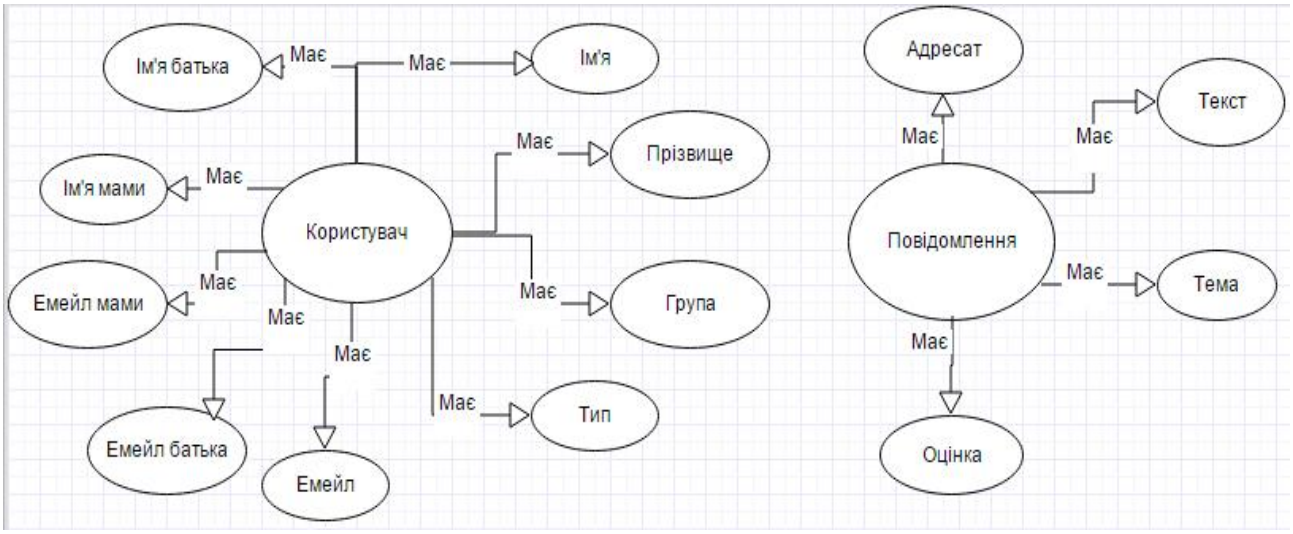


Рисунок 2.9 - Структура діаграми корпоративної моделі даних

Далі формується таблиця ідентифікаторів, яка представляє атрибути, їх тип і розмірність, які будуть в відношеннях.

Таблиця 2.1

Ідентифікатори

Об'єкт	Властивість	Тип	Розмірність	Ідентифікатор
Користувач	Ім'я	Text	50	FirstName
	Прізвище	Text	50	LastName
	Тип	Bool	1	Type
	Емелй	Text	50	Mail
	Група	Text	10	Group
	Ім'я батька	Text	50	Father Name
	Ім'я матері	Text	50	Mather Name

Продовження таблиці 2.1

Користувач	Емелй батька Емелй матері	Text Text	50 50	Father mail Mather name
Повідомлення	Текст Тема Адресат Оцінка	Text Text Text Int	500 50 50 3	Text Topic Addressee Rating

Після отриманих даних розробляється реляційна модель даних. Вона представлення у вигляді ERD, яка в свою чергу показує відношення та зв'язки між ними. ERD на програмному рівні проектується за допомогою коду, щоб дані, які нам потрібні, для роботи бути логічно взаємопов'язані і забезпечували зберігання даних в таблиці. ERD представлена на рисунку 2.10. Фізична реалізація входить у проектування бази даних. Фізичне проектування включає побудову ERD за допомогою коду. Для фізичного проектування обрано базу даних Sqlite, так як вона є безкоштовною і є більш оптимальним рішенням, оскільки буде використовуватись як сервер в розроблюваній системі.

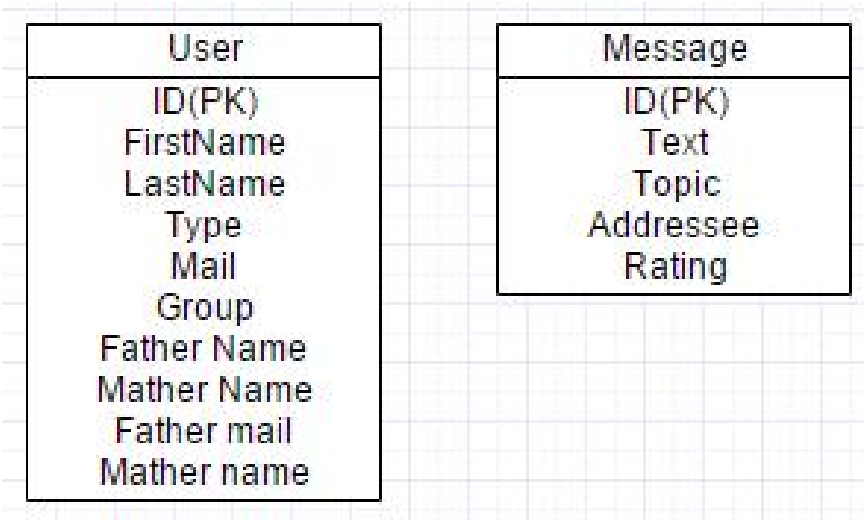


Рисунок 2.10 - Структура діаграми ERD

Висновки до розділу 2:

1) Побудовані діаграми діяльності програмного продукту, для наочного зображення основних процесів та дії, які виконуватиме система.

2) Побудовані діаграми потоків даних DFD розроблюваної системи, які відображають роботу системи при виконанні тих чи інших операцій.

3) Спроектована ER діаграма бази даних додатку, яка дозволяє розробнику переглянути, які дані мають зберігатися в базі даних і, як вони повинні взаємодіяти між собою.

РОЗДІЛ 3
ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Програмна реалізація проекту

Android - операційна система для комунікаторів, планшетних комп'ютерів, цифрових програвачів, наручних годинників, нетбуків і смартфонів, заснована на ядрі Linux[5].

Додатки під операційну систему Android розробляються в основному з використанням Java. Скомпільований програмний код (разом з усіма файлами ресурсів та іншої необхідної інформацією) упаковується в спеціальний файл-архів, Android Package. Цей файл має розширення * .apk і упаковується спеціальною утилітою aapt tool. Саме він надалі поширюється як програма і інсталюється на мобільні пристрої. Один такий файл пов'язаний з кодом однієї програми. І кожен додаток в Android живе у своєму власному світі - в такій машині. За замовчуванням, кожна програма виконується в своєму власному процесі, управлінням якого займається ядро Linux, яке також здійснює менеджмент пам'яті. Таким чином, найчастіше код додатку виконується в ізоляції від усіх інших додатків[6].

Мова Java є однією з наймолодших в сімействі мов програмуваня і була розроблена з розрахунку на те, щоб професійний програміст міг легко її опанувати та ефективно використовувати. За основу Java взятий синтаксис C++ - безсумнівно однієї з найбільш популярних мов програмування сучасності. Проте, Java - це цілком самостійна мова програмування, і при її створенні не йшлося про будь-яку сумісність з C++. Тому деякі механізми реалізовані в Java інакше, а деякі зовсім відсутні. Ідеологічно ж Java побудована дещо інакше ніж C++. Розробники Java грунтувалися на досвіді розробки програм на C++ і прагнули позбутися можливостей, які зарекомендували себе непевними. Так, в Java відсутня перегрузка операторів

а також автоматичне приведення несумісних типів - конструкції, які при неуважному використанні є джерелом важких для виявлення помилок. Взагалі, інтерфейси Java більш прості та прозорі для розуміння. Написати на Java програму з графічним інтерфейсом значно легше. Звичайно, простота інтерфейсів компенсується меншою гнучкістю, бібліотека Java не така багата, як стандартні бібліотеки C/C++. Але згадаймо, що Java задуманий для використання на різних платформах і тому реалізує в собі найбільш стандартні можливості задля легшої адаптації під конкретне середовище[7]

Для реалізації проекту було обрано середовище розробки Android Studio. Android Studio — інтегроване середовище розробки (IDE) для платформи Android, представлене 16 травня 2013 року на конференції Google I/O менеджером по продукції корпорації Google — Еллі Паверс (Ellie Powers). 8 грудня 2014 року компанія Google випустила перший стабільний реліз Android Studio 1.0.

Android Studio прийшов на зміну плагіну ADT для платформи Eclipse. Середовище побудоване на базі сирцевих текстів продукту IntelliJ IDEA Community Edition, що розвивається компанією JetBrains. Android Studio розвивається в рамках відкритої моделі розробки та поширюється під ліцензією Apache 2.0[8].

Середовище розробки адаптоване для виконання типових завдань, що вирішуються в процесі розробки застосунків для платформи Android.[4] У тому числі у середовище включені засоби для спрощення тестування програм на сумісність з різними версіями платформи та інструменти для проектування застосунків, що працюють на пристроях з екранами різної роздільності (планшети, смартфони, ноутбуки, годинники, окуляри тощо). Крім можливостей, присутніх в IntelliJ IDEA, в Android Studio реалізовано кілька додаткових функцій, таких як нова уніфікована підсистема складання, тестування і розгортання застосунків, заснована на складальному інструментарії Gradle і підтримуюча використання засобів безперервної інтеграції.

Для прискорення розробки застосунків представлена колекція типових елементів інтерфейсу і візуальний редактор для їхнього компонування, що надає зручний попередній перегляд різних станів інтерфейсу застосунку (наприклад, можна подивитися як інтерфейс буде виглядати для різних версій Android і для різних розмірів екрану). Для створення нестандартних інтерфейсів присутній майстер створення власних елементів оформлення, що підтримує використання шаблонів. У середовище вбудовані функції завантаження типових прикладів коду з GitHub.

До складу також включені пристосовані під особливості платформи Android розширені інструменти рефакторингу, перевірки сумісності з минулими випусками, виявлення проблем з продуктивністю, моніторингу споживання пам'яті та оцінки зручності використання. У редактор доданий режим швидкого внесення правок. Система підсвічування, статичного аналізу та виявлення помилок розширена підтримкою Android API. Інтегрована підтримка оптимізатора коду ProGuard. Вбудовані засоби генерації цифрових підписів. Надано інтерфейс для управління перекладами на інші мови.

Все що ми бачимо на екрані нашого смартфону візуалізується при інтерпретації xml коду, генерування якого контролюється java кодом (контролером) який вставляє потрібні об’єкти в потрібні місця в xml. Нижче наведено приклад xml та java коду вікна авторизації в системі, на рисунку 3.1 представлено скріншот початкового вікна системи.

XML код вікна «Авторизація»:

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:background="@drawable/loginbackground"
    tools:context="com.example.vova.ratingsystemtneu.Login"
    style="@style/Base.Theme.AppCompat.Light.DarkActionBar"
    android:backgroundTintMode="src_over">
```

```
    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/login"
        android:layout_above="@+id/pass"
        android:layout_alignLeft="@+id/pass"
        android:layout_alignStart="@+id/pass"
        android:layout_alignRight="@+id/pass"
        android:layout_alignEnd="@+id/pass"
        android:gravity="center" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/login"
        android:id="@+id/Login"
        android:layout_centerVertical="true"
        android:layout_alignLeft="@+id/pass"
        android:layout_alignStart="@+id/pass"
        android:layout_toStartOf="@+id/sign"
        android:layout_toLeftOf="@+id/sign" />
    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:inputType="textPassword"
        android:ems="10"
        android:id="@+id/pass"
        android:layout_above="@+id/Login"
        android:layout_centerHorizontal="true"
        android:gravity="center" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Sign up"
        android:id="@+id/sign"
        android:layout_alignTop="@+id/Login"
        android:layout_alignRight="@+id/pass"
        android:layout_alignEnd="@+id/pass"/>
</RelativeLayout>
```

Java код вікна «Авторизація»:

```
public class Login extends AppCompatActivity implements View.OnClickListener
{
    SharedPreferences sPref;
    final String SAVED_TEXT = "saved_text";
    Boolean isInternetPresent = false;
    ConnectionDetector cd;
    EditText login;
    EditText password;
    Student resurs = new Student();
    private static final String TAG = "myLogs";
```

```
Button log;
Button signUp;
ProgressDialog progressDialog;    @Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_login);
    progressDialog = new ProgressDialog(Login.this);
    progressDialog.setTitle("");
    cd = new ConnectionDetector(getApplicationContext());
    login = (EditText) findViewById(R.id.login);
    password = (EditText) findViewById(R.id.pass);
    login.setFilters(new InputFilter[]{filter});
    log = (Button) findViewById(Login);
    signUp = (Button) findViewById(sign);
    log.setOnClickListener(this);
    signUp.setOnClickListener(this);
    loadText();
}    @Override
public void onClick(View v){
    switch (v.getId()) {
        case R.id.Login:
            proces();
            saveText();
            break;
        case R.id.sign:
            final Intent signIntent = new Intent(this, sign_up.class);
            startActivity(signIntent);
         break;
    }
}
```

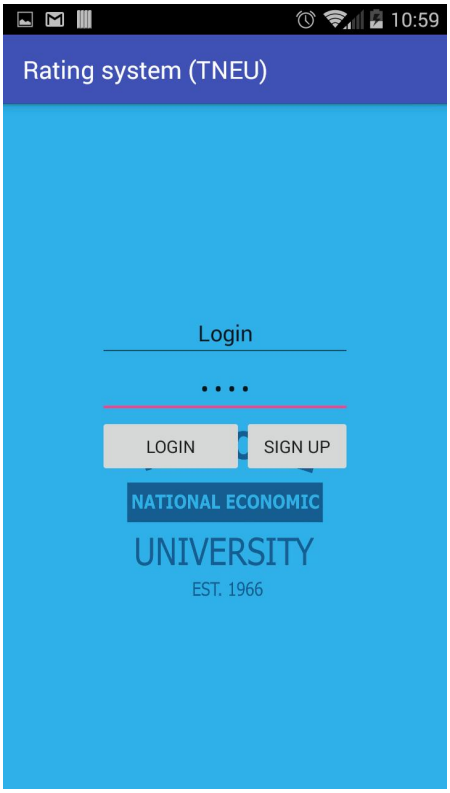


Рисунок 3.1 - Вікно «Авторизація»

Далі наведено приклад java-коду отримання json успішності конкретного студента з Rating system ТНЕУ:

```
@Override
protected String doInBackground(String... params) {
    HttpURLConnection connection = null;
    BufferedReader reader = null;
    try {
        URL url = new URL(params[0]);
        connection = (HttpURLConnection) url.openConnection();
        connection.connect();
        InputStream stream = connection.getInputStream();
        reader = new BufferedReader(new InputStreamReader(stream));
        String line = "";
        while ((line = reader.readLine()) != null) {
            buffer.append(line);
        }
        JSONTparsText = buffer.toString();
    } catch (MalformedURLException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    } finally {
        if (connection != null) {
            connection.disconnect();
        }
        try {
            if (reader != null) {
                reader.close();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
    return JSONTparsText;
}
```

Далі наведено приклад парсингу json в список типу Subjects:

```
public List<Subjects> stringPars(String parsText, String semestr){
    List<Subjects> firstSemestrsList = new ArrayList<>();

    try {
        JSONObject jsonObject = new JSONObject(parsText);
        JSONObject studentJSON = jsonObject.getJSONObject("student");
        JSONObject firstSemester = studentJSON.getJSONObject(semestr);

        JSONArray subjects = firstSemester.getJSONArray("subjects");

        for (int i = 0; i < subjects.length(); i++) {
            Subjects subject = new Subjects();

            subject.setName(subjects.getJSONObject(i).getString("name"));

subject.setControlType(subjects.getJSONObject(i).getString("controlType"));
            subject.setTotalScore(subjects.getJSONObject(i).getInt("totalScore"));
            JSONArray moduls;
            moduls = subjects.getJSONObject(i).getJSONArray("modules");
```

```
                List<Subjects.Modules> modulesList = new ArrayList<>();
                for (int j = 0; j < moduls.length(); j++) {
                    Subjects.Modules module = new Subjects.Modules();
                    module.setDate(moduls.getJSONObject(j).getString("date"));
                    module.setWeight(moduls.getJSONObject(j).getInt("weight"));
                    module.setScore(moduls.getJSONObject(j).getInt("score"));
                    modulesList.add(module);
                }
                subject.setModulList(modulesList);
                firstSemestrsList.add(subject);
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
        return firstSemestrsList;
    }
```

Приклади xml-коду адаптера для відображення успішності студента в системі та заповнення його даними наведено у додатку А. На рисунку 3.2 представлено скріншот відображення успішності студента.

Рисунок 3.2 - Вікно відображення успішності студента.

Весь лістинг програмної системи знаходиться у додатку Б.

3.2 Програмна реалізація бази даних

База даних реалізована на СУБД MS SQL-SERVER. СУБД SQL-Server з'явилася в 1989 році та з того часу зазнала багато змін. Значні зміни зазнали масштабованість продукту, його цілісність, зручність адміністрування, продуктивність і функціональні можливості.

Microsoft SQL Server - це реляційна система управління базами даних (СКБД). У реляційних базах даних дані зберігаються в таблицях. Взаємопов'язані дані можуть групуватися в таблиці, крім того, можуть бути встановлені також і взаємовідносини між таблицями. Звідси і пішла назва реляційні - від англійського слова relational (споріднений, пов'язаний відносинами, взаємозалежний). Користувачі отримують доступ до даних на сервері через додатки, а адміністратори, виконуючи завдання конфігурування, адміністрування та підтримки бази даних, виробляють безпосередній доступ до сервера. SQL Server є масштабованої базою даних, це означає, що вона може зберігати значні обсяги даних і підтримувати роботу багатьох користувачів, що здійснюють одночасний доступ до бази даних.

Microsoft SQL Server 6.5 - одна з найбільш потужних СУБД архітектури клієнт-сервер. Ця СУБД дозволяє задовольняти такі вимоги, що пред'являються до систем розподіленої обробки даних, як тиражування даних, рівнобіжна обробка, підтримка великих баз даних на відносно недорогих апаратних платформах при збереженні простоти управління і використання.

MS SQL Server не призначений безпосередньо для розробки користувальних додатків, а виконує функції керування базою даних. Сервер має засоби віддаленого адміністрування і керування операціями, організовані на базі об'єктно-орієнтованої розподіленої середовища управління.

Приведемо приклад коду створення відношень бази даних:

```
CREATE TABLE Messages (
    ID        int IDENTITY NOT NULL,
    Rating    int NOT NULL,
    Topic     varchar(50) NOT NULL,
    Text      varchar(50) NOT NULL,
    Addressee varchar(50) NOT NULL,
    PRIMARY KEY (ID));
```

На рисунку 3.4 представлено дизайн таблиці Messages.

	Column Name	Data Type	Allow Nulls
🔑	ID	int	☐
	Rating	int	☐
	Topic	varchar(50)	☐
	mText	varchar(50)	☐
	Addressee	varchar(50)	☐

Рисунок 3.3 – дизайн таблиці Messages

```
CREATE TABLE User (
    ID          int IDENTITY NOT NULL,
    FirstName   varchar(50) NOT NULL,
    LastName    varchar(50) NOT NULL,
    uType       bit NOT NULL,
    Mail        varchar(50) NOT NULL,
    uGroup      varchar(10) NOT NULL,
    Father_Name  varchar(50),
    Mather_Name  varchar(50),
    Father_mail  varchar(50),
    Mather_mail  varchar(50),
    PRIMARY KEY (ID));
```

На рисунку 3.5 представлено дизайн таблиці User.

	Column Name	Data Type	Allow Nulls
🔑	ID	int	☐
	FirstName	varchar(50)	☐
	LastName	varchar(50)	☐
	uType	bit	☐
	Mail	varchar(50)	☐
	uGroup	varchar(10)	☐
	Father_Name	varchar(50)	☑
	Mather_Name	varchar(50)	☑
	Father_mail	varchar(50)	☑
	Mather_mail	varchar(50)	☑

Рисунок 3.4 - дизайн таблиці User

Java код функції ідентифікації користувача в системі:

```
public Integer getUserType(String login, String password){
        Integer res = null;
        Cursor c = null;
        String[] columns = new String[]{"type"};
        String having = "login = '" + login +  "' and password = '" + password +
"';";
        c = _db.query("User", columns, having, null, null, null, null);
        if (c.moveToFirst()) {
            if (c.moveToFirst()) {
                int idColIndex = c.getColumnIndex("type");
                do {
                    Log.d(LOG_TAG,
                            "type = " + c.getInt(idColIndex) +"----------------------
--------");
                    res =  c.getInt(idColIndex);
                } while (c.moveToNext());
            }
        }else {
            Log.d(LOG_TAG, "0 rows");
            res = 0;
        }
        c.close();
        return res;
    }
```

Java код функції отримання списку студентів певної групи:

```
public List<StudentDataList> getNamesArray(String group) {
        List<StudentDataList> users = new ArrayList<>();
        Cursor c = null;
        String[] columns = new String[]{"lname",
"fname","login","password","email","mather_name",
                "father_name","mather_mail","father_mail"};
        String having = "agroup = '" + group + "' and type = '0';";
        c = _db.query("User", columns, having, null, null, null,"lname");
        if (c.moveToFirst()) {
            int nameColIndex = c.getColumnIndex("fname");
            int lnameColIndex = c.getColumnIndex("lname");
            int passwordColIndex = c.getColumnIndex("password");
            int loginColIndex = c.getColumnIndex("login");
            int emailColIndex = c.getColumnIndex("email");
            int matherNameColIndex = c.getColumnIndex("mather_name");
            int fatherNameColIndex = c.getColumnIndex("father_name");
            int matherMailColIndex = c.getColumnIndex("mather_mail");
            int fatherMailColIndex = c.getColumnIndex("father_mail");
            do {
                StudentDataList studentDataList = new StudentDataList();
                String nameStr = c.getString(nameColIndex) +
                                 " " + c.getString(lnameColIndex);
                String logStr = "login=" + c.getString(loginColIndex);
                String pasStr = "password=" + c.getString(passwordColIndex);
                String matherNam = c.getString(matherNameColIndex);
                String fatherNam = c.getString(fatherNameColIndex);
                String matherMail = c.getString(matherMailColIndex);
                String fatherMail = c.getString(fatherMailColIndex);
                String email = c.getString(emailColIndex);
                Log.d(LOG_TAG, nameStr);
                studentDataList.setName(nameStr);
                studentDataList.setLog(logStr);
                studentDataList.setPass(pasStr);
                studentDataList.setFatherMail(fatherMail);
                studentDataList.setMatherMail(matherMail);
                studentDataList.setMatherName(matherNam);
                studentDataList.setFatherName(fatherNam);
                studentDataList.setMail(email);
                studentDataList.setPass(pasStr);
                users.add(studentDataList);
            } while (c.moveToNext());
        } else{
            StudentDataList studentDataList = new StudentDataList();
            studentDataList.setName("0 rows");
            studentDataList.setLog(null);
            studentDataList.setPass(null);
            users.add(studentDataList);
            Log.d(LOG_TAG, "0 rows");
        }
        c.close();
        return users;
    }
```

Висновки до розділу 3:

1) Розроблений основний функціонал нашої системи, наведенні приклади реалізації основних функцій додатку. Зокрема, авторизації в системі та відображення успішності.

2) Наведено DDL-код створення відношень у БД і java-коди функцій ідентифікації користувача в системі, та функції отримання списку студентів групи.

РОЗДІЛ 4
ТЕСТУВАННЯ ТА ДОСЛІДНА ЕКСПЛУАТАЦІЯ

4.1 Тестування

Невід'ємною частиною будь-якої реалізації програмного продукту є процес тестування або так званої перевірки роботи програми згідно вимог, які були зазначенні при початку реалізації продукту. Під час перевірки програмного продукту зазвичай виявляються дефекти системи, при яких система може не працювати, блокуватися або працювати не вірно. В разі виявлення дефекту або помилки, розробник має виправити її.

Тестування за своєю специфікою поділяється на два види: ручне тестування та автоматизоване. Під час ручного тестування всі операції виконуються вручну, тобто самі перевіряємо і оцінюємо, як працює та чи інша функція[13]. Автоматизоване тестування дає змогу перевірити код реалізації функцій, за допомогою якого маємо можливість протестувати фунціонал, і перевірити, чи повертається потрібний результат. Кожен з цих видів містить в собі вже свої види тестування.

Було проведено такі види тестування системи:

- Інсталяційне тестування. Під час проведення даного тестування додаток було успішно встановлено на смартфони з такими операційними системами:
  - Android 4.4.4(Kit Kat)
  - Android 5.0(Lollipop)
  - Android 5.1(Lollipop)
  - Android 6.0(Marshmallow)
- Тестування продуктивності. Під час проведення даного тестування смартфон продовжував успішно працювати з додатком при таких умовах:
  - низькому рівні заряду акумулятора;

- поганому покритті мережею;
- низькій кількості доступної пам'яті;
- одночасному доступі до сервера кількома користувачами.

- Навантажувальне тестування:
  - на смартфонах запускалося декілька сторонніх додатків які використовували підключення до мережі інтернет;
- Тестування перериванням:
  - Отримання повідомлень під час роботи з додатком;
  - Відповідь на дзвінок під час роботи з додатком;
  - Підключення та відключення USB кабелю.
- Також проведено тестування при різних типах підключення до мережі інтернет(2g, 3g, WiFi), яке успішно завершилося у всіх трьох випадках.

Всі тести додаток пройшов успішно, додаток готовий для використання та експлуатації, фатальних багів виявлено не було.

4.2 Розгортання програмного продукту

Для запуску системи програмного продукту потрібна операційна система Android версії 4 та вище. Відносно апаратних вимог, то процесор має бути покоління MT6513 1 x 650 GHz та вище, оперативної пам’яті від 256 МБ та 50 МБ вільного простору на диску.

4.3 Інструкція користувача

Для початку роботи з додатком потрібно завантажити файл формату .apk, та просто відкрити його на своєму смартфоні, після чого Android сам розпакує та створить всі необхідні для додатку файли та створить ярлик програми на робочому столі та іконку в головному меню. Далі для того щоб

почати користуватися додатком потрібно просто натиснути на нього на екрані вашого пристрою. Також невід'ємною умовою функціонування додатку є наявність підключення до мережі Інтернет.

Для того щоб куратор міг бачити список усіх своїх студентів йому потрібно бути зареєстрованим в системі, також усім студентам теж потрібно зареєструватися в системі вказавши назву групи у якій вони навчаються, інакше список буде не повний. Скріншот вікна «Реєстрація» в системі представлено на рисунку 4.1.

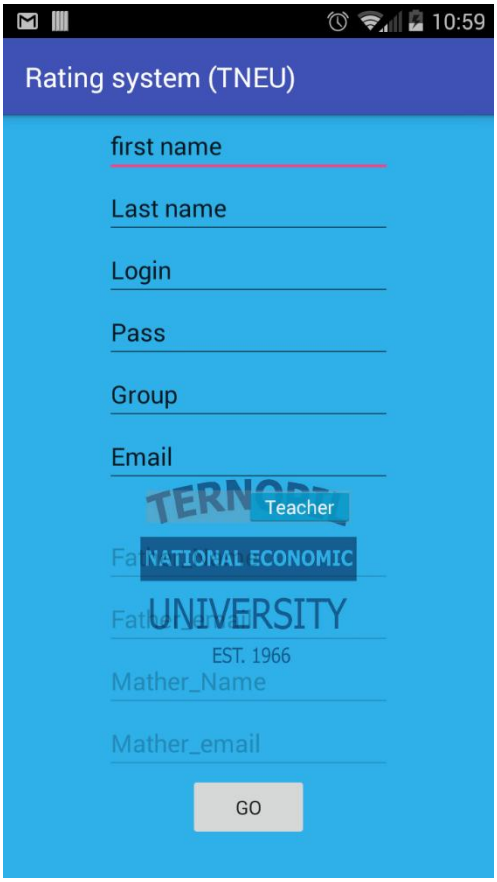


Рисунок 4.1 - Вікно «Реєстрація»

Студенту ж непотрібно робити ніяких додаткових кроків щоб побачити свою успішність, достатньо просто відкрити додаток, ввести логін та пароль від рейтингової системи ТНЕУ та натиснути клавішу «Login». Скріншот вікна «Авторизація» в системі представлено на рисунку 4.2.

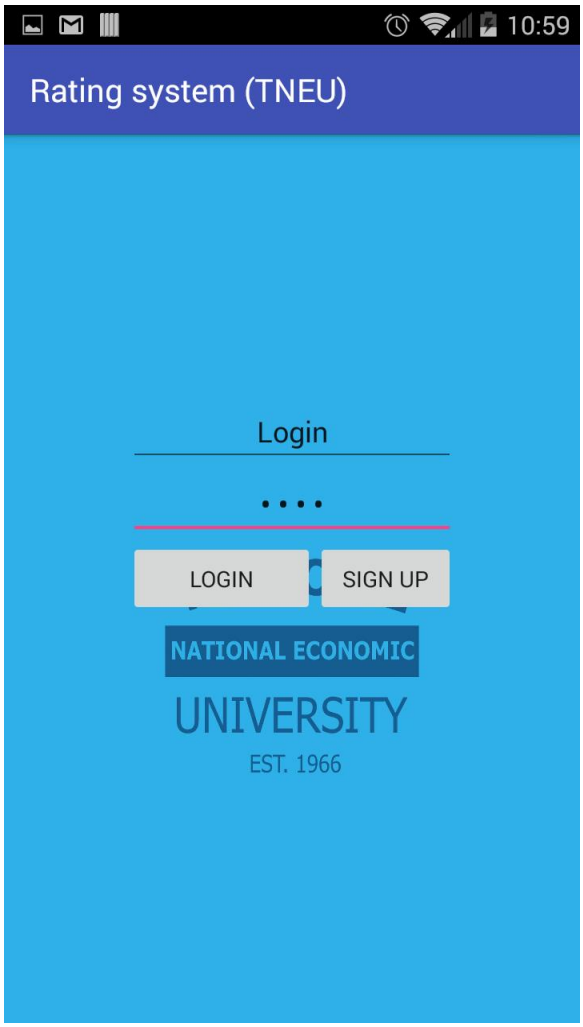


Рисунок 4.2 - Вікно «Авторизація»

Після успішної авторизації, система ідентифікує користувача як «Куратор» або «Студент». Прии ідентифікації користувача як «Куратор» система виведе на екран список тих студентів, які навчаються в академічній групі під його кураторством, скріншот вікна «Список студентів групи» представлений на рисунку 4.3.

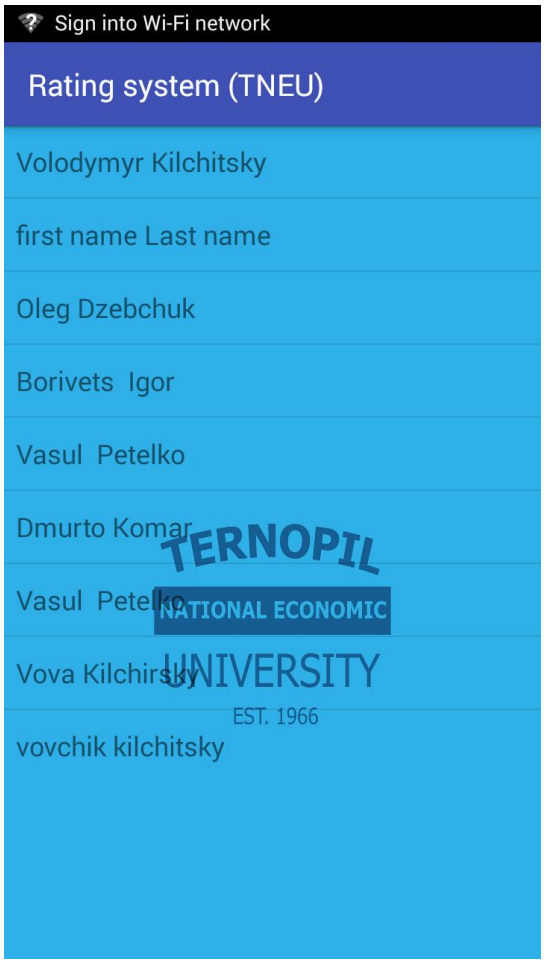


Рисунок 4.3 – Вікно «Список студентів групи»

Для того щоб отримати детальну інформацію про успішність конкретного студента, куратору потрібно знайти його у виведеному на екрані списку та просто натиснути на нього(вибрати його із списку) і система покаже йому повну інформацію про успішність студента(кількість пропусків, та оцінка за кожен з навчальних предметів), скріншот вікна «Перегляд успішності» представлений на рисунку 4.4. Система відобразить таке ж вікно при ідентифікації користувача як «Студент», де буде відображена його власна успішність за два семестри одного навчального року, між якими можна переключатися за допомогою switch.

Рисунок 4.4 – Вікно «Перегляд успішності»

Якщо куратор бачить негативну оцінку в списку, він просто вибирає предмет із списку. Після чого система запропонує йому форму з автоматично згенерованим текстом повідомлення, який при бажанні можна змінити, через яку можна відправити повідомлення батькам студента чи самому студенту про його успіхи по даному напрямку навчання. Для того щоб повідомлення було відправлено декільком адресатам(обом батькам студента, та йому самому) достатньо просто відзначити адресатів напроти їхніх імен, та натиснути кнопку «Send», скріншот вікна «Відправка повідомлення» представлений на рисунку 4.5.

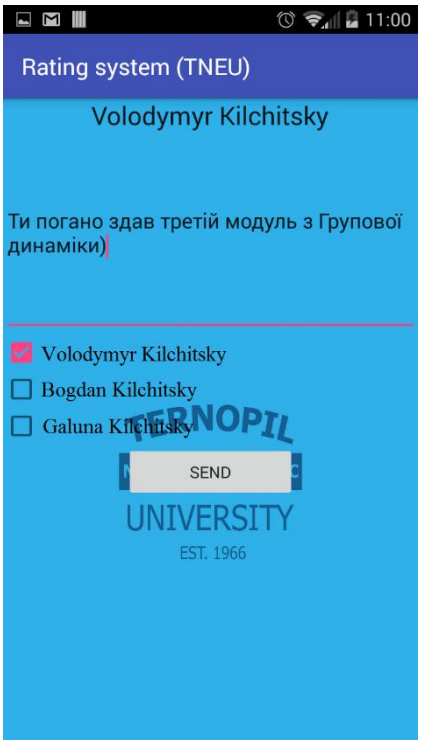


Рисунок 4.5 - Вікно «Відправка повідомлення»

Після цього система виведе на список ваших емейл клієнтів для надсилання повідомлень, скріншот вікна «Список емейл клієнтів» представлений на рисунку 4.6.

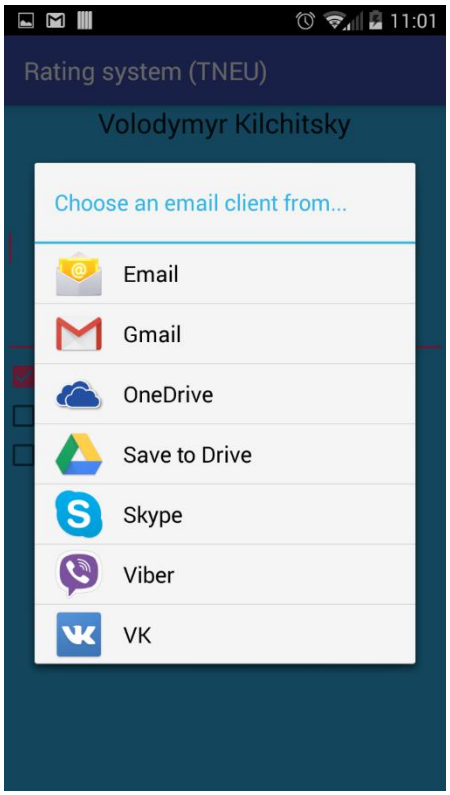


Рисунок 4.6 - Вікно «Список емейл клієнтів»

Після цього система відкриє вибраний користувачем емейл клієнт, із вставленим текстом повідомлення, темою та списком адресатів, і йому залишається просто натиснути клавішу відправки повідомлення, і вони будуть надіслані як прості емейл листи всім вказаним адресатам, скріншот вікна «Повідомлення в емейл клієнті» представлений на рисунку 4.7.

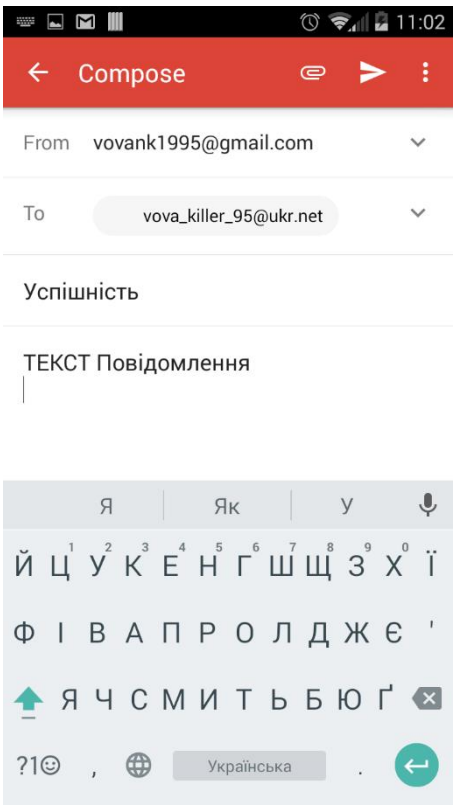


Рисунок 4.7 - Вікно «Повідомлення в емейл клієнті»

При виникненні помилок з авторизацією, потрібно перевірити чи працює сайт Rating system ТНЕУ (режим доступу: http://mod.tanet.edu.te.ua/), тому що додаток бере всю потрібну йоку інформацію для відображення з нього.

Базові функції системи:

- реєстрація користувачів в системі;
- авторизації користувачів в системі і надання їм відповідних прав доступу в системі.
- перегляд успішності;

- відправка email студентам та їх батькам в разі необхідності куратором.

Для того щоб куратор бачив весь список своїх студентів, їм всім потрібно бути зареєстрованими в системі, з правильно вказаним полем групи в котрій вони навчаються.

Висновки до розділу 4:

1. Проведене тестування програми на наявність помилок або дефектів, яке показало, що додаток повністю придатний для користування.

2. Створено інструкцію для користувача програми, у якій детально описано які кроки потрібно зробити щоб використовувати додаток на повну, та описано всі можливі сценарії використання програми різними типами користувачів.

ВИСНОВКИ

Під час створення програмної системи у дипломній роботі було пройдено усі етапи, а саме:

1. На етапі аналізу предметної області було досліджено предметну область – рейтенгову систуму ТНЕУ. Проведено огляд та аналіз кількох веб-систем аналогів, що реалізують схожі функції предметної області.
2. На етапі проектування програмної системи розроблено її архітектуру та представлено у вигляді класів програми. Вивчено інтерфейс серверної частини та способи інтеграції.
3. На етапі програмної реалізації, з використанням об’єктно-орієнтованого підходу в інтегрованому середовищі розробки програмного забезпечення Android Studio, реалізовано функції системи, які написані на мові програмування java з імплементацією запитів до серверу. Розроблено інтерфейс програми.
4. На етапі тестування було проведено функціональне та тестування безпеки. Також розроблено інструкцію користувача, де вказується правила використання програми.

Програмна система дає можливість швидкого та зручного доступу до рейтенгової системи ТНЕУ студентами, та кураторами які в свою чергу можуть повідомити сттудента та його батьків про успіхи його самого, чи їх сина, що збільшує стимул навчання кожного студента. Після затвердження її керівництвом університету, дану програму буде введено в подальшу експлуатацію.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1) [Електронний ресурс]. – Режим доступу: http://npu.edu.ua/!ebook/book/html/D/ispu_kiovist_Ficyla_Pedagogika_VSh/1020.html
2) Н.М. Гаркуша, О.В. Цуканова, О.О. Горошанська. Моделі і методи прийняття рішень в аналізі та аудиті: Навч. посіб. — 2-ге вид. Рекомендовано МОН / Гаркуша Н.М., Цуканова О.В., Горошанська О.О. — К., 2012. — 591 с.
3) Вендров А.М. Проектирование программного обеспечения экономических информационных систем: Учебник. – 2-е изд., перераб. и доп.– М.: Финансы и статистика, 2006. – 544 с: ил.
4) Випущені вихідні коди Android [Електронний ресурс]. - Режим доступу : URL : <http://habrahabr.ru/post/132603/>
5) Android 4.2 [Електронний ресурс]. - Режим доступу : URL : <http://zoom.cnews.ru/publication/printed/40400>
6) Основи безпеки операційної системи Android [Електронний ресурс]. - Режим доступу : URL : <http://habrahabr.ru/post/176131/>
7) Google Android: ази розробки [Електронний ресурс]. - Режим доступу : URL : <http://www.pcmag.ru/solutions/sub_detail.php?ID=14240>
8) [Електронний ресурс]. – Режим доступу: http://javaland.com.ua/programuvannya/
9) [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Java
10) [Електронний ресурс]. – Режим доступу: http://ami.lnu.edu.ua/books/Java/01.html
11) [Електронний ресурс]. – Режим доступу: http://ami.lnu.edu.ua/books/Java/01.html
12) [Електронний ресурс]. – Режим доступу: http://www.sun-awards.com.ua/articles/stati_o_java/effect.html
13) 12. [Електронний ресурс]. – Режим доступу: http://bius.ru/java-book/30-CHto-takoe-Java/383-bogataya-obektnaya-sreda.html

ДОДАТОК А

Код компонентів вікна відображення успішності

Приклад xml-коду адаптера для відображення успішності студента в системі:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical" >

    <LinearLayout
        android:orientation="horizontal"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:baselineAligned="false">

        <LinearLayout
            android:layout_weight="2"
            android:orientation="vertical"
            android:layout_width="match_parent"
            android:layout_height="match_parent">

            <LinearLayout
                android:orientation="vertical"
                android:layout_width="match_parent"
                android:layout_height="wrap_content">

                <TextView
                    android:layout_width="match_parent"
                    android:layout_height="64dp"
                    android:textAppearance="?android:attr/textAppearanceLarge"
                    android:text="Системний аналіз та проектування комп\u0027ютерних
інформаційних систем"
                    android:id="@+id/subjectNsme"
                    android:gravity="center_vertical|left"
                    android:layout_gravity="center_vertical|left"
                    android:textSize="18dp"
                    android:layout_marginBottom="10dp"/>

                <LinearLayout
                    android:orientation="horizontal"
                    android:layout_width="match_parent"
                    android:layout_height="match_parent">

                    <LinearLayout
                        android:layout_weight="1"
                        android:orientation="vertical"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content">

                        <TextView
                            android:layout_width="wrap_content"
                            android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceSmall"
                            android:text="qwert"
```

```
                android:textSize="14dp"
                android:id="@+id/date"
                android:layout_marginBottom="3dp"
                android:layout_gravity="center" />

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceSmall"
                android:text="qwert"
                android:textSize="15dp"
                android:layout_marginBottom="3dp"
                android:id="@+id/weight"
                android:layout_gravity="center" />

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceSmall"
                android:text="qwert"
                android:textSize="15dp"
                android:id="@+id/score"
                android:layout_gravity="center" />
        </LinearLayout>

        <LinearLayout
            android:layout_weight="1"
            android:orientation="vertical"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" >

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
android:textAppearance="?android:attr/textAppearanceSmall"
                android:text="qwert"
                android:textSize="14dp"
                android:id="@+id/date2"
                android:layout_marginBottom="3dp"
                android:layout_gravity="center" />

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceSmall"
                android:text="qwert"
                android:textSize="15dp"
                android:layout_marginBottom="3dp"
                android:id="@+id/weight2"
                android:layout_gravity="center" />

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceSmall"
                android:text="qwert"
                android:textSize="15dp"
```

```
                android:id="@+id/score2"
                android:layout_gravity="center" />
        </LinearLayout>

        <LinearLayout
            android:layout_weight="1"
            android:orientation="vertical"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" >

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
android:textAppearance="?android:attr/textAppearanceSmall"
                android:text="qwert"
                android:textSize="14dp"
                android:id="@+id/date3"
                android:layout_marginBottom="3dp"
                android:layout_gravity="center" />

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
android:textAppearance="?android:attr/textAppearanceSmall"
                android:text="qwert"
                android:textSize="15dp"
                android:layout_marginBottom="3dp"
                android:id="@+id/weight3"
                android:layout_gravity="center" />

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
android:textAppearance="?android:attr/textAppearanceSmall"
                android:text="qwert"
                android:textSize="15dp"
                android:id="@+id/score3"
                android:layout_gravity="center" />
        </LinearLayout>

        <LinearLayout
            android:layout_weight="1"
            android:orientation="vertical"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content" >

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceSmall"
                android:text="qwert"
                android:textSize="14dp"
                android:id="@+id/date4"
                android:layout_marginBottom="3dp"
                android:layout_gravity="center" />

            <TextView
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
```

```
android:textAppearance="?android:attr/textAppearanceSmall"
                            android:text="qwert"
                            android:textSize="15dp"
                            android:layout_marginBottom="3dp"
                            android:id="@+id/weight4"
                            android:layout_gravity="center" />
                        <TextView
                            android:layout_width="wrap_content"
                            android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceSmall"
                            android:text="qwert"
                            android:textSize="15dp"
                            android:id="@+id/score4"
                            android:layout_gravity="center" />
                    </LinearLayout>

                    <LinearLayout
                        android:layout_weight="1"
                        android:orientation="vertical"
                        android:layout_width="wrap_content"
                        android:layout_height="wrap_content" >

                        <TextView
                            android:layout_width="wrap_content"
                            android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceSmall"
                            android:text="qwert"
                            android:textSize="14dp"
                            android:id="@+id/date5"
                            android:layout_marginBottom="3dp"
                            android:layout_gravity="center" />

                        <TextView
                            android:layout_width="wrap_content"
                            android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceSmall"
                            android:text="qwert"
                            android:textSize="15dp"
                            android:layout_marginBottom="3dp"
                            android:id="@+id/weight5"
                            android:layout_gravity="center" />

                        <TextView
                            android:layout_width="wrap_content"
                            android:layout_height="wrap_content"

android:textAppearance="?android:attr/textAppearanceSmall"
                            android:text="qwert"
                            android:textSize="15dp"
                            android:id="@+id/score5"
                            android:layout_gravity="center" />
                    </LinearLayout>
                </LinearLayout>
            </LinearLayout>
        </LinearLayout>

        <LinearLayout
            android:layout_weight="6"
```

```
            android:orientation="vertical"
            android:layout_width="match_parent"
            android:layout_height="match_parent">
            <TextView
                android:layout_weight="1"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:textAppearance="?android:attr/textAppearanceLarge"
                android:text="100"
                android:id="@+id/subtotalScore"
                android:textStyle="bold"
                android:textSize="@dimen/abc_action_bar_progress_bar_size"
                android:gravity="center"
                android:layout_gravity="center" />

            <FrameLayout
                android:layout_weight="1"
                android:layout_width="wrap_content"
                android:layout_height="100dp">
                <ImageView
                    android:layout_weight="1"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:id="@+id/subcontrolType"
                    android:src="@drawable/zalik"
                    android:foregroundGravity="center"
                    android:layout_gravity="center_vertical" />
            </FrameLayout>
        </LinearLayout>
    </LinearLayout>
</LinearLayout>
```

Заповнення вікна данними:

```
public class ShowProgress extends AppCompatActivity {

    private static final String TAG = "myLogs";
    ListView listView1;
    Switch toggle;
    String semestr;
    List<Subjects> first;
    StringPars stringPars;
    String parsText;
    Integer userType;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.show_progress);
        parsText = getIntent().getStringExtra("parsText");
        userType = getIntent().getIntExtra("UserType", 0);
        stringPars= new StringPars();
        toggle = (Switch) findViewById(R.id.switch1);
        toggle.setOnCheckedChangeListener(new
CompoundButton.OnCheckedChangeListener() {
            @Override
            public void onCheckedChanged(CompoundButton buttonView, boolean
isChecked) {
                if (isChecked) {
                    semestr = "secondSemester";
```

```
                    show(parsText,semestr);
                } else {
                    semestr = "firstSemester";
                    show(parsText, semestr);
                }
            }
        });
        if(toggle.isChecked()){
            semestr = "secondSemester";
        }
        else {
            semestr = "firstSemester";
            if(iff(parsText,"secondSemester") == true) {
                toggle.setEnabled(false);
                Context context = getApplicationContext();
                CharSequence text = "Second semester is empty!";
                int duration = Toast.LENGTH_SHORT;

                Toast toast = Toast.makeText(context, text, duration);
                toast.setGravity(Gravity.CENTER,0,0);
                toast.show();
            }
        }
        show(parsText,semestr);

        if(userType == 1){

            final Intent intent = new Intent(this, SendMail.class);
            final String name = getIntent().getStringExtra("Name");
            final String email = getIntent().getStringExtra("Email");
            final String matherName = getIntent().getStringExtra("MatherName");
            final String fatherName = getIntent().getStringExtra("FatherName");
            final String matherMail = getIntent().getStringExtra("MatherMail");
            final String fatherMail = getIntent().getStringExtra("FatherMail");

            listView1.setOnItemClickListener(new AdapterView.OnItemClickListener() {
                @Override
                public void onItemClick(AdapterView<?> parent, View view, int
position, long id) {
                    Integer itemPosition = position;

                    intent.putExtra("Name", name);
                    intent.putExtra("Email", email);
                    intent.putExtra("MatherName", matherName);
                    intent.putExtra("FatherName", fatherName);
                    intent.putExtra("MatherMail", matherMail);
                    intent.putExtra("FatherMail", fatherMail);

                    startActivity(intent);
                }
            });
        }
    }

    private void show(String parsText,String semestr){
        first = stringPars.stringPars(parsText, semestr);

        listView1 = (ListView) findViewById(R.id.listview1);
        YourAdapter adapter = new YourAdapter(this, first);
        listView1.setAdapter(adapter);
    }
```

```
    private boolean iff(String parsText,String semestr){
        if(stringPars.stringPars(parsText, semestr).isEmpty()) {
            return true;
        }
        else {
            return false;
        }
    }
}
```

ДОДАТОК Б

Лістинг програмної системи

Лістинг класу Login.java:

```
public class Login extends AppCompatActivity implements View.OnClickListener
//implements View.OnClickListener
{

    SharedPreferences sPref;
    final String SAVED_TEXT = "saved_text";

    Boolean isInternetPresent = false;
    ConnectionDetector cd;

    EditText login;
    EditText password;

    Student resurs = new Student();
    private static final String TAG = "myLogs";

    Button log;
    Button signUp;

    ProgressDialog progressDialog;

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



        progressDialog = new ProgressDialog(Login.this);
        progressDialog.setTitle("");

        cd = new ConnectionDetector(getApplicationContext());

        login = (EditText) findViewById(R.id.login);
        password = (EditText) findViewById(R.id.pass);

        login.setFilters(new InputFilter[]{filter});

        log = (Button) findViewById(Login);
        signUp = (Button) findViewById(sign);

        log.setOnClickListener(this);
        signUp.setOnClickListener(this);


        loadText();

    }

    @Override
    public void onClick(View v){
        switch (v.getId()) {
            case R.id.Login:
                proces();
```

```
                saveText();
                break;
            case R.id.sign:
                final Intent signIntent = new Intent(this, sign_up.class);
                startActivity(signIntent);
             break;
        }
    }


    void proces(){
        final Intent intent = new Intent(this, ShowProgress.class);
        String log = "";
        log = login.getText().toString();
        String pass = "";
        pass = password.getText().toString();

        String uml =
"https://moduleok.appspot.com/api/getScoresByPassword?login="+login.getText().toStrin
g()
                +"&password="+ password.getText().toString();



        //заповнення полів
        if (log.length() < 1 && pass.length() < 1) {
            message("Enter your login and password!");
        }else if (log != "" && pass .length() < 1){
            message("Enter your password!");
        }else if (log.length() < 1 && pass != ""){
            message("Enter your login!");
        }else

            //наявність інтернету
            if(isInternetPresent = cd.ConnectingToInternet() == false){
                message("Internet connection not found!");
            }else{


                DBHelper dbHelper = new DBHelper(this.getBaseContext());
                SQLiteDatabase db = dbHelper.getWritableDatabase();
                getnNamesArrayFromDB userType = new
getnNamesArrayFromDB(dbHelper,db);

                Integer test = userType.getUserType(log,pass);
                Log.d(SAVED_TEXT,test.toString() +
"++++++++++++++++++++++++++++++++++++++++++");

                if(userType.getUserType(log,pass) == 0){
                    boolean secces = false;

                    String json = "";

                    try {

                        json  = getJson(uml);////----------------------------------
------------------маєм джейсон


                    } catch (ExecutionException e) {
                        e.printStackTrace();
```

```
                    } catch (InterruptedException e) {
                        e.printStackTrace();
                    }

                    secces = parsSuccess(json);


                    //коректність даних
                    if (secces == false){
                        message("Incorrect login or password!");
                    }else {

                        intent.putExtra("parsText", json);
                        startActivity(intent);
                    }
                }else {

                    final Intent viewIntent = new Intent(this,
ViewStudentList.class);
                    viewIntent.putExtra("parsText", "pzs-41");
                    startActivity(viewIntent);
                }

            }
    }

    void message(String text){

        AlertDialog.Builder mess = new  AlertDialog.Builder(Login.this);
        mess.setMessage(text);
        mess .setCancelable(false);
        mess.setNegativeButton("ok!",
                new DialogInterface.OnClickListener() {
                    public void onClick(DialogInterface dialog, int id) {
                        dialog.cancel();
                        Log.d(TAG, "click");
                    }
                });
        mess.show();

    }


    Boolean parsSuccess(String json){
        Boolean out = false;
        try {
            JSONObject jsonObject = new JSONObject(json);
            resurs.setSuccess(jsonObject.getBoolean("success"));
            Log.d(TAG, resurs.getSuccess().toString());

            out = resurs.getSuccess();

        } catch (JSONException e) {
            e.printStackTrace();
        }

        return out;
    }


    String getJson(String uml)throws ExecutionException, InterruptedException {
```

```
        String text = new JSONTask().execute(uml).get();
        return text;
    }


    InputFilter filter = new InputFilter() {

        public CharSequence filter(CharSequence source, int start, int end,
                                   Spanned dest, int dstart, int dend) {
            for (int i = start; i < end; i++) {

                if (!Character.isLetterOrDigit(source.charAt(i))) {

                    Toast.makeText(getApplicationContext(), "Invalid Input",
Toast.LENGTH_SHORT).show();
                    return "";
                }
            }
            return null;
        }

    };

    void saveText() {
        sPref = getPreferences(MODE_PRIVATE);
        SharedPreferences.Editor ed = sPref.edit();
        ed.putString(SAVED_TEXT, login.getText().toString());
        ed.commit();
        Toast.makeText(this, "Text saved", Toast.LENGTH_SHORT).show();
    }
    void loadText() {
        sPref = getPreferences(MODE_PRIVATE);
        String savedText = sPref.getString(SAVED_TEXT, "");
        login.setText(savedText);
        Toast.makeText(this, "Text loaded", Toast.LENGTH_SHORT).show();
    }

}
```

Лістинг класу sign_up.java

```
public class sign_up extends AppCompatActivity implements View.OnClickListener {

    final String LOG_TAG = "myLogs";
    DBHelper dbHelper;
    final String myTag = "DocsUpload";

    EditText lname;
    EditText fname;
    EditText login;
    EditText password;
    EditText group;
    EditText mail;

    Switch type;

    EditText mather_name;
    EditText father_name;
    EditText mather_mail;
```

```
    EditText father_mail;

    Button submit;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        setContentView(R.layout.sign_up);

        lname = (EditText) findViewById(R.id.lname);
        fname = (EditText) findViewById(R.id.fname);
        login = (EditText) findViewById(R.id.Login);
        password = (EditText) findViewById(R.id.Pass);
        group = (EditText) findViewById(R.id.Group);
        mail = (EditText) findViewById(R.id.email);

        type = (Switch) findViewById(R.id.type);



        mather_name = (EditText) findViewById(R.id.Mather_Name);
        mather_mail = (EditText) findViewById(R.id.Mather_Mail);
        father_name = (EditText) findViewById(R.id.Father_Name);
        father_mail = (EditText) findViewById(R.id.Father_Mail);


        type.setOnCheckedChangeListener(new CompoundButton.OnCheckedChangeListener()
{
            @Override
            public void onCheckedChanged(CompoundButton buttonView, boolean
isChecked) {
                if (isChecked) {
                    mather_name.setEnabled(false);
                    mather_mail.setEnabled(false);
                    father_name.setEnabled(false);
                    father_mail.setEnabled(false);
                } else {
                    mather_name.setEnabled(true);
                    mather_mail.setEnabled(true);
                    father_name.setEnabled(true);
                    father_mail.setEnabled(true);
                }
            }
        });

        submit = (Button) findViewById(R.id.SEND);

        submit.setOnClickListener(this);
    }


    @Override
    public void onClick(View v) {
        ContentValues cv = new ContentValues();

        String Slname = lname.getText().toString();
        String Sfname = fname.getText().toString();
        String Slogin = login.getText().toString();
        String Spassword  = password.getText().toString();
```

```
        String Sgroup = group.getText().toString();
        String Smail = mail.getText().toString();

        Boolean Btype = type.isChecked();


        String Smather_name = mather_name.getText().toString();
        String Sfather_name = father_name.getText().toString();
        String Smather_mail = mather_mail.getText().toString();
        String Sfather_mail = father_mail.getText().toString();


        DBHelper dbHelper = new DBHelper(this.getBaseContext());
        SQLiteDatabase db = dbHelper.getWritableDatabase();


        switch (v.getId()) {
            case R.id.SEND:
                Log.d(LOG_TAG, "--- Insert in User: ---");


                cv.put("fname", Sfname);
                cv.put("lname", Slname);
                cv.put("login", Slogin);
                cv.put("password", Spassword);
                cv.put("agroup", Sgroup);
                cv.put("email", Smail);


                if(Btype == true){
                    cv.put("type", Btype);

                    cv.put("father_name", Sfather_name);
                    cv.put("mather_name", Smather_name);
                    cv.put("father_mail", Sfather_mail);
                    cv.put("mather_mail", Smather_mail);
                }


                long rowID = db.insert("User", null, cv);
                Log.d(LOG_TAG, "row inserted, ID = " + rowID);
                Toast.makeText(this, "Text saved", Toast.LENGTH_SHORT).show();
                break;
        dbHelper.close();
    }
}
```

Лістинг класу ShowProgress.java

```
public class ShowProgress extends AppCompatActivity {

    private static final String TAG = "myLogs";
    ListView listView1;
    Switch toggle;
    String semestr;
    List<Subjects> first;
    StringPars stringPars;
    String parsText;
    Integer userType;
```

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

        parsText = getIntent().getStringExtra("parsText");
        userType = getIntent().getIntExtra("UserType", 0);

        stringPars= new StringPars();

        toggle = (Switch) findViewById(R.id.switch1);
        toggle.setOnCheckedChangeListener(new
CompoundButton.OnCheckedChangeListener() {
            @Override
            public void onCheckedChanged(CompoundButton buttonView, boolean
isChecked) {
                if (isChecked) {
                    semestr = "secondSemester";
                    show(parsText,semestr);
                } else {
                    semestr = "firstSemester";
                    show(parsText, semestr);
                }
            }
        });

        if(toggle.isChecked()){
            semestr = "secondSemester";
        }
        else {
            semestr = "firstSemester";
            if(iff(parsText,"secondSemester") == true) {
                toggle.setEnabled(false);
                Context context = getApplicationContext();
                CharSequence text = "Second semester is empty!";
                int duration = Toast.LENGTH_SHORT;

                Toast toast = Toast.makeText(context, text, duration);
                toast.setGravity(Gravity.CENTER,0,0);
                toast.show();
            }
        }


        show(parsText,semestr);

        if(userType == 1){

            final Intent intent = new Intent(this, SendMail.class);

            final String name = getIntent().getStringExtra("Name");
            final String email = getIntent().getStringExtra("Email");
            final String matherName = getIntent().getStringExtra("MatherName");
            final String fatherName = getIntent().getStringExtra("FatherName");
            final String matherMail = getIntent().getStringExtra("MatherMail");
            final String fatherMail = getIntent().getStringExtra("FatherMail");

            listView1.setOnItemClickListener(new AdapterView.OnItemClickListener() {
                @Override
                public void onItemClick(AdapterView<?> parent, View view, int
```

```
position, long id) {
                    Integer itemPosition = position;

                    intent.putExtra("Name", name);
                    intent.putExtra("Email", email);
                    intent.putExtra("MatherName", matherName);
                    intent.putExtra("FatherName", fatherName);
                    intent.putExtra("MatherMail", matherMail);
                    intent.putExtra("FatherMail", fatherMail);

                    startActivity(intent);
                }
            });
        }

    }

    private void show(String parsText,String semestr){
        first = stringPars.stringPars(parsText, semestr);

        listView1 = (ListView) findViewById(R.id.listview1);
        YourAdapter adapter = new YourAdapter(this, first);
        listView1.setAdapter(adapter);
    }


    private boolean iff(String parsText,String semestr){
        if(stringPars.stringPars(parsText, semestr).isEmpty()) {
            return true;
        }
        else {
            return false;
        }
    }
}
```

Лістинг класу ViewStudentList.java

```
public class ViewStudentList extends AppCompatActivity implements
View.OnClickListener {
    private static final String TAG = "view Student Logs --";

    private String _group;

    Student resurs = new Student();
    ConnectionDetector cd;
    Boolean isInternetPresent = false;

    @Override
    protected void onCreate(Bundle savedInstanceState) {

        final Intent intent = new Intent(this, ShowProgress.class);

        super.onCreate(savedInstanceState);
        setContentView(R.layout.students);


        cd = new ConnectionDetector(getApplicationContext());
```

```
        _group = getIntent().getStringExtra("parsText");

        final ListView studentList = (ListView) findViewById(R.id.studentsList);

        DBHelper dbHelper = new DBHelper(this.getBaseContext());
        SQLiteDatabase db = dbHelper.getWritableDatabase();
        getnNamesArrayFromDB namesArrayFromDB = new
getnNamesArrayFromDB(dbHelper,db);

        final List<StudentDataList> students =
namesArrayFromDB.getNamesArray(_group);

        List<String> names = new ArrayList<>();

        for (StudentDataList sdl:students) {
            names.add(sdl.getName());
        }

        ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
                android.R.layout.simple_list_item_1, names);

        studentList.setAdapter(adapter);

        studentList.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> parent, View view, int position,
long id) {

                Integer itemPosition = position;

                StudentDataList sdl = students.get(position);
                String value = (String)studentList.getItemAtPosition(position);

                String log = sdl.getLog();
                String pass = sdl.getPass();

                Toast.makeText(ViewStudentList.this,itemPosition.toString() + "" +
log +"" + pass, Toast.LENGTH_SHORT).show();


                String uml =
"https://moduleok.appspot.com/api/getScoresByPassword?"+log+"&"+pass;

                Log.d(TAG, uml);


                if(isInternetPresent = cd.ConnectingToInternet() == false){
                    message("Internet connection not found!");
                }else{

                    boolean secces = false;

                    String json = "";

                    try {

                        json  = getJson(uml);////---------- маєм джейсон

                        Log.d(TAG, json);
```

```
                } catch (ExecutionException e) {
                    e.printStackTrace();
                } catch (InterruptedException e) {
                    e.printStackTrace();
                }

                secces = parsSuccess(json);

                //коректність даних
                if (secces == false){
                    message("Incorrect login or password!");
                }else {

                    intent.putExtra("UserType", 1);
                    intent.putExtra("parsText", json);


                    intent.putExtra("Name", value);
                    intent.putExtra("Email", sdl.getMail());
                    intent.putExtra("MatherName", sdl.getMatherName());
                    intent.putExtra("FatherName", sdl.getFatherName());
                    intent.putExtra("MatherMail", sdl.getMatherMail());
                    intent.putExtra("FatherMail", sdl.getFatherMail());

                    startActivity(intent);
                }
            }
        }
    });

}

@Override
public void onClick(View v) {

}


void message(String text){

    AlertDialog.Builder mess = new  AlertDialog.Builder(ViewStudentList.this);
    mess.setMessage(text);
    mess .setCancelable(false);
    mess.setNegativeButton("ok!",
            new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int id) {
                    dialog.cancel();
                    //Log.d(TAG, "click");

                }
            });
    mess.show();

}

String getJson(String uml)throws ExecutionException, InterruptedException {
    String text = new JSONTask().execute(uml).get();
    return text;
}

Boolean parsSuccess(String json){
```

```
        Boolean out = false;
        try {
            JSONObject jsonObject = new JSONObject(json);
            resurs.setSuccess(jsonObject.getBoolean("success"));
            //Log.d(TAG, resurs.getSuccess().toString());

            out = resurs.getSuccess();

        } catch (JSONException e) {
            e.printStackTrace();
        }

        return out;
    }
}
```

Лістинг класу SendMail.java

```
public class SendMail extends AppCompatActivity implements View.OnClickListener {

    TextView bossText;
    EditText message;
    CheckBox student;
    CheckBox mather;
    CheckBox father;
    Button send;

    String name;
    String email;
    String matherName;
    String fatherName;
    String matherMail;
    String fatherMail;

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

        bossText = (TextView) findViewById(R.id.bossText);
        message = (EditText) findViewById(R.id.messageBox);
        student = (CheckBox) findViewById(R.id.student);
        father = (CheckBox) findViewById(R.id.father);
        mather = (CheckBox) findViewById(R.id.mather);
        send = (Button) findViewById(R.id.send);



        name = getIntent().getStringExtra("Name");
        email = getIntent().getStringExtra("Email");
        matherName = getIntent().getStringExtra("MatherName");
        fatherName = getIntent().getStringExtra("FatherName");
        matherMail = getIntent().getStringExtra("MatherMail");
        fatherMail = getIntent().getStringExtra("FatherMail");

        bossText.setText(name);
        student.setText(name);
        father.setText(fatherName);
        mather.setText(matherName);
```

```
        send.setOnClickListener(this);
    }

    @Override
    public void onClick(View v) {
        sendEmail();

        message.setText("");
    }
    protected void sendEmail() {

        String[] recipients = getRecepientList();



        Intent email = new Intent(Intent.ACTION_SEND, Uri.parse("mailto:"));
        email.setType("message/rfc822");

        email.putExtra(Intent.EXTRA_EMAIL, recipients);
        email.putExtra(Intent.EXTRA_SUBJECT, "Успішність");
        email.putExtra(Intent.EXTRA_TEXT, message.getText().toString());

        try {
            // the user can choose the email client
            startActivity(Intent.createChooser(email, "Choose an email client
from..."));

        } catch (android.content.ActivityNotFoundException ex) {
            Toast.makeText(SendMail.this, "No email client installed.",
                    Toast.LENGTH_LONG).show();
        }
    }

    private String[] getRecepientList(){
        String[] recipients = new String[3];
        if(student.isChecked()){recipients[0] = email;}
        if(father.isChecked()){recipients[1] = matherMail;}
        if(mather.isChecked()){recipients[2] = fatherMail;}
        return recipients;
    }
}
```

Лістинг класу DBHelper.java

```
class DBHelper extends SQLiteOpenHelper {

    final String LOG_TAG = "myLogs";
    public DBHelper(Context context) {
        super(context, "myDB1", null, 1);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        Log.d(LOG_TAG, "--- onCreate database ---");

        db.execSQL("create table User ("
                + "id integer primary key autoincrement,"
                + "fname text,"
                + "lname text,"
```

```
                + "login text,"
                + "password text,"
                + "agroup text,"
                + "email text,"
                + "type INTEGER DEFAULT 0,"
                + "father_name text,"
                + "father_mail text,"
                + "mather_name text,"
                + "mather_mail text"
                + ");");
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {}
}
```

Лістинг класу getnNamesArrayFromDB.java

```
public class getnNamesArrayFromDB {

    private static final String TAG = "myLogs";

    private DBHelper _dbHelper;
    private SQLiteDatabase _db;

    public getnNamesArrayFromDB(DBHelper dbHelper,SQLiteDatabase db){
        _dbHelper = dbHelper;
        _db = db;
    }

    public List<StudentDataList> getNamesArray(String group) {
        List<StudentDataList> users = new ArrayList<>();

        Cursor c = null;

        String[] columns = new String[]{"lname",
"fname","login","password","email","mather_name",
                "father_name","mather_mail","father_mail"};
        String having = "agroup = '" + group + "' and type = '0';";
        c = _db.query("User", columns, having, null, null, null,"lname");

        if (c.moveToFirst()) {

            int nameColIndex = c.getColumnIndex("fname");
            int lnameColIndex = c.getColumnIndex("lname");
            int passwordColIndex = c.getColumnIndex("password");
            int loginColIndex = c.getColumnIndex("login");

            int emailColIndex = c.getColumnIndex("email");
            int matherNameColIndex = c.getColumnIndex("mather_name");
            int fatherNameColIndex = c.getColumnIndex("father_name");
            int matherMailColIndex = c.getColumnIndex("mather_mail");
            int fatherMailColIndex = c.getColumnIndex("father_mail");

            do {
                StudentDataList studentDataList = new StudentDataList();
                String nameStr = c.getString(nameColIndex) +
                                 " " + c.getString(lnameColIndex);
                String logStr = "login=" + c.getString(loginColIndex);
```

```
                String pasStr = "password=" + c.getString(passwordColIndex);

                String matherNam = c.getString(matherNameColIndex);
                String fatherNam = c.getString(fatherNameColIndex);
                String matherMail = c.getString(matherMailColIndex);
                String fatherMail = c.getString(fatherMailColIndex);
                String email = c.getString(emailColIndex);

                Log.d(LOG_TAG, nameStr);

                studentDataList.setName(nameStr);
                studentDataList.setLog(logStr);
                studentDataList.setPass(pasStr);

                studentDataList.setFatherMail(fatherMail);
                studentDataList.setMatherMail(matherMail);
                studentDataList.setMatherName(matherNam);
                studentDataList.setFatherName(fatherNam);

                studentDataList.setMail(email);

                studentDataList.setPass(pasStr);


                users.add(studentDataList);

            } while (c.moveToNext());
        } else{
            StudentDataList studentDataList = new StudentDataList();
            studentDataList.setName("0 rows");
            studentDataList.setLog(null);
            studentDataList.setPass(null);
            users.add(studentDataList);
            Log.d(LOG_TAG, "0 rows");
        }
        c.close();
        return users;
    }

    public Integer getUserType(String login, String password){

        Integer res = null;

        Cursor c = null;

        String[] columns = new String[]{"type"};
        String having = "login = '" + login +  "' and password = '" + password +
"';";
        c = _db.query("User", columns, having, null, null, null, null);



        if (c.moveToFirst()) {
            if (c.moveToFirst()) {

                int idColIndex = c.getColumnIndex("type");

                do {
                    Log.d(LOG_TAG,
                            "type = " + c.getInt(idColIndex) +"----------------------
```

```
--------");

                    res =  c.getInt(idColIndex);
                } while (c.moveToNext());
            }
        }else {
            Log.d(LOG_TAG, "0 rows");
            res = 0;
        }

        c.close();

        return res;
    }


    public void readStrFromDB(){
        Log.d(LOG_TAG, "--- Rows in mytable: ---");
        Cursor c = _db.query("User", null, null, null, null, null, null);

        if (c.moveToFirst()) {

            int idColIndex = c.getColumnIndex("id");
            int nameColIndex = c.getColumnIndex("type");
            int emailColIndex = c.getColumnIndex("agroup");

            do {
                Log.d(LOG_TAG,
                        "ID = " + c.getInt(idColIndex) +
                                ", name = " + c.getString(nameColIndex) +
                                ", agroup = " + c.getString(emailColIndex));

            } while (c.moveToNext());
        } else
            Log.d(LOG_TAG, "0 rows");
        c.close();
    }

    final String LOG_TAG = "myLogs";
}
```