

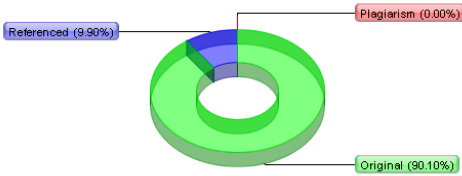
Детектор Плагиата v. 2215 - Отчёт оригинальности: 13.06.2024 10:12:25

Проанализированный документ: Diplom_Сумбулов (1).doc Лицензия: ВОЛОДИМИР МАТІЄВСЬКИЙ

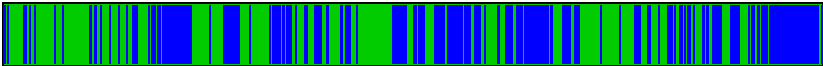
- Тип поиска: Поиск переписанного
- Язык: Uk
- Тип проверки: Интернет
- ТЕЕ и кодировка: ifilter

Детальный анализ тела документа:

Диаграмма соотношения частей:



Граф распределения зон:



Источники плагиата: 0

Детали обработанных ресурсов: 113 - OK / 4 - Ошибка

Важные замечания:

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

Античит-отчет UACE:

<div><div>1. Статус: Анализатор Включен Нормализатор Включен сходство символов установлено на 100%</div><div>2. Обнаруженный процент загрязнения UniCode: 44,3% с лимитом: 4%</div><div>3. Процент нераспознанных символов после нормализации: 25,1%</div><div>4. Все подозрительные символы будут отмечены фиолетовым цветом: Abcd...</div><div>5. Найдены невидимые символы: 0</div></div> <div>Рекомендации по оценке: Особое внимание следует уделить анализу этого отчета! Предполагается, что этот документ содержит значительное количество символов, чуждых языку документа. Это прямое указание на то, что автор документа использовал специальное программное обеспечение\онлайн-веб-сервис, чтобы эффективно скрыть текст в попытке избежать обнаружения потенциального плагиата. Настоятельно рекомендуется передать это дело на более высокий уровень! В случае сомнений обращайтесь в службу поддержки Детектора плагиата!</div> <div>Алфавитная статистика и анализ символов:</div>	<div>Активные ссылки (URL-адреса, извлеченные из документа): URL не найдены</div> <div>Исключённые ресурсы: URL не найдены</div> <div>Включённые ресурсы: URL не найдены</div>
--	--

Детальный анализ документа:	
Міністерство освіти і науки України	
ДЗ	
Цитирования: 0,03%	id: 1
"луганський Національний університет імені тараса шевченка"	
Навчально-науковий інститут математики та інформаційних технологій (назва факультету, інституту)Кафедра інформаційних технологій та систем(назва кафедри)	
Пояснювальна записка до дипломного проєкту (роботи)	
БАКАЛАВРА (освітньо-кваліфікаційний рівень)	
на тему: РОЗРОБКА ANDROID-ДОДАТКА ДЛЯ ОБЛІКУ ВІДВІДУВАННЯ ЗАНЯТЬ СТУДЕНТАМИ ЗАКЛАДІВ ВИЩОЇ ОСВІТИ	
Виконав: студент 4 курсу	
напряму підготовки (спеціальності)	
121	
Цитирования: 0,02%	id: 2
"Інженерія програмного забезпечення" _	
(шифр і назва напряму підготовки, спеціальності)	
Бабенко В. В.	
(прізвище та ініціали)	
Керівник __ Донченко В. Ю.	
(прізвище та ініціали)	
Рецензент ____ Козуб Ю. Г.	
(прізвище та ініціали)	
Полтава - 2024 року	
ЗМІСТ	
ВСТУПЗ	
РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ	
Цитирования: 0,02%	id: 3
"ОБЛІК ВІДВІДУВАНОСТІ СТУДЕНТІВ"	
5	
1.1. Коротка характеристика об'єкту управління	
Цитирования: 0,02%	id: 4
"закладів вищої освіти"	
5	
1.2. Опис предметної області	
Цитирования: 0,02%	id: 5
"Облік відвідуваності студентів"	
10	
1.3. Огляд існуючих аналогів15	
1.4. Специфікація вимог до Android-додатку для обліку відвідуваності занять студентами26	
РОЗДІЛ 2. ПРОЄКТУВАННЯ ANDROID-ДОДАТКА ДЛЯ ОБЛІКУ ВІДВІДУВАННЯ ЗАНЯТЬ СТУДЕНТАМИ ЗАКЛАДІВ ВИЩОЇ ОСВІТИ40	
2.1. Розробка архітектури програмної системи40	
2.2. Проектування структури бази даних49	
РОЗДІЛ ІІІ. ПРОГРАМНА РЕАЛІЗАЦІЯ ANDROID-ДОДАТКА ДЛЯ ОБЛІКУ ВІДВІДУВАННЯ ЗАНЯТЬ СТУДЕНТАМИ ЗАКЛАДІВ ВИЩОЇ ОСВІТИ55	
3.1. Програмна реалізація проєкту55	
3.2. Програмна реалізація бази даних62	
3.3.Тестування64	

3.4. Розгортання програмного продукту	66
3.5. Інструкція користувача	67
ВИСНОВКИ	72
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	73
ДОДАТОК А	75
ДОДАТОК Б	77
ДОДАТОК В	79
ДОДАТОК Г	81
ВСТУП	
<p>У сучасній системі освіти постійно відбуваються зміни, що зумовлені процесами економічного, політичного чи соціального характеру. Змінюються, як пріоритети в системі освіти і у системі управління освітою, так і зміст та структура самого освітнього процесу. Крім цього, у галузі освіти, досить часто, випробовуються й новітні технології.</p> <p>Зокрема, істотні зміни в інформаційній сфері діяльності людини призводять до суттєвого зниження ефективності застарілих підходів до контролю відвідуваності студентами занять. Саме тому, сьогодні є доцільною розробка спеціальної системи, яка дозволить автоматизувати процес обліку відвідування занять студентами та зробити його швидшим, зручнішим та надійнішим.</p> <p>Проголи, запізнення, відсутність учнів на заняттях - це проблема кожного навчального закладу. Такі речі роблять процес навчання непередбачуваним, що надалі позначається на загальних показниках успішності. Автоматизація процесу контролю відвідуваності, може вирішити ці проблеми надавши гнучкі засоби для моніторингу за пропусками занять студентами. Така система, дозволить швидко виявляти проблеми з безпричинними пропусками занять студентом та надасть певні засоби для запобігання прогулам. Крім того, такий засіб, буде дозволяти студенту стежити за власною відвідуваністю, дозволяючи пересвідчитися в тому, що він повною мірою використовує можливості навчання, які доступні для нього.</p> <p>Найкращим рішенням для автоматизації процесу контролю відвідуваності є система, яка інтегрує у собі веб-орієнтоване програмне рішення та програмні рішення для мобільних платформ. На сьогоднішній день уже існує достатньо велика кількість веб-орієнтованих продуктів для контролю відвідуваності, фактично такі системи вже є у більшості навчальних закладів. Тому, основним предметом розробки дипломної роботи є програмний продукт для мобільних платформ, який надаватиме зручні засоби для контролю відвідуваності та інтегруватиметься із аналогічними веб-орієнтованими системами.</p> <p>Об'єкт дослідження - засоби контролю та обліку відвідування занять у закладах вищої освіти.</p> <p>Предмет дослідження - шляхи автоматизації обліку та контролю відвідування занять студентами за допомогою інформаційних технологій.</p> <p>Мета роботи - розробка Android-додатка для обліку відвідування занять студентами з можливістю інтеграції із веб-орієтованим аналогом.</p> <p>Методи дослідження: техніко-економічні з використанням комп'ютерних технологій, технічний аналіз, методи моделювання інформаційних процесів.</p> <p>Для досягнення поставленої мети необхідно вирішити наступні завдання:</p> <p>проаналізувати предметну область</p> <p>Цитування: 0,02% id: 6</p> <p>"Облік відвідуваності студентів"</p> <p>;</p> <p>провести моделювання та аналіз програмного забезпечення для архітектури програмної системи та структури бази даних;</p> <p>розробити і реалізувати Android-додаток для обліку відвідування занять студентами закладів вищої освіти;</p> <p>Практичною цінністю роботи є розроблений Android-додаток для обліку відвідування занять студентами закладів вищої освіти.</p> <p>В першому розділі проаналізовано предметну область</p> <p>Цитування: 0,02% id: 7</p> <p>"Облік відвідуваності студентів".</p> <p>Проведено огляд існуючих аналогів. Розглянуто специфікацію вимог до Android-додатка.</p> <p>В другому розділі виконано аналіз процесу розробки Android-додатку для обліку відвідування занять студентами закладів вищої освіти. Проведено моделювання та описано архітектуру розробленого додатку.</p> <p>У третьому розділі аргументується вибір середовища розробки додатку і мови програмування. Розглянуто програмну реалізацію Android-додатка для обліку відвідування занять студентами закладів вищої освіти. Проведено тестування програми та розроблено інструкцію користувача.</p> <p>РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ</p> <p>Цитування: 0,02% id: 8</p> <p>"ОБЛІК ВІДВІДУВАНОСТІ СТУДЕНТІВ"</p>	

1.1. Коротка характеристика об'єкту управління

Цитирования: 0,02%	id: 9
---------------------------	--------------

"закладів вищої освіти"

Заклад вищої освіти (далі ЗВО) - освітньо-науковий заклад, що заснований та діє у відповідності із освітнім законодавством, маючи один із чотирьох ступенів акредитації, реалізує освітньо-професійні програми, забезпечує виховання, навчання та професійну підготовку осіб відповідно до їх покликання.

На сьогодні фактично кожний навчальний заклад потребує систему для обліку відвідуваності своїх студентів.

Управління ЗВО здійснюють за допомогою багатьох відділів та підрозділів. Усі вони певним чином пов'язані між собою. У залежності від типу закладу та його внутрішнього устрою ці підрозділи можуть називатися по-різному, але їх сутність є досить схожою. Далі розглянемо основні структурні підрозділи, які мають зв'язок із предметною областю курсової роботи.

Колегіальне керівництво закладом, тобто вирішення ключових питань щодо його діяльності, здійснюється найвищим адміністративним органом (переважно ректоратом). Фактично, цей орган приймає усі основні рішення щодо будь-яких питань пов'язаних з роботою ЗВО. Таким чином, він здійснює керування усіма структурними підрозділами вищого навчального закладу та забезпечує їх ефективну роботу.

Наступним, у схемі управління навчальними процесами є

Цитирования: 0,01%	id: 10
---------------------------	---------------

"навчальний підрозділ".

Навчальний підрозділ також складається із організаційних і навчально-наукових структурних підрозділів (наприклад факультетів). Кожний із них має відповідний освітній напрямок, а їхнім основним завданням є підготовка студентів по обраних спеціальностях. Кожний із організаційних і навчально-наукових структурних підрозділів має власний керуючий орган(наприклад деканат) та власні навчальні та наукові-дослідні підрозділи(кафедри), основною метою, яких є проведення навчально-виховної та методичної діяльності. Останні із згаданих органів та підрозділів фактично є тими, які безпосередньо керують навчальним процесом та забезпечують його виконання.

На рисунку 1.1 зображено загальну схему структурної організації управління ЗВО, яка забезпечує контроль навчального процесу, основні елементи схеми були описані вище.

Рис. 1.1. Схема організаційної структури управління

Питання відвідуваності студентами занять завжди було проблематичним для будь-якого ЗВО. У сучасному світі є безліч найрізноманітніших причин та факторів, які можуть спонукати студентів до пропуску занять. Цікаво, що проведені дослідження свідчать про те, що причини невідвідуваності можуть

змінюватися в залежності від того на якому курсі навчається студент. Наприклад на першому курсі студентам завжди важко звикнути до нової форми навчання й вони часто пропускають заняття через проблеми з адаптацією, на другому - причиною пропусків можуть проблемні стосунки з однокурсниками, на третьому у студента може знизитися мотивація (думки про неправильно обрану професію) або з'явитися самовпевненість у власних знаннях, на четвертому - проблеми з відвідуваністю зазвичай пов'язані з намаганням поєднати навчання та роботу.

Отже, у студентів можуть бути різні причини пропуску занять, крім того, ці причини можуть змінюватися. Незмінним залишається одне - потреба у обліку відвідуваності занять. Правильний підхід до цього процесу дозволить зробити його максимально швидким та надійним. Що у свою чергу дозволить не просто виявляти студентів, які повинні бути виключені через систематичні пропуски та заборгованості, а вчасно відстежити проблеми з відвідуваністю у студента й запобігти подальшим пропускам.

Розглянемо процес обліку відвідуваності студентами занять у вищих навчальних закладах детальніше. Староста групи отримує журнал обліку та особисто вносить в нього дані про відсутність студентів групи для кожного заняття. Вигляд журналу обліку відображено у таблиці 1.1.

Таблиця 1.1.

Вигляд журналу обліку

Дні тижня	Понеділок	Вівторок	Середа	Дата заняття	Прізвище та ініціали	Підпис
Викладача						

Староста групи зобов'язується передавати відомості про відвідуваність з певною регулярністю, переважно щоденно. Цей процес може відбуватися одним із наступних шляхів:

староста передає журнал у відповідний керуючий орган(далі деканат), який має свій журнал і переносить в нього потрібні дані;

староста вносить дані в спеціальну систему(базу даних) самостійно або це роблять відповідні працівники деканату.

На рисунку 1.2 зображено варіант коли староста сам вносить дані у електронний журнал відвідуваності.

Рис. 1.2. Вигляд електронного журналу

Далі відповідно до використовуваного методу проводять облік та визначають потрібні статистичні дані. Наприклад, обраховують кількість пропусків без поважних причин окремо для кожного студента. Таким чином, студенти, які перевищили встановлений навчальним закладом ліміт пропусків, потрапляють у списки на виключення. Сутність процесу обліку

відвідуваності студентами занять описує діаграма [IDEF0](#), яка зображена на рисунку 1.3.

Рис. 1.3. Декомпована діаграма [IDEF0](#)

Процес обліку відвідуваності за такою схемою є складним та містить велику кількість недоліків. На будь-якому з вищезгаданих етапів може трапитися помилка, особливо, якщо вся інформація обробляється вручну. Деякі старости можуть не завжди подавати точні дані, бажаючи

 Цитування: 0,01%	id: 11
"прикрити"	

товаришів. Фактично, кожний із викладачів веде власний журнал обліку відвідуваності, що є повторенням процесу. Крім того, процес отримання та аналізу статистичних даних про відвідуваність за описаною схемою, є досить повільним і, як наслідок, процес повідомлення про проблеми з відвідуваністю є відсутнім, аж до того моменту, поки студент не перевищить ліміт пропусків.

Після розгляду основної проблеми, очевидно, що для її розв'язку, для певних класів користувачів, необхідно буде використовувати мобільну платформу. Проте найкраще буде використовувати мобільну платформу інтегровану із веб-орієнтованою програмною системою. У більшості навчальних закладів уже є веб-сайти для обліку відвідуваності студентів. Проте, вони здатні лише на часткове вирішення проблеми. Вони є хорошим рішенням для працівників деканату, які отримують автоматизований засіб для швидкого керування, який дозволяє миттєво обраховувати статистичні показники та працювати з великими об'ємами даних. Однак веб-сайт аж ніяк не вирішує проблему для інших користувачів системи. Зрозуміло, що практично не можливим є використання веб-орієнтованого продукту студентами та викладачами під час навчального процесу. Це демонструє, що веб-орієнтоване вирішення проблеми має велику кількість недоліків. Хорошим прикладом є те, що викладач не зможе вводити дані у систему безпосередньо під час заняття. При інтеграції з мобільною платформою майже усі недоліки веб-орієнтованої системи нівелюватимуться перевагами системи на мобільній платформі. Щодо вибору мобільної платформи, очевидно, що найкраще вибрати найпопулярнішу, якою на сьогодні є [Android](#).

1.2. Опис предметної області

 Цитування: 0,02%	id: 12
"Облік відвідуваності студентів"	

Для належного функціонування [Android](#)-додатку для обліку відвідуваності студентів потрібно забезпечити виконання таких основних бізнес-процесів (рис. 1.4):

- процес збору даних про відвідуваність;
- процес обробки та аналізу даних;
- процес перегляду статистичних даних;
- процес повідомлення про проблеми з відвідуваністю.

Рис. 1.4. Діаграма бізнес-процесів розроблюваного програмного продукту

Розглянемо детальніше кожен з вище представлених бізнес-процесів. На рисунку 1.5 бачимо діаграму функцій, які потрібні для забезпечення виконання процесу збору даних.

Рис. 1.5. Діаграма функцій процесу збору даних про відвідуваність

Першим етапом для обліку відвідуваності студентів є збір даних про відвідуваність. Цей процес є дуже важливим, оскільки від нього залежатиме якість та достовірність результатів всіх інших бізнес-процесів. Тому, процес

має бути організованим таким чином, щоб забезпечити максимальну точність при мінімальній кількості етапів.

Характеристику бізнес-процесу збору даних наведено в таблиці 1.2.

Таблиця 1.2.

Характеристика бізнес-процесу збору даних про відвідуваність

Назва характеристики	Значення характеристики	Ім'я бізнес-процесу	Збір даних про відвідуваність
Основні учасники	Староста, викладач	Вхідна подія	Проведення заняття
Вихідні документи	Список студентів	Вихідна подія	Проведене заняття та зібрані дані про присутність студентів
Вихідні документи	Список відсутніх студентів	Клієнт бізнес-процесу	Обробка та аналіз даних

Рис. 1.6. Діаграма функцій процесу обробки та аналізу даних

На етапі обробки та аналізу даних виконується певне опрацювання даних, яке забезпечує можливість проведення різноманітних аналітичних розрахунків

та збереження їх результатів. Фактично, саме цей бізнес-процес відповідальний за створення різноманітних статистичних показників щодо відвідуваності.

Характеристику бізнес-процесу обробки та аналізу даних наведено в таблиці 1.3.

Таблиця 1.3.

Характеристика бізнес-процесу обробки та аналізу даних

Назва характеристики	Значення характеристики	Ім'я бізнес-процесу	Обробка та аналіз даних
Основні учасники	Працівник деканату, електрона система	Вхідна подія	Зібрані дані про відвідуваність передані на подальше опрацювання
Вхідні документи	Журнали із даними про відвідуваність	Вихідна подія	Створено статистичні дані про відвідуваність
Вихідні документи	Журнали із даними про відвідуваність	Клієнт бізнес-процесу	Обробка та аналіз даних

документиДокументи із статистичними показникамиКлієнт бізнес-процесуПерегляд статистичних даних

На рисунку 1.7 бачимо діаграму функцій процесу перегляду статистичних даних.

Рис. 1.7. Діаграма функцій процесу перегляду статистичних даних

На етапі обробки та аналізу даних про відвідуваність отримують статистичні показники, які зберігають у певному вигляді. Бізнес-процес перегляду статистичних даних дозволяє використовувати ці дані для того щоб отримати потрібну інформацію щодо статистики відвідування занять студентами.

Характеристику бізнес-процесу перегляду статистичних даних наведено в таблиці 1.4.

Таблиця 1.4.

Характеристика бізнес-процесу перегляду статистичних даних

Назва характеристики	Значення характеристики	Ім'я бізнес-процесу	Перегляд статистичних даних
Основні учасники	Викладач, працівник деканату	Вхідна подія	Запит про статистичні показники
Вхідні документи	Документи із статистичними показниками	Вихідна подія	Отримано статистичні показники
Вихідні документи	Клієнт бізнес-процесу	Повідомлення про проблеми з відвідуваністю	

На рисунку 1.8 бачимо діаграму функцій процесу повідомлення про проблеми з відвідуваністю.

Рис. 1.8. Діаграма функцій процесу повідомлення про проблеми з відвідуваністю

Для виявлення проблем з відвідуваністю переглядаються статистичні показники. Таким чином, якщо в якогось із студентів є значні порушення допустимої норми пропущених занять він потрапляє у список студентів, яких можуть виключити та має бути проінформованим про це.

Характеристику бізнес-процесу процесу повідомлення про проблеми з відвідуваністю наведено в таблиці 1.5.

Таблиця 1.5.

Характеристика бізнес-процесу процесу повідомлення про проблеми з відвідуваністю

Назва характеристики	Значення характеристики	Ім'я бізнес-процесу	Повідомлення про проблеми з відвідуваністю
Основні учасники	Працівник деканату	Вхідна подія	Запит на виявлення проблем з відвідуваністю
Вхідні документи	Документи із статистичними показниками	Вихідна подія	Проблеми з відвідуваністю виявлені та опрацьовані
Вихідні документи	Списки студентів з проблемами з відвідуваністю	Клієнт бізнес-процесу	

1.3. Огляд існуючих аналогів

На сучасному ІТ-ринку уже існує досить широкий спектр програмних продуктів, які дозволяють частково автоматизувати процес обліку відвідуваності у навчальному закладі. Проте, потрібно відзначити, що кожен програмний продукт має свої методи для вирішення проблеми обліку відвідуваності і є рішенням для конкретного навчального закладу. Незалежно від платформи, кожна система для моніторингу відвідуваності може представляти унікальні способи реалізації функціоналу, які можливо адаптувати під будь-яку платформу. Саме тому, надзвичайно важливо провести аналіз ринку існуючих програмних засобів не лише для мобільних платформ.

У межах дипломної роботи було розглянуто широкий асортимент програмних продуктів для розв'язання проблеми обліку відвідуваності студентів. При цьому, для детального огляду та аналізу, було обрано найпопулярніші програмні системи, такі як:

» Цитування: **0,01%** id: **13**

"BioTime EDU",

» Цитування: **0,01%** id: **14**

"MyAttendanceTracker",

(режим доступу: <https://www.myattendancechecker.com/>),

» Цитування: **0,01%** id: **15**

"Attendance",

» Цитування: **0,01%** id: **16**

"Attendance Tracker",

Варто зазначити, що кожен із цих засобів використовує унікальні методи та технології для реалізації основного функціоналу.

Розглянемо детальніше систему

» Цитування: **0,01%** id: **17**

"BioTime EDU".

Ця програмна система розроблена британською компанією

» Цитування: **0,01%** id: **18**

"3 Biometrics".

Вона майже цілком забезпечує автоматизацію процесу моніторингу за відвідуваністю у навчальному закладі. Програма має інтуїтивно зрозумілий інтерфейс, але для роботи з деякими функціями персоналу потрібно пройти підготовку. Система

» Цитування: **0,01%** id: **19**

"BioTime EDU"

передбачає встановлення біометричних терміналів на вході у кожну аудиторію та наявність засобів для комунікації з центральними серверами. Таким чином, кожний студент у якого має відбутися заняття повинний отримати доступ при вході у аудиторію через термінал, приставивши палець до сканера і зробити теж саме після завершення заняття. Приклад біометричного сканеру та процес ідентифікації студента зображено на рисунку

1.9.

Рис. 1.9. Процес ідентифікації студента при вході (виході) в аудиторію

Таким чином, система

 Цитування: 0,01%	id: 20
"BioTime EDU"	

повністю звільняє викладача від процесу перевірки присутності студентів та автоматично формує списки присутніх, відзначаючи час коли студент увійшов та вийшов. Також, система дозволяє складати розклади занять (рис. 1.10).

Рис. 1.10. Вікно для побудови розкладів занять

Крім того, програмна система

 Цитування: 0,01%	id: 21
"BioTime EDU"	

реалізує такі функції: додавання, перегляд та редагування персональних даних кожного студента, перегляд різноманітних звітів (звіт по відвідуваності, звіт по присутності в аудиторіях, звіт по студентах які перевищили ліміт пропусків та ін.), контроль доступу в аудиторії студентів та персоналу навчального закладу. Важливою особливістю даного продукту є те, що після досягнення певним студентом встановленого ліміту пропусків, він отримає на свій [e-mail](#) автоматично згенероване попередження про можливість того, що він потрапить у списки студентів на виключення.

На рисунку 1.11 проілюстровано загальний вигляд вікна для перегляду персональних даних студента та вікна програмної системи для їх редагування.

Рис. 1.11. Вікна для перегляду та редагування персональних даних студента

Далі розглянемо детальніше веб-орієнтовану програмну систему

 Цитування: 0,01%	id: 22
"MyAttendanceTracker"	

(режим доступу: <https://www.myattendancetracker.com/>). Веб-сайт було створено, як персональний помічник викладача для обліку відвідуваності його занять студентами. Однак, з часом програмна система доопрацьовувалася та удосконалювалася, що в свою чергу забезпечило значне розширення її функціоналу, який включає досить цікаві особливості. Зокрема, інтерфейс системи є досить зручним, а також він максимально оптимізований для запуску на мобільних платформах. При першому використанні програмного засобу, автоматично відкривається помічник, який допомагає розпочати роботу та ознайомитися з веб-сайтом (рис.1.12), що безумовно є перевагою цієї програмної системи. Оскільки, продукт розроблявся, як персональний помічник, викладач сам повинен формувати списки студентів, розклади занять та вносити персональні дані студентів.

Рис. 1.12. Помічник для початку роботи та знайомства з веб-сайтом

Цікавою особливістю системи є те, що він передбачає заповнення сторінки персональної інформації батьків студента (рис.1.13). Таким чином, викладач зможе надсилати скаргу на відвідуваність чи погану успішність безпосередньо батькам студента.

Рис. 1.13. Сторінка для введення персональної інформації батьків студента

Крім того, система, також, дозволяє переглядати різноманітні звіти (рис.1.14).

Рис. 1.14. Сторінка для вибору типу звіту



При цьому, продукт надає користувачу унікальну можливість - керувати структурою звітів та створювати нові види звітів (рис.1.15). Ще однією особливістю системи є те, що викладач сам може вирішувати, яким користувачам надати доступ до звітів про відвідуваність.

Рис. 1.15. Сторінка для створення нового виду звіту

На рисунку 1.16 зображено основну сторінку веб-орієнтованої програмної системи для введення даних про відвідуваність.

Рис. 1.16. Сторінка для введення даних про відвідуваність

Далі розглянемо систему для моніторингу відвідуваності у навчальному закладі

 Цитування: 0,01%	id: 23
"Attendance"	
 Цитування: 0,01%	id: 24
"Attendance"	








- це мобільний додаток для платформи [Android](#), який позиціонує себе як систему для викладачів, які хочуть мати можливість вести облік присутності своїх студентів використовуючи телефони. Інтерфейс додатку є досить простим та функціональним. Для того, щоб допомогти користувачу швидко навчитися користуватися системою, передбачено підказки, які вбудовані безпосередньо у інтерфейс. Додаток орієнтований на одного користувача, який сам вноситиме дані про студентів та налаштовуватиме свій розклад.













У головному вікні додатку відображається перелік занять для поточної дати, що проілюстровано на рисунку 1.17.

Рис. 1.17 Основне вікно додатку

Після вибору конкретного заняття, користувачу відображається список студентів, які повинні бути на ньому присутніми. Для кожного студента викладач може обрати відповідний стан із контекстного меню, що проілюстровано на рисунку 1.18. Після цього список оновиться і буде відображати усі введені зміни. Вибір стану присутності для кожного студента із контекстного меню є не надто вдалим рішенням, оскільки при великій кількості студентів такий підхід забиратиме у викладача досить велику кількість часу. Крім

того, це не зручно для викладача тим, що вимагатиме від нього повторення великої кількості однотипних дій.	
Рис. 1.18. Вікно вибору стану присутності	
Далі програма автоматично зберігає введену інформацію, щоб користувач мав змогу переглядати списки присутніх на кожному занятті для певного предмету (рис.1.19).	
Рис. 1.19 - Вікно зі списком занять з обраного предмету	
Однією із переваг системи, безумовно є можливість імпорту списків студентів із текстового документу із розширенням .txt. Для цього потрібно буде створити документ за наданим форматом(рис.1.20). Після чого вибрати функцію імпорту списку із документу та вибрати потрібний документ знайшовши його розташування на диску.	
Рис. 1.20. Вікно імпорту списку студентів	
Наступним розглянемо Android-додаток	
<div><div><div>Цитирования: 0,01%</div><div>id: 25</div></div><div>"Attendance Tracker",</div></div>	
який дозволяє проводити моніторинг відвідуваності фактично для будь-яких подій, зокрема, й відвідуваності занять студентами. Інтерфейс програми є простим, але не зовсім зрозумілим, що можна відзначити, як недолік цієї програмної системи. Розробники вирішили цю проблему за допомогою детальної інструкції користувача (рис.1.21).	
Рис. 1.21. Вікно інструкції користувача	
Як і у попередньому розглянутому програмному продукті, тут користувачу потрібно самому створити свій розклад та внести список учасників подій. Після цього, він зможе вибрати потрібне заняття та ввести дані про відсутність чи присутність студентів. Важливо зазначити, що додаток забезпечує можливість відправки e-mail повідомлення групі людей з певним станом присутності (рис.1.22), що безумовно є перевагою.	
Рис. 1.22. Вікно для надсилання повідомлення	
Крім того, додаток дозволяє переглядати звіти відвідуваності в обраному форматі та експортувати потрібний користувачу звіт у файл (рис.1.23). Основною перевагою додатку є можливість зробити резервне копіювання усіх даних, для подальшого використання на інших пристроях.	
Рис. 1.23. Вікно для експорту статистики відвідуваності	
Проведення порівняльної характеристики розглянутих програмних систем-аналогів дало можливість виокремити їх основні переваги та недоліки, що відображено в таблиці 1.6.	
Таблиця 1.6	
Порівняльна характеристика програмних продуктів	
Назва програмного продукту	
<div><div><div>Цитирования: 0,04%</div><div>id: 26</div></div><div>"BigTime EDU""MyAttendanceTracker.com""Attendance""Attendance Tracker"</div></div>	
Інтерфейс користувачаЗрозумілий інтерфейс, для роботи з деякими функціями потрібна спеціальна підготовкаЗрозумілий та зручний інтерфейс, максимально адаптований під мобільні платформиЗрозумілий та зручний інтерфейсІнтерфейс є складним для розумінняДопомога користувачуКористувачі системи потребува- тимуть постійної підтримки спеціалістів та техніківПрисутні підказки для початку роботи з системоюПрисутні підказки, які вбудовано безпосередньо в користу- вацький інтерфейсПрисутня інструкція користувачаФункціонал для студентівРеалізована можливість доступу та надано певний функціоналРеалізована можливість доступу та надано певний функціоналДоступ має лише один користувач (тобто тільки викладач)Доступ має лише один користувач (тобто тільки викладач)Робота основних функцій без доступу до ІнтернетуНеможлива, оскільки потрібне підключення до зовнішньої БД на серверіНеможлива, оскільки це веб- орієнтована системаМожлива, оскільки реалізовано локальне сховище данихМожлива, оскільки реалізовано локальне сховище данихМожливість зв'язку із студентамиВідсутняПрисутняВідсутняВідсутняПотреба у додаткових апаратних інтерфейсахПотрібні дорогі біометричні сканери та засоби для їх підтримкиНе потрібно додаткових апаратних інтерфейсівНе потрібно додаткових апаратних інтерфейсівНе потрібно додаткових апаратних інтерфейсівРозглянемо основні переваги, які можна запозичити із кожного продукту та недоліки яких слід уникнути.	
Система	
<div><div><div>Цитирования: 0,01%</div><div>id: 27</div></div><div>"BigTime EDU"</div></div>	
надає засоби для автоматичного попередження студента про переміщення ліміту пропусків, що однозначно слід реалізувати у розроблюваній системі. Обов'язково потрібно уникнути основних недоліків	
<div><div><div>Цитирования: 0,01%</div><div>id: 28</div></div><div>"BigTime EDU"</div></div>	
- використання занадто дорогих компонентів та складного інтерфейсу. Продукт	
<div><div><div>Цитирования: 0,01%</div><div>id: 29</div></div><div>"MyAttendanceTracker"</div></div>	
не має суттєвих недоліків, проте представляє цікавий механізм використання фільтрів для статистичних даних. Єдиним корисним рішенням, яке можна запозичити з додатку	
<div><div><div>Цитирования: 0,01%</div><div>id: 30</div></div><div>"Attendance"</div></div>	
є детальна інструкція користувача, проте, додаток	
<div><div><div>Цитирования: 0,01%</div><div>id: 31</div></div><div>"Attendance Tracker"</div></div>	
має цікаву функцію, яка дозволить викладачу надсилати повідомлення своїм студентам.	

Основними недопустимими для розроблюваної системи недоліками двох вищезгаданих додатків є те, що вони не використовують зовнішньої БД та призначені для одного користувача.	
1.4. Специфікація вимог до Android -додатку для обліку відвідуваності занять студентами	
<p>Після детального дослідження предметної області можна перейти до визначення основних вимог щодо системи, яка розробляється в дипломній роботі. Очевидно, що Android-додаток надає багато цікавих рішень та переваг для вирішення проблеми обліку відвідуваності студентів. Проте необхідно відмітити, що розв'язання деяких завдань на платформі Android є недоцільним через певну обмеженість мобільних платформ. Тому, Android-додаток для обліку відвідуваності занять студентами слід розглядати, як підсистему інтегровану у продукти, які функціонують не на мобільних платформах. Така інтеграція дозволить значно розширити систему для обліку відвідуваності. Найпоширенішими та найбільш сумісними з мобільними платформами є веб-орієнтовані продукти. Фактично, Android-додаток забезпечить виконання майже всього основного функціоналу пов'язаного із певними класами користувачів, надавши для цього зручніші та більш гнучкі рішення та засоби. Крім того, використання мобільної платформи дозволить розв'язати багато проблем та незручностей, які неможливо або дуже складно вирішити за допомогою інших платформ.</p> <p>Аналіз існуючої продукції допоміг побачити непотрібність певних технічних рішень, що зробили б систему занадто складною та невиправдано дорогою. Варто відзначити, що даний етап, також, показав й основні проблеми систем пов'язані з виглядом інтерфейсу користувача та допоміг відсіяти деякий непотрібний функціонал.</p> <p>Далі можливе використання специфічної термінології, тому доцільно розглянути глосарій проекту, який наведено в додатку А.</p> <p>Розглянемо функціональні вимоги до розроблюваної системи. В системі передбачається два типи користувачів: студент і викладач. Кожному із них додаток надаватиме різний функціонал. Спільними для обох видів користувачів буде функція для перегляду повідомлень та функція авторизації. Функції для перегляду статистики відвідуваності та перегляду розкладу, також будуть спільними для студента та викладача, проте, надаватимуть їм різні можливості та матимуть деякі відмінності у інтерфейсі. Інші функції будуть доступними тільки викладачу. Відношення між користувачами та варіантами використання зображено на рисунку 1.24.</p> <p>Рис. 1.24. Діаграма варіантів використання</p> <p>Розглянемо більш детально кожний із варіантів використання. Робота з додатком розпочнеться із функції</p>	
 Цитування: 0,01%	id: 32
"Авторизація".	
Для того щоб отримати можливість користуватися додатком користувачу потрібно буде пройти процес авторизації. Лише після того як система розпізнає користувача він зможе отримати доступ до основного функціоналу користувача. Основним завданням цієї функції є визначення чи користувач має доступ до системи та визначення типу користувача. Також важливо щоб функція забезпечувала безпечне використання даних для авторизації та неможливість їх викрадення.	
У таблиці 1.7 представлено опис варіанту використання	
 Цитування: 0,01%	id: 33
"Авторизація".	
Таблиця 1.7	
Варіант використання	
 Цитування: 0,01%	id: 34
"Авторизація"	
Контекст використанняМетою функції є ідентифікація користувача та надання або ненадання йому відповідних правДійові особиСтудент або викладачПередумова-ТриггерПерший запуск додаткуСценарій- Надання даних потрібних для авторизації	
- Натиснення кнопки підтвердження авторизаціїПост умова Toast про успішність/неуспішність авторизаціїЛогічний і концептуальний опис функціональних можливостей системи для сценарію	
 Цитування: 0,01%	id: 35
"Авторизація",	
відображено на ескізі екранної форми, яка представлена на рисунку 1.25.	
Рис. 1.25. Кадр для сценарію	
 Цитування: 0,01%	id: 36
"Авторизація"	
Далі розглянемо варіант використання	
 Цитування: 0,01%	id: 37
"Переглянути розклад".	
Розклад має відображатися в основному вікні програми. В залежності від типу користувача розклад відображатиме потрібні для нього дані. Головною задачею цієї функції є формування та відображення особистого графіку занять користувача. У випадку викладача вибір заняття із списку передбачатиме відкриття нового вікна для додавання даних про присутність. Важливо забезпечити зручність перегляду розкладу.	
У таблиці 1.8 представлено опис варіанту використання	
 Цитування: 0,01%	id: 38

"Переглянути розклад".	
Таблиця 1.8	
Варіант використання	
 Цитування: 0,01%	id: 39
"Переглянути розклад"	
Контекст використанняМетою функції є формування та відображення персонального розкладу користувачаДійові особиСтудент або викладачПередумоваУспішна авторизаціяТригерЗапуск додаткуСценарій- Вибір дати та дня тижня	
- Перегляд розкладуПост умоваВідображений розкладЛогічний і концептуальний опис функціональних можливостей системи для сценарію	
 Цитування: 0,01%	id: 40
"Переглянути розклад",	
відображено на ескізі екранної форми, яка представлена на рисунку 1.26.	
Рис. 1.26. Кадр для сценарію	
 Цитування: 0,01%	id: 41
"Переглянути розклад"	
Варіант використання	
 Цитування: 0,01%	id: 42
"Переглянути повідомлення"	
має бути реалізованим у багатьох аспектах. Основною задачею цієї функції є відображення користувачу повідомлень отриманих від системи. Можливість використання цієї функції повинна бути реалізована в основному меню. Крім	
того, важливо реалізувати підтримку миттєвих нотифікацій, які дозволятимуть проінформувати користувача що в додатку для нього є нове повідомлення.	
У таблиці 1.9 представлено опис варіанту використання	
 Цитування: 0,01%	id: 43
"Переглянути повідомлення".	
Таблиця 1.9	
Варіант використання	
 Цитування: 0,01%	id: 44
"Переглянути повідомлення"	
Контекст використанняМетою функції є показ повідомлень та механізм інформування користувача про нове повідомленняДійові особиСтудент або викладачПередумоваУспішна авторизаціяТригерНове повідомленняСценарій- Відкрити список усіх повідомлень	
- Переглянути необхідне повідомленняПост умоваПісля перегляду повідомлення стають прочитанимиЛогічний і концептуальний опис функціональних можливостей системи для сценарію	
 Цитування: 0,01%	id: 45
"Переглянути повідомлення",	
відображено на ескізі екранної форми, яка представлена на рисунку 1.27.	
Рис. 1.27. Кадр для сценарію	
 Цитування: 0,01%	id: 46
"Переглянути повідомлення"	
На рисунку 1.28 має бути зображено кадр, який описує частину функції, що відповідатиме за показ безпосередньо самого повідомлення.	
Рис. 1.28 - Кадр для сценарію	
 Цитування: 0,01%	id: 47
"Переглянути повідомлення"	
Варіант використання	
 Цитування: 0,01%	id: 48
"Надіслати повідомлення",	
буде доступний лише викладачу. Ця функція має забезпечити можливість надіслати повідомлення будь-якому студенту якого навчає викладач.	
У таблиці 1.10 представлено опис варіанту використання	
 Цитування: 0,01%	id: 49
"Надіслати повідомлення".	
Таблиця 1.10	
Варіант використання	
	

Цитирования: 0,01%	id: 50
"Надіслати повідомлення"	
<p>Контекст використанняМетою функції є надсилання повідомлення студентуДійові особиВикладачПередумоваУспішна авторизаціяТригерВибір функції в менюСценарій-Надання даних потрібних для авторизації</p> <p>- Натиснення кнопки підтвердження авторизаціїПост умоваToast про успішність/неуспішність надсиланняЛогічний і концептуальний опис функціональних можливостей системи для сценарію</p>	
Цитирования: 0,01%	id: 51
"Надіслати повідомлення",	
відображено на ескізі екранної форми, яка представлена на рисунку 1.29.	
Рис. 1.29. Кадр для сценарію	
Цитирования: 0,01%	id: 52
"Надіслати повідомлення"	
Далі розглянемо варіант використання	
Цитирования: 0,03%	id: 53
"Додати дані про присутність студентів".	
<p>Ця функція буде доступна лише для викладача, після вибору предмета із розкладу. Вона відображатиме список студентів із даними про них відносно вибраного предмету. Також, функція повинна надавати зручні засоби для внесення даних про присутність студентів. Крім того, функція має забезпечити обробку та збереження цих даних.</p> <p>У таблиці 1.11 представлено опис варіанту використання</p>	
Цитирования: 0,03%	id: 54
"Додати дані про присутність студентів".	
Таблиця 1.11	
Варіант використання	
Цитирования: 0,03%	id: 55
"Додати дані про присутність студентів"	
<p>Контекст використанняМетою функції є внесення даних про присутність студентів на заняттіДійові особиВикладачПередумоваУспішна авторизаціяТригерВибір предмету або запуск додатку під час заняттяСценарій- Ввести всі потрібні дані</p> <p>- Натиснути на кнопку підтвердженняПост умоваВведені дані автоматично збережені, toast про успішність/неуспішність діїЛогічний і концептуальний опис функціональних можливостей системи для сценарію</p>	
Цитирования: 0,03%	id: 56
"Додати дані про присутність студентів",	
відображено на ескізі екранної форми, яка представлена на рисунку 1.30.	
Рис. 1.30. Кадр для сценарію	
Цитирования: 0,03%	id: 57
"Додати дані про присутність студентів"	
Варіант використання	
Цитирования: 0,03%	id: 58
"Редагувати дані про присутність студентів"	
<p>буде схожим до попереднього, проте матиме свої особливості. Функція дозволить редагувати уже збережені дані та перезаписувати їх. Щоб функція була активною дані про присутність студентів на відповідному занятті мають бути уже введені та збереженими. При редагуванні даних викладачу буде потрібно вказати причину внесення змін. Таким чином, викладач не матиме права редагувати дані коли йому заманеться без обґрунтованої на це причини.</p> <p>У таблиці 1.12 представлено опис варіанту використання</p>	
Цитирования: 0,03%	id: 59
"Редагувати дані про присутність студентів".	
Таблиця 1.12	
Варіант використання	
Цитирования: 0,03%	id: 60
"Редагувати дані про присутність студентів"	
<p>КонтекстМетою функції є надання можливості редагування ужевикористаннязбережених данихДійові особиВикладачПередумоваДодано дані про присутність студентів</p>	
Цитирования: 0,15%	id: 61
"ТригерВибір предмету із збереженими данимиСценарій- Введення потрібних даних	
<p>- Натиснення кнопки підтвердження</p> <p>- Введення причини змін</p>	

- Натиснення кнопки підтвердженняПост умова Toast про успішність/неуспішність діїЛогічний і концептуальний опис функціональних можливостей системи для сценарію "	
Редагувати дані про присутність студентів	
Цитування: 0,04%	id: 62
", збігається з ескізом екранної форми для варіанту використання "	
Додати дані про присутність студентів". Тобто, при редагуванні даних користувач здійснить новий ввід даних, а старі дані при цьому будуть перезаписані. Відмінність функції редагування від варіанту використання "Додати дані про присутність студентів" полягатиме в тому, що користувач буде змушений пояснити причину внесення правок, яка відповідно теж буде збереженою разом із новими даними.	
Відмінність між варіантом використання	
Цитування: 0,03%	id: 63
"Редагувати дані про присутність студентів"	
та варіантом використання	
Цитування: 0,03%	id: 64
"Додати дані про присутність студентів"	
відображено на рисунку 1.31.	
Рис. 1.31. Кадр для сценарію	
Цитування: 0,01%	id: 65
"Authorization"	
Варіант використання	
Цитування: 0,02%	id: 66
"Переглянути статистику відвідуваності"	
має бути доступним для викладача та студента. Основним завданням функції є надання засобів для перегляду статистичних даних різного вмісту. Користувачу має надаватися можливість вибрати вид статистики. Також, функція має забезпечити коректне відображення статистичних даних.	
У таблиці 1.13 представлено опис варіанту використання	
Цитування: 0,02%	id: 67
"Переглянути статистику відвідуваності".	
Таблиця 1.13	
Варіант використання	
Цитування: 0,02%	id: 68
"Переглянути статистику відвідуваності"	
Контекст використанняМетою функції є можливість перегляду різних статистичних данихДійові особиСтудент або викладачПередумова-ТригерПерший запуск додаткуСценарій- Надання даних потрібних для авторизації	
- Натиснення кнопки підтвердження авторизаціїПост умова Toast про успішність/неуспішність авторизаціїНа рисунку 1.32 зображено ескіз що описує частину функції	
Цитування: 0,02%	id: 69
"Переглянути статистику відвідуваності"	
та дозволяє вибрати вид потрібної статистики.	
Рис. 1.32. Кадр для сценарію	
Цитування: 0,02%	id: 70
"Переглянути статистику відвідуваності"	
На рисунку 1.33 зображено кадр що відповідає за відображення статистичних показників.	
Рис. 1.33. Кадр для сценарію	
Цитування: 0,02%	id: 71
"Переглянути статистику відвідуваності"	
Після детального розгляду варіантів та їх розкадрування можна скласти специфікацію функціональних вимог, яка наведена в таблиці 1.14.	
Таблиця 1.14	
Специфікація функціональних вимог	
Ідентифікатор вимогиНазва вимогиАтрибути вимогиПріоритетСкладністьКонтакт1.АвторизаціяОбов'язковоСередняВикладач, студент2.Перегляд розкладуОбов'язковоНизькаВикладач, студент3.Переглянути повідомленняОбов'язковоВисокаВикладач, студент4.Надіслати повідомленняОбов'язковоВисокаВикладач, студент5.Додати дані про присутність студентівОбов'язковоВисокаВикладач, студент6.Редагувати дані про присутність студентівОбов'язковоВисокаВикладач, студент7.Переглянути статистику відвідуваностіОбов'язковоВисокаВикладач, студентСпецифікація нефункціональних вимог наведена у таблиці 1.15.	
Таблиця 1.15	

Специфікація нефункціональних вимог

Ідентифікатор вимоги Назва вимоги Атрибути
 вимоги Пріоритет Складність Контакт 1. Застосовність Рекомендовано Висока Викладач, студент 1.1. Час, необхідний для навчання звичайних і досвідчених користувачів Обов'язково Середня Викладач, студент 1.2. Час відгуку для типових завдань Рекомендовано Висока Викладач, студент 1.3. Вимоги по відповідності загальним стандартам застосовності Рекомендовано Висока Викладач, студент 2. Надійність Обов'язково Висока Викладач, студент 2.1. Точність Обов'язково Висока Викладач, студент 3. Робочі характеристики Обов'язково Середня Викладач, студент 3.1. Швидкодія для транзакції Обов'язково Висока Викладач, студент 3.2. Продуктивність Обов'язково Середня Викладач, студент 3.3. Використання ресурсів Опційно Середня Викладач, студент 4. Вимоги до документації Обов'язково Середня Викладач, студент

Результатом визначення та розгляду функціональних та нефункціональних вимог є сформована специфікація вимог до розроблюваного **Android**-додатку для обліку відвідуваності занять студентами. Після виконання цього етапу можна переходити безпосередньо до проектування системи.

РОЗДІЛ 2. ПРОЄКТУВАННЯ **ANDROID**-ДОДАТКА ДЛЯ ОБЛІКУ ВІДВІДУВАННЯ ЗАНЯТЬ СТУДЕНТАМИ ЗАКЛАДІВ ВИЩОЇ ОСВІТИ

2.1. Розробка архітектури програмної системи

Після розгляду функціональних та нефункціональних вимог і формування специфікації можна перейти до етапу проектування. Оскільки, планується, що розроблюваний **Android**-додаток буде підсистемою веб-орієнтованого продукту, то, перш за все, потрібно розглянути механізми, які будуть використовуватися для інтеграції двох систем. Враховуючи особливості платформи **Android**, у такому випадку найкраще буде використовувати архітектуру, яка буде наслідувати архітектуру **REST** або принаймні буде схожою на неї. Такий підхід дозволить **Android**-додатку отримати доступ до бази даних не через пряме підключення, а через **HTTP** протокол. При цьому для отримання даних з БД та їх обробки будуть використовуватися ресурси сервера веб-орієнтованої системи. Таким чином, підхід до проектування, що базуватиметься на **RESTful** архітектурі дозволить легко відділити клієнтський додаток від програмних інтерфейсів для роботи з базою даних, що є беззаперечною перевагою для системи, яка передбачає інтеграцію з іншими системами[1]. Фактично, розроблюваний додаток стане не прив'язаним ані до структури БД, ані до використовуваної РСУБД. Що, в свою чергу надасть можливість легко перепроєктувати систему для аналогічних веб-орієнтованих програмних систем, що вирішують проблему обліку відвідуваності студентів. Для цього потрібно буде лише внести певні коригування у **API** для роботи з БД. В перспективі розроблювану систему, можна було б зробити більш універсальною та гнучкою і забезпечити її швидку інтеграцію фактично з будь-якою веб-орієнтованою системою для обліку відвідуваності. Слід зазначити, що використання **RESTful** підходу має й багато мінусів основним з яких є використання посередника між клієнтом та сервером БД. Для розроблюваного **Android**-додатку ці мінуси не є критичними чи дуже важливими, оскільки додаток має забезпечити роботу основного функціоналу й у режимі

Цитування: **0,01%**

id: **72**

"**API**",

а для цього доведеться зберігати усі потрібні данні у локальному сховищі.

Ще одною надзвичайно важливою причиною для вибору **RESTful** архітектури є те, що для забезпечення стабільної роботи функціоналу потрібно використання певного серверу БД та сервісу **Google Cloud Messaging**. Для забезпечення функцій надсилання, отримання та генерування миттєвих повідомлень **GCM** повинен мати доступ до бази даних, який фактично не можливо надати без використання веб-серверу.

Отже для забезпечення потрібного функціоналу та стабільної роботи **Android**-додатку для обліку відвідуваності занять студентами потрібно використання трьох серверів. Зв'язок між усіма вузлами та компонентами проєктованої системи відображено на рисунку 2.1.

Рис. 2.1. Діаграма розгортання розроблюваної програмної системи

Розглянемо детальніше залежності між компонентами програмного забезпечення для повнішого представлення статичного аспекту архітектури додатку.

В системі буде два пакети, кожний із яких відповідатиме за функціонал для відповідного класу користувача. Крім того, в системі будуть окремі модулі відповідальні за більшість операцій з даними та за можливість роботи з веб-сервером через використання мережевого протоколу. Ці модулі будуть використовуватися обома вищезгаданими пакетами. Залежності між описаними компонентами відображено на рисунку 2.2.

Рис. 2.2. Діаграма компонентів розроблюваної програмної системи

Структурна модель додатку є досить складною. Більшість модулів системи відповідатимуть за інтерфейс або стосуватимуться особливостей забезпечення роботи з платформою **Android**. Тому їх відображення на даному етапі, є не потрібним оскільки воно зробить діаграму класів занадто громіздкою та важкою для читання та розуміння. Отже, на рисунку 2.3 діаграма відображена лише тими класами які забезпечуватимуть виконання бізнес-логіки, яку можливо відділити від графічної частини та частини для організації роботи з даними.

Рис. 2.3. Діаграма класів

Наведена вище діаграма класів фактично показує структуру модуля **UserData**. Кожний із класів буде використовуватися для роботи з даними та забезпечуватиме певну бізнес-логіку, яку можливо відділити від елементів, що забезпечуватимуть графічне представлення. Відображення інших класів, що відповідатимуть за частину функціоналу не доцільне, оскільки вони будуть тісно пов'язані із класами що відповідатимуть за графічний інтерфейс користувача.

Розглянемо детальніше динамічну складову архітектури системи. Для цього графічно опишемо основні бізнес-процеси та бізнес-правила, закладені в програмну систему, за допомогою засобів мови моделювання **UML**[2].

На рисунку 2.4. зображена діаграма послідовності для варіанту використання

Цитирования: 0,01% id: 73

"Надсилання повідомлення",

яка відображає взаємодії об'єктів впорядкованих за часом.

Рис. 2.4. Діаграма послідовності для функції

Цитирования: 0,01% id: 74

"Надсилання повідомлення"

Ця діаграма ілюструє процес надсилання повідомлення викладачем та показує коли і протягом якого часу будуть активні графічні інтерфейси. Вданому випадку зображено перехід до [activity](#)

Цитирования: 0,01% id: 75

"SendMessage",

не із основного меню а з [activity](#)

Цитирования: 0,01% id: 76

"Presence",

яке відповідає за збір даних про відвідуваність. Після основного вікна для надсилання повідомлення викладачу буде потрібно ввести необхідні дані та підтвердити надсилання повідомлення. Якщо всі поля будуть заповнені вірно, [activity](#)

Цитирования: 0,01% id: 77

"SendMessage"

відправить повідомлення на сервер.

Процес надсилання повідомлення демонструє діаграма станів яка зображена на рисунку 2.5.

Рис. 2.5 Діаграма станів для системного процесу

Цитирования: 0,01% id: 78

"Надсилання повідомлення"

Важливим етапом є відображення нових повідомлень. При отриманні сповіщення про нові повідомлення користувач матиме змогу прочитати повідомлення безпосередньо відкривши додаток або натиснувши на сповіщення, що автоматично запустить додаток. Цей процес описано за допомогою діаграми станів зображеної на рисунку 2.6.

Рис. 2.6. Діаграма станів для функції

Цитирования: 0,02% id: 79

"Перевірка наявності нових повідомлень"

На рисунку 2.7 зображена діаграма послідовності, яка відображає взаємодії графічних об'єктів впорядкованих за часом для варіанту використання

Цитирования: 0,01% id: 80

"Перегляд статистики"

та його можливості. Функція буде доступною тільки з головного меню та працюватиме лише тоді, коли буде доступ до мережі Інтернет.

Рис. 2.7. Діаграма послідовності для функції

Цитирования: 0,01% id: 81

"Перегляд статистики"

Далі розглянемо одні з найважливіших функцій для викладача

Цитирования: 0,02% id: 82

"Створення списку присутніх",

яка відповідатиме за збір даних про відвідуваність. Доступ до цієї опції можна буде отримати при виборі предмету із персонального розкладу у головному вікні програми. Функція відображатиме список студентів для обраного заняття та забезпечуватиме збір даних. Дані про студентів будуть завантажуватися із локального сховища. При відкритті програми під час заняття автоматично відкриватиметься список студентів для відповідного заняття, щоб викладач міг швидко відмітити присутніх.

Цей процес відображено за допомогою діаграми послідовності на рисунку 2.8.

Рис. 2.8. Діаграма послідовності для функції

Цитирования: 0,02% id: 83

"Створення списку присутніх"

Для кращого розуміння механізму

Цитирования: 0,02% id: 84

"Створення списку присутніх",

його слід розглянути із боку системних процесів, що будуть відповідальні за його реалізацію. Дії системних процесів показує діаграма станів, яка зображена на рисунку 2.9.


Рис. 2.9. Діаграма станів для функції

Цитирования: 0,02% id: 85

"Створення списку присутності"

Механізм надсилання списку відображено на рисунку 2.10.

Рис. 2.10. Діаграма станів для функції

 Цитування: 0,02%	id: 86
"Надсилання списку присутності"	

Важливою вимогою є забезпечення безпечної авторизації користувачів. Для цього найкраще буде реалізувати механізм подібний до подвійної автентифікації. Після введення користувачем персонального Ідентифікатора на його зареєстрований у системі [email](#) буде відправлятися тимчасовий пароль, який використовуватиметься для підтвердження прав доступу до додатку. Цей процес відображено за допомогою діаграми послідовності на рисунку 2.11.

Рис. 2.11. Діаграма послідовності для функції

 Цитування: 0,01%	id: 87
"Авторизація"	

Основні функції, що забезпечуватимуть чітке виконання вимог до системи було розглянуто вище. Решта функціоналу системи буде відповідати за обробку даних та представлення даних користувачу, тому його детальний розгляд є недоцільним. Оскільки, реалізація цих функцій практично не матиме ніякого впливу на бізнес-логіку та бізнес-правила механізму їх реалізації будуть обрані безпосередньо розробниками проекту.

2.2. Проектування структури бази даних

Зважаючи на те, що розроблювана система повинна бути інтегрованою із веб-орієнтованим продуктом, потрібно розробити рекомендаційну структуру бази даних, яка буде містити дані які будуть необхідні для роботи [Android](#)- додатку. Оскільки, система буде побудована на базі [RESTful](#) архітектури, зрозуміло, що навіть та частина БД, яка буде потрібна для підтримки функціонування додатку допускати певні відмінності від спроектованої бази. Майже усі відмінності можна буде нівелювати, як засобами самої РСУБД, так і засобами веб-сервера, який надаватиме інтерфейс для доступу до БД. У якості РСУБД для цього проекту буде використовуватися [MySQL](#).

Враховуючи той фактор, що більшість інформації, яку використовуватиме [Android](#)-додаток буде зберігатися у локальному сховищі даних потрібно детальніше розглянути методи доступу до глобальної бази даних.

Для того щоб отримати доступ до глобальної БД додаток надсилатиме запити через протокол [HTML](#) до спеціального веб-застосунку, який розташований на веб-сервері, передаючи йому при цьому необхідну інформацію. Веб-застосунок, опрацьовуючи запити додатку, підключатиметься до сервера з БД для отримання необхідних даних, після чого, оброблятиме дані перетворивши їх у потрібний формат та надсилатиме клієнту-додатку відповідь із даними у вигляді [JSON](#). Отримані дані додаток збереже у [Shared Preferences](#).

Після проведеного аналізу предметної області можна визначити дані, які будуть необхідні додатку та визначити, які дані додаток буде повинний зберігати в локальному сховищі.

Для функціонування додатку у режимі

 Цитування: 0,01%	id: 88
"Offline"	

обов'язково потрібно зберігати у локальному сховищі дані що репрезентуватимуть персональний розклад користувача та дані про уже отримані повідомлення. У випадку викладача необхідно також буде зберігати списки студентів. Усі операції із статистичними даними працюватимуть тільки в режимі

 Цитування: 0,01%	id: 89
"Online"	

Дані про відвідуваність, які вводитиме викладач будуть зберігатися локально та відправлятися на сервер при появі стабільно підключення до мережі Інтернет.

На етапі експлуатації системи для функціонування [Android](#)-додатку будуть використовуватися наступні об'єкти та дані про них:

Предмет у розкладі:

Назва;

Тип;

Викладач;

Група;

Аудиторія;

День тижня коли проходять заняття;

Дати занять;

Час коли відбувається заняття;

Тривалість заняття;

Дані про парність/непарність тижня.

Студент:

Ім'я;

Прізвище;

Email;

Курс;

Дані про присутність на заняттях;

Назва групи;

Назва факультету.

Викладач:

Ім'я;

Прізвище;

Email.

Повідомлення:

Тема;

Час створення;

Статус;

Пріоритет;

Відправник;

Отримувач.

Користувач;

GCM ID;

Тип доступу;

Пароль.

Після опису вхідної та вихідної інформації, яка обробляється в рамках функцій предметної області розроблюваної програмної системи можна зобразити глобальну даталогічну модель даних у вигляді логічної моделі ERD на рисунку 2.12.

Рис. 2.12. Логічна модель ERD

Для всіх відношень автоматично створюються індекси по первинному ключі.

Індексування значно пришвидшує пошук даних у БД. Проте надмірне використання індексування призводить до втрати продуктивності виконання запитів на поновлення та знищення[3].

Для ефективного пошуку передбачено створення foreign keys для наступних атрибутів:

ClassID - зовнішній ключ

TeacherID- зовнішній ключ

LessonID- зовнішній ключ

ScheduleID- зовнішній ключ

PresenceStatusID - зовнішній ключ

Також передбачається, що атрибут Email буде індексованим.

Для забезпечення правильного збереження даних на етапі надсилання даних про відвідуваність передбачається використання транзакцій.

При написанні інтерфейсу комунікації БД із веб-сервером розробники повинні будуть використовувати Stored Procedures, для нестандартних операцій з даними БД.Для можливості генерування автоматичних повідомлень базою даних будуть створені спеціальні тригери.Результатом фізичного проектування є DDL код, який надано в додатку.

Після побудови логічної моделі та аналізу даних можна побудувати фізичну модель ERD, яка зображена на рисунку 2.13.

Рис. 2.13. Фізична модель ERD

РОЗДІЛ III. ПРОГРАМНА РЕАЛІЗАЦІЯ ANDROID-ДОДАТКА ДЛЯ ОБЛІКУ ВІДВІДУВАННЯ ЗАНЬ ТРУДЕНТАМИ ЗАКЛАДІВ ВИЩОЇ ОСВІТИ

3.1. Програмна реалізація проекту

Найкращим вибором серед мов для написання додатку для мобільної платформи Android сьогодні є Java. Java є широко розповсюдженою мовою програмування, яка має багато переваг. Основною перевагою використання мови Java у платформі Android є те, що вона виконується на віртуальній машині і тому немає необхідності перекомпільовувати додатки для кожного телефону.

Додаток, що розробляється, буде підтримуватися на усіх версіях операційної системи Android починаючи із SDK 16(тобто версії Android 4.1.2). Це рішення можна обґрунтувати тим, що підтримувати сумісність усіх функцій системи із попередніми застарілими версіями Android буде досить складно та недоцільно, оскільки за останніми статистичними даними

ці версії використовують менш ніж 5% користувачів платформи [Android](#) і число таких користувачів продовжує суттєво скорочуватися.

Для написання веб-сервісів, які забезпечуватимуть доступ до глобальної бази даних, оптимальним вибором є використання мови [PHP](#), яка має достатню кількість засобів, щоб швидко та якісно розв'язати проблему доступу до БД та має достатню кількість уже готових фреймворків для створення [REST API](#) та роботи з ним. Для написання [REST API](#) використано мікрофреймворк [Slim](#).

Основним завданням веб-сервісів, які надають додатку засоби для комунікацій із БД на сервері, є можливість отримання та надсилання даних через мережевий протокол [HTTP](#). Для забезпечення зручної та швидкої роботи із даними, усі дані пересилаються у форматі [JSON](#). Частина коду, яка наведена нижче демонструє частину [API](#) для роботи із базою даних та забезпечує можливість отримання викладачем списку студентів із глобальної БД. За допомогою методу [POST](#) додаток пересилає потрібні ідентифікаційні дані, після чого скрипт витягує потрібні дані із БД, форматує їх та надсилає додатку.

```
$con=connect(); if ($con!=null) {

mysql_set_charset($con,

Цитирования: 0,01% id: 90
"utf8"

);

$ID=mysql_real_escape_string($con,$_POST['ID']);

$stmt = mysql_prepare($con, 'CALL GetStudents(?)');

mysql_stmt_bind_param($stmt, 'i', $ID);

mysql_stmt_execute($stmt);

$result = $stmt->get_result(); close($con);

if (mysql_num_rows($result) 0) {

$response[

Цитирования: 0,01% id: 91
"Classes"

] = array();

$className=null;

while ($row = mysql_fetch_array($result)) {

if($className==null)

{

$className=$row[

Цитирования: 0,02% id: 92
"ClassID"];

$Class[$className]

=array();

}

else if($className!=$row[

Цитирования: 0,01% id: 93
"ClassID"]

)

{

$className=$row[

Цитирования: 0,02% id: 94
"ClassID"];

$Class[$className]

=array();

}

}

$student=array();

$student[

Цитирования: 0,12% id: 95
"ID"]=$row["ID"];

$student["Name"]=$row["Name"];

$student["Surname"]=$row["Surname"];

$student["Email"]=$row["Email"]

;

array_push($Class[$className] , $student );

}
```

```

array_push($response[
Цитирования: 0,01% id: 96
"Classes"
], $Class);

$response[
Цитирования: 0,01% id: 97
"success"
] = 1;

echo json_encode($response,JSON_UNESCAPED_UNICODE);

} else {

$response[
Цитирования: 0,01% id: 98
"success"]
= 0;

$response[
Цитирования: 0,01% id: 99
"message"
] =
Цитирования: 0,02% id: 100
"User not exist"
;

echo json_encode($response,JSON_UNESCAPED_UNICODE);

}

}

else {

$response[
Цитирования: 0,01% id: 101
"success"]
= 0;

$response[
Цитирования: 0,01% id: 102
"message"
] =
Цитирования: 0,01% id: 103
"Connection failed"
;

echo json_encode($response,JSON_UNESCAPED_UNICODE);

}}

else {

$response[
Цитирования: 0,01% id: 104
"success"]
= 0;

$response[
Цитирования: 0,01% id: 105
"message"
] =
Цитирования: 0,03% id: 106
"Required field ( ) is missing"
; echo json_encode($response,JSON_UNESCAPED_UNICOD}

Усі процеси надсилання та отримання даних додатком виконуються у бекграунді за
допомогою спеціального native класу AsyncTask. Під час процесу надсилання чи
завантаження даних користувач бачитиме повідомлення про зайнятість додатку
Цитирования: 0,03% id: 107
"Завантаження даних, зачекайте будь ласка...".

Для забезпечення стабільної роботи із проколом HTTP використовується спеціальна
бібліотека OKHTTP. Ця бібліотека підтримує усі сучасні засоби для зручної роботи із
протоколом HTTP. Для отримання даних додатком та коректного їх опрацювання у форматі
JSON був розроблений спеціальни клас, лістинг коду якого наведено нижче.

public class JSONParser {

public static final MediaType JSON = MediaType.parse(
Цитирования: 0,01% id: 108

```

```

"application/json; charset=utf-8"
);

OkHttpClient client = new OkHttpClient(); static InputStream is = null;

static JSONObject jsonObj = null; static String jsonAns = null;

JSONObject postRequest(String url, RequestBody json) throws IOException {

try {

Request request = new Request.Builder().url(url).post(json).build();

Response response = client.newCall(request).execute();

jsonAns = response.body().string();

} catch (UnsupportedEncodingException e)

{ Log.e(
❏ Цитирования: 0,01% id: 109
"e:",
Log.getStackTraceString(e));

} catch (IOException e) {

Log.e(
❏ Цитирования: 0,01% id: 110
"e:",
Log.getStackTraceString(e));

} catch (Exception e) {

Log.e(
❏ Цитирования: 0,01% id: 111
"e:",
Log.getStackTraceString(e));

}

try {

jsonObj = new JSONObject(jsonAns);

} catch (JSONException e) {

e.toString();

}

Log.e(
❏ Цитирования: 0,01% id: 112
"JSON Parser",
❏ Цитирования: 0,02% id: 113
"Error parsing data "
+

return jsonObj;

}

}

Для локального збереження даних використовується спеціальний файл SharedPreferences
формату xml. Цей файл зберігається у захищеній пам'яті пристрою Android та дозволяє
зберігати дані у форматі key-value. У фрагменті коду наведеному нижче продемонстровано
завантаження даних із SharedPreferences та їх обробка для подальшої можливості роботи із
потрібними даними.

protected String doInBackground(String... args) { try {

sPref = this.getSharedPreferences (
❏ Цитирования: 0,01% id: 114
"com.chudentAttendance",
Context.MODE_PRIVATE);

try {

days = new JSONArray(sPref.getString(
❏ Цитирования: 0,01% id: 115
"Schedule",
❏ Цитирования: 0% id: 116
""
));

} catch (JSONException e) {

Log.e(
❏

```

```
Цитирования: 0,01% id: 117
"e.";
Log.getStackTraceString(e));
}

for (int i = 0; i < days.length(); i++) { JSONObject day = days.getJSONObject(i); Log.e(
Цитирования: 0,01% id: 118
"e.",
day.toString());

JSONArray d = day.getJSONArray("dayOfWeek"); for (int j = 0; j < d.length(); j++) {

d.getJSONObject(j);

JSONObject subject =

subject.getString(
Цитирования: 0,01% id: 119
"Name"
);

String id = subject.getString(
Цитирования: 0,01% id: 120
"id"
); String name =

String dayOfWeek =

subject.getString(
Цитирования: 0,01% id: 121
"DayOfWeek"
);

String classroom =

Цитирования: 0,01% id: 122
"Classroom:"
+ subject.getString(
Цитирования: 0,01% id: 123
"Classroom"
);

String time =

subject.getString(
Цитирования: 0,01% id: 124
"Time"
);

HashMap<String, String> map = new HashMap<String, String> ();

map.put(
Цитирования: 0,01% id: 125
"id",
id);

map.put(
Цитирования: 0,01% id: 126
"Name",
name); map.put(
Цитирования: 0,01% id: 127
"Classroom",
classroom); map.put(
Цитирования: 0,01% id: 128
"DayOfWeek",
dayOfWeek); map.put(
Цитирования: 0,01% id: 129
"Time",
time); scheduleList.add(map);

}

}

} catch (JSONException e) {

Log.e(
Цитирования: 0,01% id: 130
"e.",
Log.getStackTraceString(e));

}

return null;
```

}

Для організації зручного інтерфейсу користувача додаток використовуватиме стандартні елементи інтерфейсу, які прийнято використовувати для платформи [Android](#). Для забезпечення зручності інтерфейсу та його презентабельного вигляду практично усі елементи було кастомізовано. Усі іконки, які використовуються в додатку є стандартними та знайомими для користувачів [Android](#). Основне вікно додатку, яке дозволяє переглянути користувачам їх особистий розклад, містить елементи керування у меню, які дозволяють переключатися між днями розкладу. Крім того, у основному вікні можна переміщуватися між днями за допомогою [swipe](#) або можна вибрати конкретну дату розклад для якої потрібно відобразити. У тому випадку, коли користувач є викладачем він зможе вибрати предмет в розкладі та заповнити дані про відвідуваність. У всіх вікнах додатку присутнє меню у формі [Action Bar](#) та працює кнопка

» Цитування: 0,01%

id: 131

"назад",

яка дозволяє повернутися до попереднього вікна або попереднього стану вікна.

Вікна, які дозволятимуть користувачам переглядати статистику, функція отримання нових повідомлень та функція надсилання даних про відвідуваність буде працювати тільки в тому випадку якщо пристрій матиме стабільне підключення до мережі Інтернет. За перевірку підключення до мережі відповідає клас [InternetConnectionChecker](#), лістинг коду цього класу наведено нижче.

```
public class InternetConnectionChecker { Context _context;

public InternetConnectionChecker(Context context) {

_context = _context;

}

public boolean isOnline() {

ConnectivityManager cm = (ConnectivityManager)

_context.getSystemService(Context.CONNECTIVITY_SERVICE);

NetworkInfo netInfo = cm.getActiveNetworkInfo(); return netInfo != null &&

netInfo.isConnectedOrConnecting();

}

}
```

Для того, щоб працювали [push notification](#), які сповіщатимуть про нове повідомлення миттєво, використовується сервіс [GCM](#). Нижче відображено структуру повідомлення, яке надсилатиметься через [GCM](#).

```
'message' = 'here is a message. message',

'title' = 'This is a title. title', 'vibrate' = 1,

'sound' = 1,

'created_at' = '2019-05-15 12:00:00'
```

Після відкриття отриманої нотифікації автоматично буде запущено додаток де користувач зможе отримати нотифікацію. Нотифікації отримуватимуть навіть коли додаток або пристрій знаходиться в неактивному стані. Єдиною вимогою для роботи миттєвих повідомлень є підключення до Інтернету. Нижче наведено код функції, яка виконує обробку отриманого повідомлення.

@Override

```
public void onMessageReceived(String from, Bundle bundle) { String title = bundle.getString(
```

» Цитування: 0,01%

id: 132

"title"

);

```
String message = bundle.getString(
```

» Цитування: 0,01%

id: 133

"message"

```
); String image = bundle.getString(
```

» Цитування: 0,01%

id: 134

"image"

);

```
String timestamp = bundle.getString(
```

» Цитування: 0,01%

id: 135

"created_at"

);

```
if (!NotificationUtils.isApplsInBackground(getApplicationContext()))
```

{

```
Intent pushNotification = new Intent(Config.PUSH_NOTIFICATION);
```

```
pushNotification.putExtra(
```

»

<div>Цитирования: 0,01%</div> <div>"message",</div> <div>message);</div> <div>LocalBroadcastManager.getInstance(this).sendBroadcast(pushNotification);</div> <div>NotificationUtils notificationUtils = new NotificationUtils();</div> <div>notificationUtils.playNotificationSound();</div> <div>} else {</div> <div>Intent resultIntent = new Intent(getApplicationContext(), MainActivity.class);</div> <div>resultIntent.putExtra(</div>	id: 136
<div>Цитирования: 0,01%</div> <div>"message",</div> <div>message);</div> <div>if (TextUtils.isEmpty(image))</div> <div>{ showNotificationMessage(getApplicationContext(),</div> <div>title, message, timestamp, resultIntent);</div> <div>}</div> <div>}</div>	id: 137
<h3>3.2. Програмна реалізація бази даних</h3> <p>Оскільки розроблюваний додаток націлений на інтеграцію із веб- продуктами, то зрозуміло, що бізнес-правила, інструменти для роботи з БД та послідовність дій створення самої бази даних може різнитися в залежності від продукту з яким додаток буде інтегруватися. Фактично, ані вибір інших інструментів, ані СУБД та середовища її функціонування ніяк не вплине на роботу додатку, адже за зв'язок із БД на сервері відповідатимуть веб-служби, які можна буде переписати потрібним чином, при цьому не вносячи ніяких змін у самий Android-додаток.</p> <p>Можливість використання Android-додатку передбачає, що усі потрібні для його роботи таблиці уже створені та наповнені даними, а REST API при потребі переписано та адаптовано у відповідності до веб-продукту та СУБД.</p> <p>Єдиною вимогою при інтеграції додатку із веб-системою є використання рекомендованої структури бази даних, тобто забезпечення додатку усіма потрібними даними.</p> <p>Для можливості надходження повідомлень від системи, потрібна реалізація спеціальних тригерів, які генеруватимуть потрібні повідомлення. Найкращим варіантом для підрахунку статичних даних теж є використання тригерів.</p> <p>Для отримання та запису даних Android-додаток використовуватиме веб- служби REST API, які в свою чергу отримуватимуть дані з БД та записуватимуть дані в БД. Для роботи веб-служб із БД найкраще буде використовувати stored procedures.</p> <p>Передбачаються наступні stored procedures для забезпечення роботи додатку:</p>	
<div>Цитирования: 0,01%</div> <div>"TemporaryUserPassword"</div> <div>- після авторизації користувачу надсилатиметься тимчасовий пароль, ця процедура зберігатиме його в базу даних;</div>	id: 138
<div>Цитирования: 0,01%</div> <div>"GetTeacherSubjects"</div> <div>- отримує із БД персональний розклад для викладача;</div>	id: 139
<div>Цитирования: 0,01%</div> <div>"GetStudentSubjects"</div> <div>- отримує із БД персональний розклад для студента;</div>	id: 140
<div>Цитирования: 0,01%</div> <div>"GetStudents"</div> <div>- отримує із БД список студентів, для викладача, у яких він веде заняття;</div>	id: 141
<div>Цитирования: 0,01%</div> <div>"GetClassSubjects"</div> <div>- отримує таблицю зв'язків між класами та предметами;</div>	id: 142
<div>Цитирования: 0,01%</div> <div>"PushLesson"</div> <div>- записує в БД дані про проведене заняття;</div>	id: 143
<div>Цитирования: 0,01%</div> <div>"PushPresenceStatus"</div> <div>- записує в БД дані про відвідуваність занять студентами;</div>	id: 144
<div>Цитирования: 0,01%</div> <div>"GetStatForTeacherAllSubjects"</div> <div>- отримує з БД статистичні дані про відвідуваність по всіх предметах викладача;</div>	id: 145

<p>Цитирования: 0,01%</p> <p>"GetStatForTeacherSpecificSubjects"</p> <p>- отримує з БД статистичні дані про відвідуваність по певному предметі викладача;</p>	id: 146
<p>Цитирования: 0,01%</p> <p>"GetStatForTeacherSpecificStudent"</p> <p>- отримує з БД статистичні дані про відвідуваність по всіх предметах викладача для певного студента;</p>	id: 147
<p>Цитирования: 0,01%</p> <p>"GetStatForStudentAllSubjects"</p> <p>- отримує з БД статистичні дані про відвідуваність по всіх предметах студента;</p>	id: 148
<p>Цитирования: 0,01%</p> <p>"GetStatForStudentSpecificSubjects"</p> <p>- отримує з БД статистичні дані про відвідуваність по певному предметі студента;</p>	id: 149
<p>Цитирования: 0,01%</p> <p>"GetMessages"</p> <p>- отримує з БД повідомлення для користувача;</p>	id: 150
<p>Цитирования: 0,01%</p> <p>"CreateMessage"</p> <p>- записує в БД створене повідомлення;</p>	id: 151
<p>Цитирования: 0,01%</p> <p>"ChangeMessage"</p> <p>- змінює в системі певні дані повідомлення. Усі згадані stored procedures наведені в DDL коді Додатку.</p> <p>Важливо щоб надсилання даних про відвідуваність проведеного викладачем заняття записувалися у базу даних у межах транзакції.</p>	id: 152
<h3>3.3.Тестування</h3> <p>При розробці програмного продукту було проведено модульне та системне тестування. Було проведено функціональне тестування та тестування графічного інтерфейсу. Під час розробки системи та її тестування було налаштовано та наповнено тестову базу даних.</p> <p>Під час написання програмної реалізації більша частина функціоналу була покрита модульними тестами. Для модульного тестування було використано фреймворк JUnit. Загалом було написано 240 тестових випадків. Результати тестування відображено на рисунку 3.1.</p> <p>Рис. 3.1. Результат модульного тестування</p> <p>Під час фази розробки тестів було спроектовано функціональні тестові випадки. Таблиця показує розподіл функціональних тестових випадків і наборів тестових даних для цих випадків за варіантами використання.</p> <p>Таблиця 3.1.</p> <p>Розподіл функціональних тестових</p> <p>Варіанти використанняТестові випадкиТестові данніCheck logging22Check option</p>	
<p>Цитирования: 0,01%</p> <p>"Exit"</p> <p>22Check option</p>	id: 153
<p>Цитирования: 0,01%</p> <p>"TakeAttendance"</p> <p>22Check option</p>	id: 154
<p>Цитирования: 0,01%</p> <p>"Send Message"</p> <p>33Check option</p>	id: 155
<p>Цитирования: 0,01%</p> <p>"Get Statistics"</p> <p>22Check option</p>	id: 156
<p>Цитирования: 0,02%</p> <p>"Send attendance data"</p> <p>223агалом1212</p>	id: 157
<h3>3.4. Розгортання програмного продукту</h3> <p>Розроблений програмний продукт орієнтований на інтеграцію з функціонуючим веб-орієнтованим працюючим аналогом. Для коректної роботи клієнтської частини, серверна частина системи має бути повністю налаштована. Усі дані, які потрібні для роботи додатку мають бути присутніми у БД веб-орієнтованого продукту. Для роботи Android-додатку може бути використана фактично будь яка РСУБД та може бути використана основна БД або проміжна база даних синхронізована для імпорту й експорту з основною БД. Єдиною</p>	

вимогою до структури БД є наявність усіх даних, які потрібні для роботи основного функціоналу [Android](#)-додатку. Основним завданням при налаштуванні серверної частини системи є забезпечення рекомендованої структури БД з усіма механізмами для обробки даних або адаптація [REST API](#) для БД з іншою структурою. Налаштування серверної частини є досить складним та потребує використання відповідних технічних спеціалістів.

Для функціонування клієнтської частини на мобільний пристрій потрібно завантажити та встановити [Android apk](#) файл надавши йому потрібні дозволи. Встановлення [Android](#)-додатку є стандартною процедурою для користувачів мобільних пристроїв із операційною системою [Android](#) та не потребує спеціальних знань чи вмінь.

Розроблений додаток буде повністю сумісний із усіма пристроями починаючи із [SDK](#) 16, що відповідає версії 4.1.2 операційної системи [Android](#). На пристроях із більш ранім [SDK](#) додаток не працюватиме.

3.5. Інструкція користувача

Для авторизації у додатку користувачу потрібно буде ввести свій [email](#) та натиснути кнопку підтвердження. Після цього, система надішле йому на вказаний [email](#) тимчасовий пароль, який потрібно буде ввести у вікні, що відкриється. Повторна авторизація не буде потрібною поки користувач не змінить пристрій або не вийде із системи.

На рисунку 3.2 зображено вікно, яке дозволяє користувачу авторизацію в додатку.

Рис. 3.2. Вікно для авторизації в додатку

Після успішної авторизації, додаток сформує ваш персональний розклад, який буде відображатися в головному вікні програми. Для навігації по розкладу можна використовувати відповідні елементи меню, або свайп слайдер. Головне вікно та способи навігації у ньому зображено на рисунку 3.3.

Рис. 3.3. Основне вікно додатку

Крім того користувач завжди має змогу повернутися до поточної дати вибравши елемент навігації

Цитирования: 0,01%

id: 158

"Календар".

Якщо відкрито розклад для поточної дати, то натискання елемента навігації

Цитирования: 0,01%

id: 159

"Календар"

відкриє календар який дозволить перейти до розкладу для вибраної дати, що відображено на рисунку 3.4.

Рис. 3.4. Вікно

Цитирования: 0,01%

id: 160

"Календар"

Якщо користувач має права

Цитирования: 0,01%

id: 161

"Викладач",

то він зможе відкрити вікно для заповнення даних про відвідуваність натиснувши на потрібний предмет у розкладі, що відображено на рисунку 3.5.

Рис. 3.5. Перехід до вікна для заповнення даних про відвідуваність

У вікні для заповнення даних про відвідуваність, яке зображено на рисунку 3.6, викладач зможе ввести дані як за допомогою елементів меню, так і за допомогою елементів керування у списку студентів.

Рис. 3.6. Вікно для заповнення даних про відвідуваність

Вибравши зі списку студентів викладач зможе відправити їм повідомлення що відображено на рисунку 3.7

Рис. 3.7. Вікно для відправки повідомлення

В основному вікні знаходиться головне меню додатку, яке зображено на рисунку 3.8.

Рис. 3.8. Головне меню додатку

При виборі опції головного меню

Цитирования: 0,01%

id: 162

"Статистика"

перед користувачем відкриється вікно, в якому він зможе вибрати необхідні фільтри, що відображено на рисунку 3.9.

Рис. 3.9. Вікно для вибору фільтрів для статистики

Вибравши потрібні фільтри користувач зможе переглянути статистику натиснувши на відповідну кнопку. Вікно для перегляду статистики відображено на рисунку 3.10.

Рис. 3.10. Вікно із статистикою

При виборі опції головного меню

Цитирования: 0,01%

id: 163

"Повідомлення"

перед користувачем відкриється вікно, в якому він зможе переглянути отримані повідомлення, що відображено на рисунку 3.11.

Рис. 3.11. Вікно із повідомленнями

Опція меню

 Цитування: 0,01%	id: 164
"Допомога"	

доступна із будь якого вікна програми. У вікні допомоги яке зображено на рисунку 3.12 відображено повну інструкцію користування додатком.

Рис. 3.12. Вікно

 Цитування: 0,01%	id: 165
"Допомога"	

ВИСНОВКИ

У дипломній роботі було розглянуто розробку [Android](#)-додатка для обліку відвідування занять студентами закладів вищої освіти.

Результатом написання дипломної роботи є готовий програмний продукт, який являє собою [Android](#)-додаток для обліку відвідування занять студентами. Однією із основних переваг розробленої програмної системи є можливість інтеграції із веб-орієнтованим аналогом. Така інтеграція дозволить вирішити одну із основних проблем навчальних закладів - проблему контролю відвідування студентами занять.

Розроблений програмний продукт майже повністю автоматизує процес заповнення та перегляду даних про відвідуваність. Також, розроблена програмна система надає засоби для інформування студентів про проблеми із відвідуванням занять та дозволяє студентам стежити за власною статистикою відвідування занять. Крім того, додаток надає користувачу можливість перегляду свого персонального розкладу. За допомогою [Android](#)-додатка викладачі зможуть заповнювати журнал відвідування занять студентами безпосередньо на заняті та відправляти дані коли їм зручно. Відправлені дані одразу ж будуть оброблені на сервері та доступні для усіх учасників контролю процесу відвідування занять. Таким чином, виключаються зайві етапи збору даних про відвідуваність, що зменшує кількість помилок та виключає можливість недостовірності даних.

Важливою особливістю [Android](#)-додатка є те, що більша частина його функціоналу працює у режимі

 Цитування: 0,01%	id: 166
"Offline".	

Також, додаток надає викладачу можливість надсилання миттєвих повідомлень студентам, що дозволяє йому за потреби швидко передати студентам потрібну інформацію.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

[Android Media Playback \(EN\)](#). (<http://developer.android.com/guide/topics/media/mediaplayer.html>)

[Android SDK for Real-Time Apps | PubNub \(EN\)](#).
(<http://www.pubnub.com/docs/java/android/android-sdk.html>)

[C.J. DateAn Introduction to Database / C.J. DateAn // - Pearson](#) -2003. - 1024.

[Get the Android SDK. \(EN\)](#). (<http://developer.android.com/sdk/index.html?hl=sk>)

[Kathy Sierra Head First Java / Kathy Sierra // - O'Reilly Media](#) - 2005. - 688.

[Martin Fowler UML Distilled: A Brief Guide to the Standard Object Modeling Language/ Martin Fowler // - Addison-Wesley Professional](#) - 2003. - 208.

[Ryan Hodson Android Programming Succinctly / Ryan Hodson // - Syncfusion Inc..](#) - 2014. - 113.

[Subbu Allamaraju RESTful Web Services Cookbook / Subbu Allamaraju// O'Reilly Media / Yahoo Press.](#) - 2010. - 316.

Перспективи розробки під [Android \(EN\)](#).
(http://armikael.com/mobile/prospects_of_development_on_android.html)

Порівняльна характеристика Андроїд і [iOS](#) шляхом розгляду двох корпорацій-розробників.
(<http://ans-android.com/sopostavitelnaya-harakteristika-android-i-ios-putem-sravneniya-dvuh-korporatsiy-razrabotchikov-google-i-apple>)

Голощапов А. [Google Android](#): программирование для мобильных устройств / А. Голощапов. - СПб.: БХВ-Петербург, 2010. - 448 с.

Основи мови програмування [Java](#) [Електронний ресурс]. - Режим доступу: [URL: http://npk.edu.ua/3285-2](http://npk.edu.ua/3285-2). 130

Харди Б., Филлипс Б., Стюарт К., Марсикано К. [Android](#). Программирование для профессионалов. 2-е изд. - СПб.: Питер, 2016. - 640 с.

ДОДАТОК А

Глосарій

ТермінОпис термінуОсновні поняття та категорії предметної області та проекту
[Action Bar](#)Панель керування додатком, яка знаходиться в верхній частині екрану
[Activity](#)Аналог вікна у ОС [Android](#)
[Android](#)Найпопулярніша платформа та операційна система для мобільних пристроїв, яка створена на ядрі [Linux](#) й належить корпорації [Google Inc](#).
[APK](#)Інсталяційний файл програми для [Android](#), який містить байт-код програми, ресурси, маніфест
[AsyncTask](#)Клас, що забезпечує простий і зручний механізм для переміщення трудомістких операцій в фоновий потік.
[Google Cloud Messaging \(GCM\)](#)Сервіс для надсилання миттєвих нотифікацій до клієнта-додатка
[Android](#)
[JSON](#)Це текстовий формат обміну даними між комп'ютерами. [JSON](#) базується на тексті, і може бути з легкістю прочитаним людиною.
[Layout](#)Макет який визначає візуальну структуру користувацького

інтерфейсу [REST](#) Підхід до архітектури мережевих протоколів, які забезпечують доступ до інформаційних ресурсів [RESTful](#) Такий, що відповідає архітектурі

[REST](#) та підтримує її [Shared Preference](#): Спеціальний файл для конкретного додатку який зберігається в захищеній внутрішній пам'яті та дозволяє зберігати дані у форматі [key-value pair](#)

[Stored procedure](#) Об'єкт бази даних, який представляє собою набір [SQL](#) інструкцій, який компілюється один раз і зберігається на сервері [Toast](#) Спливаюче повідомлення в ОС [Android](#) [Trigger](#) Це [stored procedure](#) особливого типу, яку користувач не викликає явно, а використання якої обумовлено настанням визначеної події (дії) у реляційній базі даних [Нотифікація](#) Тип повідомлення в [Android](#)-додатку, яке відображається в системному рядку стану [Транзакція](#) Група послідовних операцій з базою даних, яка є логічною одиницею роботи з даними. Транзакція може бути виконана успішно, або не виконана зовсім. Користувачі системи [Викладач](#) Користувач, який матиме повний доступ до функціоналу додатку та можливість внесення даних до головної БД [Студент](#) Користувач, який матиме обмежений доступ до функціоналу додатку лише в інформаційних цілях [Вхідні та вихідні документи](#) Розклад занять [Документ](#), який є персональним графіком занять для викладача чи студента [Список](#) із даними про відвідуваність [Документ](#), який формується викладачем при проведенні занять та містить дані про відвідуваність [Список](#) із статистичними даними [Документ](#), який містить певні статистичні відомості [Список студентів](#) [Документ](#) із даними про студентів

ДОДАТОК Б

[CREATE TABLE Class](#) (

[ClassID](#) [INTEGER NOT NULL](#),

[Name](#) [VARCHAR](#)(20) [NULL](#),

[Course](#) [INTEGER NULL](#),

[Faculty](#) [VARCHAR](#)(100) [NULL](#)

);

[ALTER TABLE Class](#)

[ADD PRIMARY KEY](#) ([ClassID](#));

[CREATE TABLE ClassSchedule](#) (

[ClassID](#) [INTEGER NOT NULL](#), [ScheduleID](#) [INTEGER NOT NULL](#), [TeacherID](#) [INTEGER NOT NULL](#)

);

[ALTER TABLE ClassSchedule](#)

[ADD PRIMARY KEY](#) ([ClassID](#), [ScheduleID](#), [TeacherID](#));

[CREATE TABLE Lesson](#) (

[LessonID](#) [INTEGER NOT NULL](#), [ScheduleID](#) [INTEGER NOT NULL](#), [TeacherID](#) [INTEGER NOT NULL](#),

[Date&Time](#) [DATE NULL](#)

);

[ALTER TABLE Lesson](#)

[ADD PRIMARY KEY](#) ([LessonID](#), [ScheduleID](#), [TeacherID](#));

[CREATE TABLE Message](#) (

[MessageID](#) [INTEGER NOT NULL](#),

[DateOfCreation](#) [DATE NULL](#), [Name](#) [VARCHAR](#)(50) [NULL](#),

[Message](#) [VARCHAR](#)(100) [NULL](#),

[Type](#) [boolean NULL](#),

[UserID](#) [INTEGER NOT NULL](#), [TeacherID](#) [INTEGER NOT NULL](#), [StudentID](#) [INTEGER NOT NULL](#),

[Status](#) [VARCHAR](#)(20) [NULL](#),

[Priority](#) [INTEGER NULL](#)

);

[ALTER TABLE Message](#)

[ADD PRIMARY KEY](#) ([MessageID](#), [UserID](#), [TeacherID](#), [StudentID](#));

[CREATE TABLE Schedule](#) (

[ScheduleID](#) [INTEGER NOT NULL](#), [TeacherID](#) [INTEGER NOT NULL](#), [Name](#) [VARCHAR](#)(20) [NULL](#),

















[Type](#) [VARCHAR](#)(20) [NULL](#),

[Classroom](#) [VARCHAR](#)(20) [NULL](#), [DayOfWeek](#) [VARCHAR](#)(20) [NULL](#), [Parity](#) [VARCHAR](#)(20) [NULL](#),







[DateOfStart](#) [DATE NULL](#), [DateOfEnd](#) [DATE NULL](#), [Time](#) [DATE NULL](#),

[Durability](#) [DATE NULL](#)

```
);  
  
ALTER TABLE Schedule  
ADD PRIMARY KEY (ScheduleID,TeacherID);  
  
CREATE TABLE Student (  
StudentIDINTEGER NOT NULL, ClassIDINTEGER NOT NULL,  
NameVARCHAR(50) NULL,  
SurnameVARCHAR(50) NULL,  
EmailVARCHAR(100) NULL  
);  
  
ALTER TABLE Student  
ADD PRIMARY KEY (StudentID,ClassID); CREATE TABLE Teacher  
(  
TeacherIDINTEGER NOT NULL, NameVARCHAR(20) NULL,  
SurnameVARCHAR(20) NULL,  
EmailVARCHAR(20) NULL  
);  
  
ALTER TABLE Teacher  
ADD PRIMARY KEY (TeacherID);  
  
CREATE TABLE ThePresenceStatus (  
ThePresenceStatusID INTEGER NOT NULL, LessonIDINTEGER NOT NULL, ScheduleIDINTEGER NOT  
NULL, StudentIDINTEGER NOT NULL, ClassIDINTEGER NOT NULL, TeacherIDINTEGER NOT NULL,  
DatetimeDATE NULL,  
StatusVARCHAR(20) NULL,  
Additionl_Information VARCHAR(20) NULL, LateINTEGER NULL,  
ResonOfChangesVARCHAR(20) NULL  
);  
  
ALTER TABLE ThePresenceStatus  
ADDPRIAMRYKEY  
(ThePresenceStatusID,LessonID,ScheduleID,StudentID,ClassID,TeacherID);  
  
CREATE TABLE User (  
UserIDINTEGER NOT NULL,  
EmailVARCHAR(100) NULL, TeacherIDINTEGER NOT NULL, StudentIDINTEGER NOT NULL,  
GCM_IDVARCHAR(100) NULL,  
TemporaryPaswordVARCHAR(200) NULL  
);  
  
ALTER TABLE User  
ADD PRIMARY KEY (UserID,TeacherID,StudentID); ALTER TABLE ClassSchedule  
ADD FOREIGN KEY Has (ClassID) REFERENCES Class (ClassID);  
  
ALTER TABLE ClassSchedule  
ADDFOREIGNKEYR_22(ScheduleID,TeacherID)REFERENCESSchedule (ScheduleID, TeacherID);  
  
ALTER TABLE Lesson  
ADDFOREIGNKEYR_37(ScheduleID,TeacherID)REFERENCESSchedule (ScheduleID, TeacherID);  
  
ALTER TABLE Message  
ADD FOREIGN KEY R_20 (UserID, TeacherID, StudentID) REFERENCES User (UserID, TeacherID,  
StudentID);  
  
ALTER TABLE Schedule  
ADD FOREIGN KEY R_50 (TeacherID) REFERENCES Teacher (TeacherID); ALTER TABLE Student  
ADD FOREIGN KEY R_35 (ClassID) REFERENCES Class (ClassID);  
  
ALTER TABLE ThePresenceStatus  
ADD FOREIGN KEY R_45 (LessonID, ScheduleID, TeacherID) REFERENCES Lesson (LessonID,
```

ScheduleID, TeacherID);	
ALTER TABLE ThePresenceStatus	
ADD FOREIGN KEY R_46 (StudentID, ClassID) REFERENCES Student (StudentID, ClassID);	
ДОДАТОК В	
№DescriptionSteps to reproduceExpected resultActual resultStatusEnvironment1.Check logging with registered email1.Enter registered email	
2.Click	
	id: 167
System creates temporary password and send this password on entered email.System creates temporary password and send this password on entered email.PassedAndroid 4.2.12.Check logging with unregistered email1.Enter unregistered email	
2.Click	
	id: 168
System shows toast	
	id: 169
System shows toast for unregistered email.PassedAndroid 4.2.13.Check option	
	id: 170
1. Open main menu	
5.Click on option	
	id: 171
System deletes all user data. App is closed.System deletes all user data. App is closed.PassedAndroid 4.2.14.Check option	
	id: 172
with teacher permissions:1.Click on subject in scheduleAttendance window is openedAttendance window is openedPassedAndroid 4.2.15.Check option	
	id: 173
with student permissions:1.Click on subject in scheduleNothing happenedNothing happenedPassedAndroid 4.2.1	
6.Check option	
	id: 174
with entered messageClick on subject in schedule	
Select some students	
Click on option	
	id: 175
Enter some message	
Click	
	id: 176
Message is sent. System shows toast	
	id: 177
Message is sent. System shows toast	
	id: 178
PassedAndroid 4.2.17.Check option	
	id: 179
with not entered messageClick on subject in schedule	
Select some students	
Click on option	
	id: 180
Click	
	id: 181
System shows toast	
	

Цитирования: 0,02%	id: 182
System shows toast	
Цитирования: 0,02%	id: 183
"Enter message please".	
PassedAndroid d 4.2.18.Check option	
Цитирования: 0,01%	id: 184
"Send Message"	
without Internet connectionClick on subject in schedule	
Select some students	
Click on option	
Цитирования: 0,01%	id: 185
"Send Message"	
System shows toast	
Цитирования: 0,03%	id: 186
"Internet connection is not available"	
System shows toast	
Цитирования: 0,03%	id: 187
"Internet connection is not available"	
PassedAndroid d 4.2.19.Check option	
Цитирования: 0,01%	id: 188
"Get Statistic"	
Open main menu	
Click on option	
Цитирования: 0,01%	id: 189
"Statistic"	
3. Click on	
Цитирования: 0,01%	id: 190
"Get Statistic"	
Statistic is showedStatistic is showedPassedAndroid d 4.2.1	
10.Check option	
Цитирования: 0,01%	id: 191
"Get Statistic"	
without Internet connectionOpen main menu	
Click on option	
Цитирования: 0,01%	id: 192
"Statistic"	
System shows toast	
Цитирования: 0,03%	id: 193
"Internet connection is not available"	
System shows toast	
Цитирования: 0,03%	id: 194
"Internet connection is not available"	
PassedAndroid d 4.2.111.Check option	
Цитирования: 0,02%	id: 195
"Send attendance data"	
Click on subject in schedule	
Click on	
Цитирования: 0,01%	id: 196
"Send data"	
Data is sent.	
System shows toast	
Цитирования: 0,02%	id: 197
"Data is sent successfully"	
Data is sent.	
System shows toast	
Цитирования: 0,02%	id: 198
"Data is sent successfully"	
PassedAndroid d 4.2.112.Check option	
Цитирования: 0,02%	id: 199
"Send attendance data"	
without Internet connectionClick on subject in schedule	
Click on	
Цитирования: 0,01%	id: 200
"Send data"	
System shows toast	

 Цитирования: 0,04%	id: 201
"Internet connection is not available, please try later"	
System shows toast	
 Цитирования: 0,04%	id: 202
"Internet connection is not available, please try later"	
PassedAndroid 4.2.1	
ДОДАТОК Г	
AttendanceListAdapter.java	
<pre>package com.studentattendance;</pre>	
<pre>import java.util.ArrayList;</pre>	
<pre>import java.util.HashMap;</pre>	
<pre>import android.content.Context; import android.graphics.Color; import android.graphics.Typeface;</pre>	
<pre>import android.view.LayoutInflater; import android.view.View;</pre>	
<pre>import android.view.ViewGroup; import android.widget.BaseAdapter; import</pre>	
<pre>android.widget.Button; import android.widget.ListAdapter; import android.widget.TextView;</pre>	
<pre>public class AttendanceListAdapter extends BaseAdapter implements ListAdapter {</pre>	
<pre>private ArrayList HashMap String, String list = new</pre>	
<pre>ArrayList HashMap String, String ();</pre>	
<pre>private Context context;</pre>	
<pre>public AttendanceListAdapter(ArrayList HashMap String, String list, Context context) {</pre>	
<pre>this.list = list;</pre>	
<pre>this.context = context;</pre>	
<pre>}</pre>	
<pre>@Override</pre>	
<pre>public int getCount() {</pre>	
<pre>return list.size();</pre>	
<pre>}</pre>	
<pre>@Override</pre>	
<pre>public Object getItem(int pos) {</pre>	
<pre>return list.get(pos);</pre>	
<pre>}</pre>	
<pre>@Override</pre>	
<pre>public long getItemId(int pos) {</pre>	
<pre>return 0;</pre>	
<pre>}</pre>	
<pre>public void buttonsNotActive(Button presentBtn, Button absentBtn, Button lateBtn)</pre>	
<pre>{</pre>	
<pre>presentBtn.setBackgroundResource(R.drawable.ic_action_accept);</pre>	
<pre>absentBtn.setBackgroundResource(R.drawable.ic_action_cancel);</pre>	
<pre>lateBtn.setBackgroundResource(R.drawable.ic_late);</pre>	
<pre>}</pre>	
<pre>public void makePresentActive(Button presentBtn, Button absentBtn, Button lateBtn, int pos) {</pre>	
<pre>buttonsNotActive(presentBtn, absentBtn, lateBtn);</pre>	
<pre>presentBtn.setBackgroundResource(R.drawable.ic_action_accept_active); list.get(pos).put(</pre>	
 Цитирования: 0,01%	id: 203
"late",	
 Цитирования: 0%	id: 204
""	
);	
<pre>list.get(pos).put(</pre>	
 Цитирования: 0,01%	id: 205
"status",	
 Цитирования: 0,01%	id: 206
" "	
);	

```

}

public void makeAbsentActive(Button presentBtn, Button absentBtn, Button lateBtn, int pos) {

    buttonsNotActive(presentBtn, absentBtn, lateBtn);
    absentBtn.setBackgroundResource(R.drawable.ic_action_cancel_active); list.get(pos).put(
        "late",
    );
    list.get(pos).put(
        "status",
    );
    lateBtn.setBackgroundResource(R.drawable.ic_late_active);
}

public void makeLateActive(Button presentBtn, Button absentBtn, Button lateBtn,
int pos) {
    list.get(pos).put(
        "status",
    );
    makePresentActive(presentBtn, absentBtn, lateBtn, pos); list.get(pos).put(
        "late",
    );
    lateBtn.setBackgroundResource(R.drawable.ic_late_active);
}

public void loadData() {
}

public void saveData() {
}

@Override

public View getView(final int position, View convertView, ViewGroup parent) { View view =
convertView;

if (view == null) {

    LayoutInflater inflater = (LayoutInflater)
context.getSystemService(Context.LAYOUT_INFLATER_SERVICE); view =
inflater.inflate(R.layout.attendance_list_item, null);
}

if (list.get(position).get(
    "select"
).equals(
    " "
)) view.setBackgroundColor(Color.parseColor(
    "#201212"
));

else

view.setBackgroundColor(Color.TRANSPARENT);

// Handle TextView and display string from your list
Typeface myTypeface = Typeface.createFromAsset(context.getAssets(),
    "fonts/segoe_print_bold.ttf"
);

TextView stdName = (TextView) view.findViewById(R.id.stdName);

```

```

stdName.setTypeface(myTypeface);

stdName.setText((String) (list.get(position)).get(
Цитирования: 0,01% id: 219
"stdName"
));

// TextView className = (TextView) view.findViewById(R.id.className);

// className.setTypeface(myTypeface);

// className.setText((String) (list.get(position)).get(
Цитирования: 0,01% id: 220
"className"
));

// Handle buttons and add onClickListeners

final Button presentBtn = (Button) view.findViewById(R.id.present_btn); final Button absentBtn =
(Button) view.findViewById(R.id.absent_btn); final Button lateBtn = (Button)
view.findViewById(R.id.late_btn);

if (list.get(position).get(
Цитирования: 0,01% id: 221
"status"
).equals(
Цитирования: 0,01% id: 222
" "
) && list.get(position).get(
Цитирования: 0,01% id: 223
"late"
).equals(
Цитирования: 0% id: 224
""
))

makePresentActive(presentBtn, absentBtn, lateBtn, position);

else if (list.get(position).get(
Цитирования: 0,01% id: 225
"status"
).equals(
Цитирования: 0,01% id: 226
" "
)) makeAbsentActive(presentBtn, absentBtn, lateBtn, position);

else if (!list.get(position).get(
Цитирования: 0,01% id: 227
"late"
).equals(
Цитирования: 0% id: 228
""
)) makeLateActive(presentBtn, absentBtn, lateBtn, position);

presentBtn.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

makePresentActive(presentBtn, absentBtn, lateBtn, position); notifyDataSetChanged();

}

});

absentBtn.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

makeAbsentActive(presentBtn, absentBtn, lateBtn, position); notifyDataSetChanged();

}

});

lateBtn.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

makeLateActive(presentBtn, absentBtn, lateBtn, position); notifyDataSetChanged();

}
}

```

```

});

return view;

}

}

AttendanceTaker.java

package com.studentattendance;

import android.app.ActionBar; import android.app.Activity; import android.app.ProgressDialog;
import android.content.Context; import android.content.Intent;

import android.content.SharedPreferences;

import android.content.SharedPreferences.Editor;

import android.graphics.Color;

import android.graphics.drawable.ColorDrawable;

import android.os.AsyncTask; import android.os.Bundle; import android.text.Html; import
android.util.Log; import android.view.Menu;

import android.view.MenuInflater; import android.view.MenuItem; import android.view.View;

import android.view.Window;

import android.widget.AdapterView; import android.widget.Button; import
android.widget.ListView; import android.widget.Toast;

import com.google.gson.Gson;

import com.google.gson.reflect.TypeToken;

import org.json.JSONArray; import org.json.JSONException; import org.json.JSONObject;

import java.io.IOException; import java.util.ArrayList; import java.util.HashMap;

import okhttp3.FormBody;

import okhttp3.RequestBody;

public class AttendanceTaker extends Activity { SharedPreferences sPref;

private ProgressDialog pDialog;

JSONObject jsonSchedule; Intent intent;

String id, idOfSubject;

ArrayList HashMap String, String listAttendance; AttendanceListAdapter atd;

Boolean allSelected = false; ListView lView;

Button Ok;

@Override

protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState);
getWindow().requestFeature(Window.FEATURE_ACTION_BAR);
setContentView(R.layout.attendance_taker);

ActionBar actionBar = getActionBar(); actionBar.setBackgroundDrawable(new
ColorDrawable(Color.parseColor(
Цитирования: 0,01% id: 229
"# 71a3a"
))); actionBar.setTitle(Html.fromHtml(
Цитирования: 0,01% id: 230
" font_color='#0000748' /font "
)); actionBar.show();

sPref = this.getSharedPreferences(
Цитирования: 0,01% id: 231
"com.studentattendance",
Context.MODE_PRIVATE);

intent = getIntent();

id = intent.getStringExtra(
Цитирования: 0,01% id: 232
"id"
);

idOfSubject = id.substring(0, id.indexOf(' ')); listAttendance = new ArrayList HashMap String,
String ();

Ok = (Button) findViewById(R.id.ok); Ok.setTextColor(Color.parseColor(

```

```

Цитирования: 0,01% id: 233
"# 71"

));

// handle listview and assign adapter

lView = (ListView) findViewById(R.id.attList);

new LoadAttendanceList().execute();

lView.setOnItemClickListener(new AdapterView.OnItemClickListener() {

@Override

public void onItemClick(AdapterView ? parent, View view, int position,

long id) {

if (listAttendance.get(position).get(

Цитирования: 0,01% id: 234
"selected"

).equals(

Цитирования: 0,01% id: 235
" "

)) listAttendance.get(position).put(

Цитирования: 0,01% id: 236
"selected",

Цитирования: 0,01% id: 237
" "

);

else

listAttendance.get(position).put(

Цитирования: 0,01% id: 238
"selected",

Цитирования: 0,01% id: 239
" "

); atd.notifyDataSetChanged();

Log.e(

Цитирования: 0,01% id: 240
"Click",

(String) listAttendance.get(position).get(

Цитирования: 0,01% id: 241
"status"

));

Log.e(

Цитирования: 0,01% id: 242
"Selected",

(String) listAttendance.get(position).get(

Цитирования: 0,01% id: 243
"selected"

));

}

});

}

public void onOKStart(View v) {

if (!listAttendance.isEmpty()) {

String jsonAttendanceList = new Gson().toJson(listAttendance); sPref =

this.getSharedPreferences(

Цитирования: 0,01% id: 244
"com.studentattendance",

Context.MODE_PRIVATE);

Editor ed = sPref.edit(); ed.putString(id, jsonAttendanceList); ed.commit();

if (new InternetConnectionChecker(this).isOnline()) {

new SendAttendance().execute();

} else {

Toast.makeText(this,

Цитирования: 0,07% id: 245

```

```

"Дані збережені, але не надіслані! Будь ласка повторіть надсилання із підключенням до
Інтернету!",

Toast.LENGTH_LONG).show();

}

} else

Toast.makeText(this,
❏ Цитування: 0,02% id: 246
"Щось погане трапилось :(",

Toast.LENGTH_LONG).show();

}

public void onBackPressed() {

if (!listAttendance.isEmpty()) {

String jsonAttendanceList = new Gson().toJson(listAttendance);

sPref = this.getSharedPreferences(
❏ Цитування: 0,01% id: 247
"com.studentattendance",
Context.MODE_PRIVATE);

Editor ed = sPref.edit(); ed.putString(id, jsonAttendanceList); ed.commit();

Toast.makeText(this,
❏ Цитування: 0,03% id: 248
"Дані збережено, не забудьте їх надіслати!",
Toast.LENGTH_LONG).show();

}

finish();

}

@Override

public boolean onCreateOptionsMenu(Menu menu) { MenuInflater inflater = getMenuInflater();
inflater.inflate(R.menu.attendance_taker_actions, menu); return
super.onCreateOptionsMenu(menu);

}

@Override

public boolean onOptionsItemSelected(int featureId, MenuItem item) {

int itemId = item.getItemId();

switch (itemId) {

case R.id.selectAll:

if (!allSelected) {

for (int i = 0; i listAttendance.size(); i++) listAttendance.get().put(
❏ Цитування: 0,01% id: 249
"selected",
❏ Цитування: 0,01% id: 250
" ");

});

allSelected = true;

} else {

for (int i = 0; i listAttendance.size(); i++) listAttendance.get().put(
❏ Цитування: 0,01% id: 251
"selected",
❏ Цитування: 0,01% id: 252
" ");

});

allSelected = false;

}

atd.notifyDataSetChanged();

break;

```

```

case R.id.AllPresent:

for (int i = 0; i < listAttendance.size(); i++)

if (listAttendance.get(i).get(
Цитирования: 0,01% id: 253
"selected"
) ==
Цитирования: 0,01% id: 254
" ")
listAttendance.get(i).put(
Цитирования: 0,01% id: 255
"status",
Цитирования: 0,01% id: 256
" ");
);

atd.notifyDataSetChanged();

break;

case R.id.AllAbsent:

for (int i = 0; i < listAttendance.size(); i++)

if (listAttendance.get(i).get(
Цитирования: 0,01% id: 257
"selected"
) ==
Цитирования: 0,01% id: 258
" ")
listAttendance.get(i).put(
Цитирования: 0,01% id: 259
"status",
Цитирования: 0,01% id: 260
" ");
);

atd.notifyDataSetChanged();

break;

case R.id.action_help:

Intent helpActivity = new Intent(getApplicationContext(), HelpActivity.class);

startActivity(helpActivity);

break;

case R.id.action_send_messages:

Intent messageActivity = new Intent(getApplicationContext(), SendMessageActivity.class);

startActivity(messageActivity);

break;

}

return true;

}

class LoadAttendanceList extends AsyncTask<String, String, String> {

@Override

protected void onPreExecute() {

super.onPreExecute();

pDialog = new ProgressDialog(AttendanceTaker.this); pDialog.setMessage(
Цитирования: 0,03% id: 261
"Завантаження даних, зачекайте будь ласка...")
); pDialog.setIndeterminate(false);

pDialog.setCancelable(false); pDialog.show();

}

protected String doInBackground(String... args) {

if (sPref.contains(id)) { Log.e(
Цитирования: 0,01% id: 262
"Е:",
Цитирования: 0,01% id: 263

```


"Ss";	
);	
Gson gson = new Gson();	
listAttendance = gson.fromJson(sPref.getString(id.toString(),	
Цитирования: 0%	id: 264
""	
), new TypeToken<ArrayList<HashMap<String, String>>() {	
}.getType());	
} else {	
Gson gson = new Gson();	
ArrayList<HashMap<String, String>> classSubjects = new	
ArrayList<HashMap<String, String>>();	
JSONArray studentsList = new JSONArray(); String classID = null;	
classSubjects = gson.fromJson(sPref.getString(
Цитирования: 0,01%	id: 265
"ClassSubjects",	
Цитирования: 0%	id: 266
""	
), new TypeToken<ArrayList<HashMap<String, String>>() {	
}.getType());	
try {	
studentsList = new JSONArray(sPref.getString(
Цитирования: 0,01%	id: 267
"StudentList",	
Цитирования: 0%	id: 268
""	
));	
for (int i = 0; i < classSubjects.size(); i++) { Log.e(
Цитирования: 0,01%	id: 269
"QQQQ:",	
classSubjects.get(i).get(
Цитирования: 0,01%	id: 270
"SubjectID"	
).toString());	
Log.e(
Цитирования: 0,01%	id: 271
"QQQQ:",	
idOfSubject);	
if	
(idOfSubject.equals(classSubjects.get(i).get(
Цитирования: 0,01%	id: 272
"SubjectID"	
).toString())) { Log.e(
Цитирования: 0,01%	id: 273
"E:",	
Цитирования: 0,01%	id: 274
"Ss";	
);	
classID = classSubjects.get(i).get(
Цитирования: 0,01%	id: 275
"ClassID"	
).toString();	
}	
}	
if (classID != null)	
for (int i = 0; i < studentsList.length(); i++) { Log.e(
Цитирования: 0,01%	id: 276
"E:",	
Цитирования: 0,01%	id: 277
"Ss";	

```

);

JSONObject classes = studentsList.getJSONObject(); JSONArray students =
classes.getJSONArray(classID); for (int j = 0; j < students.length(); j++) {

JSONObject subject = students.getJSONObject();

String id = subject.getString(
    "id"
);

String name = subject.getString(
    "Surname"
).concat(
    "id: 278"
);

.concat(subject.getString(
    "Name"
)); String email = subject.getString(
    "Email"
);

String ();

HashMap<String, String> map = new HashMap<String,
map.put(
    "id",
id); map.put(
    "classID",
classID); map.put(
    "stuName",
name);

map.put(
    "email",
email);

map.put(
    "status",
    " "
);

map.put(
    "selecter",
    " "
);











map.put(
    "late",
    ""
);

listAttendance.add(map);
}

} catch (JSONException e) {

Log.e(
    "E:",
    Log.getStackTraceString(e));

```

}	
}	
return null;	
}	
protected void onPostExecute(String file_url) {	
pDialog.dismiss();	
if (!listAttendance.isEmpty()) {	
atd = new AttendanceListAdapter(listAttendance, AttendanceTaker.this);	
iView.setAdapter(ato);	
}	
}	
}	
}	
class SendAttendance extends AsyncTask<String, String, String> {	
@Override	
protected void onPreExecute() {	
super.onPreExecute();	
pDialog = new ProgressDialog(AttendanceTaker.this); pDialog.setMessage(
 Цитирования: 0,03%	id: 294
"Завантаження даних, зачекайте будь ласка..."	
); pDialog.setIndeterminate(false);	
pDialog.setCancelable(false); pDialog.show();	
}	
protected String doInBackground(String... args) {	
String attendanceData = sPref.getString(id.toString(),	
 Цитирования: 0%	id: 295
""	
).replace("\n", "	
 Цитирования: 0,01%	id: 296
");	
Log.e("	
AttData:	
 Цитирования: 0,03%	id: 297
", attendanceData); String[] lessonData = id.split(" "	
 Цитирования: 0,04%	id: 298
"); JSONObject jsonParser = new JSONObject (); String url_checkIfExist =	
"	
http://192.168.0.103/StudentAttendanceWSTest/pushAttendance.php	
 Цитирования: 0,04%	id: 299
"; RequestBody formBody = new FormBody.Builder().add(" "	
SubjectID	
 Цитирования: 0,02%	id: 300
",	
lessonData[0])	
.add(" "	
DateAndTime	
 Цитирования: 0,01%	id: 301
", lessonData[2]).add(" "	
AttendanceData	
 Цитирования: 0,1%	id: 302
", attendanceData).build();	
JSONObject json = null; try {	
json = jsonParser.postRequest(url_checkIfExist, formBody);	
} catch (IOException e) {	
Log.e(" "	
E:	
 Цитирования: 0,05%	id: 303
", Log.getStackTraceString(e));	

```

}

try {

    int success = json.getInt("
success
    Цитирования: 0,05% id: 304
");

    if (success == 1) {

        // Toast.makeText(attendanceTaker.this, "
Data is send
// successfully..)
        Цитирования: 0,03% id: 305
        Toast.LENGTH_SHORT).show();

        Log.i("
:
        Цитирования: 0,01% id: 306
        json.getString("
message
        Цитирования: 0,03% id: 307
        "));

        } else {

            Toast.makeText(attendanceTaker.this, "
Проблема із підключенням до сервера.:( Спробуйте пізніше будь ласка!
        Цитирования: 0,03% id: 308
        ",
        Toast.LENGTH_SHORT).show(); Log.e("
:
        Цитирования: 0,01% id: 309
        json.getString("
message
        Цитирования: 0,03% id: 310
        "));

        }

        } catch (JSONException e) {

            Log.e("
E: ", Log.getStackTraceString(e));

        }

        return null;

    }

    protected void onPostExecute(String file_url) { progressDialog.dismiss();

    }

    }

    }

    package com.studentattendance;

    import java.io.IOException;

    import org.json.JSONException;

    import org.json.JSONObject;

    import android.app.ProgressDialog; import android.content.Context; import android.os.AsyncTask;
    import android.util.Log;

    import okhttp3.FormBody;

    import okhttp3.RequestBody;

    public class

    extends AsyncTask<String, String, String> {

        private StatisticTypeActivity activity; private Context context;

        private ProgressDialog progressDialog;

        private String subjectsAndGroups, result;

        public GetStatisticData(Context c, StatisticTypeActivity a, String subjectsAndGroups) {

```

<code>context = c;</code>	
<code>activity = a;</code>	
<code>this.subjectsAndGroups = subjectsAndGroups;</code>	
<code>context.getSharedPreferences("com.studentattendance</code>	
<code>Цитирования: 0,07%</code>	id: 311
<code>Context.MODE_PRIVATE);</code>	
<code>}</code>	
<code>@Override</code>	
<code>protected void onPreExecute() {</code>	
<code>super.onPreExecute();</code>	
<code>pDialog = new ProgressDialog(context); pDialog.setMessage("Завантаження даних, зачекайте будь ласка...</code>	
<code>Цитирования: 0,12%</code>	id: 312
<code>"); pDialog.setIndeterminate(false);</code>	
<code>pDialog.setCancelable(false); pDialog.show();</code>	
<code>}</code>	
<code>protected String doInBackground(String... args) { JSONObject jsonParser = new JSONObject();</code>	
<code>String uri_checkIfExists =</code>	
<code>"</code>	
<code>http://192.168.0.103/StudentAttendanceWSTest/getStatisticForTeacher.php</code>	
<code>Цитирования: 0,03%</code>	id: 313
<code>"; RequestBody formBody = new</code>	
<code>RequestBody.Builder().add("</code>	
<code>SubjectsAndGroupsList</code>	
<code>Цитирования: 0,1%</code>	id: 314
<code>", subjectsAndGroups).build(); JSONObject json = null;</code>	
<code>try {</code>	
<code>json = jsonParser.postRequest(uri_checkIfExists, formBody);</code>	
<code>} catch (IOException e) {</code>	
<code>Log.e("E:</code>	
<code>Цитирования: 0,05%</code>	id: 315
<code>" Log.printStackTrace();</code>	
<code>}</code>	
<code>try {</code>	
<code>int success = json.getInt("success</code>	
<code>success</code>	
<code>Цитирования: 0,04%</code>	id: 316
<code>");</code>	
<code>if (success == 1) {</code>	
<code>result = json.getString("stat</code>	
<code>Цитирования: 0,01%</code>	id: 317
<code>"); Log.e(":</code>	
<code>:</code>	
<code>Цитирования: 0,01%</code>	id: 318
<code>" json.getString("message</code>	
<code>Цитирования: 0,02%</code>	id: 319
<code>"));</code>	
<code>} else {</code>	
<code>Log.e(":</code>	
<code>:</code>	
<code>Цитирования: 0,01%</code>	id: 320
<code>" json.getString("message</code>	
<code>Цитирования: 0,03%</code>	id: 321
<code>"));</code>	
<code>}</code>	
<code>} catch (JSONException e) {</code>	

```
Log.e("
Er:", Log.getStackTraceString(e));
}
return null;
}

protected void onPostExecute(String file_url) { activity.CallActivityForSelectedFilters(result);
pDialog.dismiss();
}
}

HelpActivity.java

package com.studentattendance;

import android.app.ActionBar; import android.app.Activity; import android.graphics.Color; import
android.graphics.Typeface;

import android.graphics.drawable.ColorDrawable;

import android.os.Bundle; import android.text.Html; import android.view.Menu;

import android.view.MenuItem; import android.view.Window;
import android.widget.TextView;

public class HelpActivity extends Activity {

@Override

protected void onCreate(Bundle savedInstanceState) {
getWindow().requestFeature(Window.FEATURE_ACTION_BAR);
super.onCreate(savedInstanceState); setContentView(R.layout.activity_help);

ActionBar actionBar = getActionBar(); actionBar.setBackgroundDrawable(new

ColorDrawable(Color.parseColor("#a71aca

Цитирования: 0,02% id: 322
")); actionBar.setTitle(Html.fromHtml("

font

color='#0bf748' Допомога /font

Цитирования: 0,05% id: 323
")); actionBar.show();

Typeface tp = Typeface.createFromAsset(getAssets(),

"

fonts/segoe_print_bold.ttf");

TextView mainWindowTV=(TextView) findViewById(R.id.mainWindowTV);
mainWindowTV.setTypeface(tp);

TextView mainWindowText1=(TextView) findViewById(R.id.mainWindowText1);
mainWindowText1.setTypeface(tp); TextView mainWindowText2=(TextView)

findViewById(R.id.mainWindowText2); mainWindowText2.setTypeface(tp);
}

@Override

public boolean onCreateOptionsMenu(Menu menu) { MenuItem inflater = getMenuInflater();
inflater.inflate(R.menu.help_activity_actions, menu); return super.onCreateOptionsMenu(menu);
}

@Override

public boolean onOptionsItemSelected(int featureId, MenuItem item) {

int itemId = item.getItemId();

switch (itemId) {

case R.id.action_back: finish();

break;

}

return true;

}

}

}

localScheduleLoaderActivity.java
```

```
package com.studentattendance;

import java.util.ArrayList;

import java.util.HashMap;

import android.app.Activity;

public class ILocalScheduleLoaderActivity extends Activity { ArrayList HashMap String, String
scheduleList;

void getScheduleData(ArrayList HashMap String, String scheduleList) {

if (!scheduleList.isEmpty()) {

this.scheduleList = scheduleList;

}

}

}

InternetConnectionChecker.java

package com.studentattendance;

import android.content.Context;

import android.net.ConnectivityManager;

import android.net.NetworkInfo;

public class InternetConnectionChecker { Context _context;

public InternetConnectionChecker(Context context) {

_context = context;

}

public boolean isOnline() {

ConnectivityManager cm = (ConnectivityManager)

_context.getSystemService(Context.CONNECTIVITY_SERVICE); NetworkInfo netInfo =

cm.getActiveNetworkInfo();

return netInfo != null && netInfo.isConnectedOrConnecting();

}

}

JSONParser.java

package com.studentattendance;

import java.io.IOException;

import java.io.InputStream;

import java.io.UnsupportedEncodingException;

import org.json.JSONException;

import org.json.JSONObject;

import android.util.Log; import okhttp3.MediaType; import okhttp3.OkHttpClient; import

okhttp3.Request; import okhttp3.RequestBody; import okhttp3.Response;

public class JSONParser {

public static final MediaType JSON = MediaType.parse("application/json; charset=utf-8");

OkHttpClient client = new OkHttpClient();

static InputStream is = null; static JSONObject jObj = null; static String jsonAns = null;

JSONObject postRequest(String url, RequestBody json) throws IOException {

try {

Request request = new Request.Builder().url(url).post(json).build(); Response response =

client.newCall(request).execute();

jsonAns = response.body().string();

} catch (UnsupportedEncodingException e) { Log.e("Er:

", Log.getStackTraceString( ));

} catch (IOException e) {
```

```
Log.i("
Er:
Цитирования: 0,05% id: 325
", Log.getStackTraceString( ));
} catch (Exception e) {
Log.i("
Er:
Цитирования: 0,03% id: 326
", Log.getStackTraceString( ));
}
Log.i("
var -
Цитирования: 0,11% id: 327
", JsonAns);
// try parse the string to a JSON object
try {
JSONObject = new JSONObject(JsonAns);
} catch (JSONException e) {
Log.i("
JSON Parser", "Error parsing data " + e.toString());
}
// return JSON String
return jsonObj;
}
}

LocalGroupListLoader.java
package com.studentattendance;

import java.util.ArrayList;
import java.util.HashMap;
import com.google.gson.Gson;
import com.google.gson.reflect.TypeToken;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.SharedPreferences;
import android.os.AsyncTask;

public class LocalGroupListLoader extends AsyncTask<String, String, String> {
private StatisticTypeActivity activity; private Context context;
private ProgressDialog pDialog; private SharedPreferences sPref;
private ArrayList<HashMap<String, String>> groupeList = new
ArrayList<HashMap<String, String>> ();
private String subjectId; private String IdOrNames;

public LocalGroupListLoader(Context c, StatisticTypeActivity a, String subjectId, String
IdOrNames) {
context = c;
activity = a; this.subjectId = subjectId; this.IdOrNames = IdOrNames;
sPref = context.getSharedPreferences(
Цитирования: 0,01% id: 328
"com.studentattendance",
Context.MODE_PRIVATE);
}

@Override
protected void onPreExecute() {
super.onPreExecute();
```



```

pDialog = new ProgressDialog(context); pDialog.setMessage(
    Цитирования: 0,03%
    id: 329
    "Завантаження даних, зачекайте будь ласка..."
); pDialog.setIndeterminate(false);

pDialog.setCancelable(false); pDialog.show();

}

protected String doInBackground(String... args) { Gson gson = new Gson();

ArrayList<HashMap<String, String>> classSubjects = new
ArrayList<HashMap<String, String>> ();

classSubjects = gson.fromJson(sPref.getString(
    Цитирования: 0,01%
    id: 330
    "classSubject",
    Цитирования: 0%
    id: 331
    ""
), new TypeToken<ArrayList<HashMap<String, String>>>() {
}.getType());

if (subjectId.equals(
    Цитирования: 0,01%
    id: 332
    "All Subjects"
)) {

for (int i = 0; i < classSubjects.size(); i++) { HashMap<String, String> hm = new HashMap<String,
String> (); hm.put(
    Цитирования: 0,01%
    id: 333
    "Name",
classSubjects.get(i).get(
    Цитирования: 0,01%
    id: 334
    "Name"
).toString());

hm.put(
    Цитирования: 0,01%
    id: 335
    "classID",
classSubjects.get(i).get(
    Цитирования: 0,01%
    id: 336
    "classID"
).toString()); hm.put(
    Цитирования: 0,01%
    id: 337
    "subjectID",

classSubjects.get(i).get(
    Цитирования: 0,01%
    id: 338
    "subjectID"
).toString());

groupeList.add(hm);

}

} else {

for (int i = 0; i < classSubjects.size(); i++) {

if

(subjectId.equals(classSubjects.get(i).get(
    Цитирования: 0,01%
    id: 339
    "subjectID"
).toString())) { HashMap<String, String> hm = new HashMap<String, String> (); hm.put(
    Цитирования: 0,01%
    id: 340
    "Name",
classSubjects.get(i).get(
    Цитирования: 0,01%
    id: 341
    "Name"
).toString()); hm.put(
    Цитирования: 0,01%
    id: 342
    "classID",

classSubjects.get(i).get(
    Цитирования: 0,01%
    id: 343
    "classID"
).toString()); hm.put(
    Цитирования: 0,01%
    id: 344
    "subjectID",

```

```

"Цитирования: 0,01%" id: 344
"Subject",

classSubjects.get().get(
"Цитирования: 0,01%" id: 345
"Subject"
).toString());

groupeList.add(hm);
}
}
}

return null;
}

protected void onPostExecute(String file_url) {

pDialog.dismiss();

if (!(idOrNames.equals(
"Цитирования: 0,01%" id: 346
"Names"
))) {

activity.setGroupeSpinnerData(groupeList);

}

}

}

package com.studentattendance;

import android.app.Activity;

import android.app.AlertDialog;

import android.content.BroadcastReceiver;

import android.content.Context; import android.content.Intent; import android.content.IntentFilter;

import android.content.SharedPreferences;

import android.content.SharedPreferences.Editor;

import android.os.AsyncTask; import android.os.Bundle; import android.os.Handler;

import android.support.v4.content.LocalBroadcastManager;

import android.util.Log; import android.view.Gravity; import android.widget.Toast;

import com.google.android.gms.common.ConnectionResult;

import com.google.android.gms.common.GoogleApiAvailability;

import org.json.JSONException;

import org.json.JSONObject;

import java.io.IOException;

import okhttp3.FormBody;

import okhttp3.RequestBody;

public class MainActivity extends Activity { SharedPreferences sPref;

Toast toast;

private ProgressDialog pDialog; JSONParser jsonParser = new JSONParser();

private static final String TAG_SUCCESS =
"Цитирования: 0,01%" id: 347
"success"
; private static String url_GetSchedule =

"Цитирования: 0,03%" id: 348
"http://192.168.0.103/StudentAttendanceWSTest/getSchedule.php"
; private static String url_GetStudents =

"Цитирования: 0,03%" id: 349
"http://192.168.0.103/StudentAttendanceWSTest/getStudentList.php"
; private static String url_GetClassSubjects =

```

```

Цитирования: 0,03% id: 350
"ip://192.168.0.103/StudentAttendanceWSTest/getClass_subjects.php"

; String UserID;

String UserType;

Handler handler = new Handler();

private String TAG = MainActivity.class.getSimpleName();

private static final int PLAY_SERVICES_RESOLUTION_REQUEST = 9000;

private BroadcastReceiver mRegistrationBroadcastReceiver;

@Override

protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);

sPref = this.getSharedPreferences(
Цитирования: 0,01% id: 351
"com.studentattendance",
Context.MODE_PRIVATE);

final Intent myIntent = new Intent(this, LoginActivity.class);

if (!sPref.contains(
Цитирования: 0,01% id: 352
"UserID"
))

handler.postDelayed(new Runnable() {

public void run() { startActivity(myIntent); finish();

}

}, 2000);

else {

if (!sPref.contains(
Цитирования: 0,01% id: 353
"Schedule"
))

new GetDataForSchedule().execute(); activitySelector();

}

mRegistrationBroadcastReceiver = new BroadcastReceiver() {

@Override

public void onReceive(Context context, Intent intent) {

// checking for type intent filter

if (intent.getAction().equals(Config.REGISTRATION_COMPLETE)) {

// gcm successfully registered

// now subscribe to `global` topic to receive app wide
notifications

String token = intent.getStringExtra(
Цитирования: 0,01% id: 354
"token"
);

Toast.makeText(getApplicationContext(),
Цитирования: 0,02% id: 355
"GCM registration token: "
+ token, Toast.LENGTH_LONG).show();

} else if (intent.getAction().equals(Config.SENT_TOKEN_TO_SERVER)) {

// gcm registration id is stored in our server's MySQL

Toast.makeText(getApplicationContext(),
Цитирования: 0,04% id: 356
"GCM registration token is stored in server!",
Toast.LENGTH_LONG).show();

} else if (intent.getAction().equals(Config.PUSH_NOTIFICATION)) {

// new push notification is received

Toast.makeText(getApplicationContext(),

```

```

    Цитирования: 0,02% id: 357
    "Роль учителя в классе!",
    Toast.LENGTH_LONG).show();

}

};

if (checkPlayServices()) { registerGCM();
}

}

public void onBackPressed() {

if (toast != null) toast.cancel();

System.exit(0);

}

void activitySelector() {

if ((sPref.getString(
    Цитирования: 0,01% id: 358
    "UserType",
    Цитирования: 0% id: 359
    ""
)).equals(
    Цитирования: 0,01% id: 360
    ""
)) {

final Intent myIntent = new Intent(this, TeacherMainScheduleActivity.class);

handler.postDelayed(new Runnable() {

public void run() { startActivity(myIntent); finish();

}

}, 1000);

} else if ((sPref.getString(
    Цитирования: 0,01% id: 361
    "UserType",
    Цитирования: 0% id: 362
    ""
)).equals(
    Цитирования: 0,01% id: 363
    ""
)) {

final Intent myIntent = new Intent(this, StudentMainScheduleActivity.class);

handler.postDelayed(new Runnable() {

public void run() { startActivity(myIntent); finish();

}

}, 1000);

}

}

class GetDataForSchedule extends AsyncTask<String, String, String> {

@Override

protected void onPreExecute() {

super.onPreExecute();

pDialog = new ProgressDialog(MainActivity.this); pDialog.setMessage(
    Цитирования: 0,01% id: 364
    "Loading data..."
); pDialog.setIndeterminate(false); pDialog.setCancelable(true);

pDialog.show();

}

protected String doInBackground(String... args) { UserID = (sPref.getString(
    Цитирования: 0,01% id: 365

```

"UserId",	
Цитирования: 0%	id: 366
"); UserType = (sPref.getString(
Цитирования: 0,01%	id: 367
"UserType",	
Цитирования: 0%	id: 368
");	
RequestBody formBodyToGetScedule = new FormBody.Builder().add(
Цитирования: 0,01%	id: 369
"ID",	
UserID).add(
Цитирования: 0,01%	id: 370
"UserType",	
UserType)	
.build();	
JSONObject scheduleJson = null; try {	
scheduleJson = jsonParser.postRequest(url_GetSchedule,	
formBodyToGetScedule);	
} catch (IOException e) {	
Log.e(
Цитирования: 0,01%	id: 371
"E:",	
Log.getStackTraceString(e));	
}	
try {	
int success = scheduleJson.getInt(TAG_SUCCESS);	
if (success == 1) {	
final String Schedule = scheduleJson.getString(
Цитирования: 0,01%	id: 372
"Schedule"	
);	
sPref = MainActivity.this.getSharedPreferences(
Цитирования: 0,01%	id: 373
"com.studentattendance",	
Context.MODE_PRIVATE);	
Editor ed = sPref.edit(); ed.putString(
Цитирования: 0,01%	id: 374
"Schedule",	
Schedule); ed.commit();	
} else {	
final String message = scheduleJson.getString(
Цитирования: 0,01%	id: 375
"message"	
); MainActivity.this.runOnUiThread(new Runnable() {	
public void run() {	
toast = Toast.makeText(MainActivity.this, message,	
Toast.LENGTH_LONG);	
}	
});	
toast.setGravity(Gravity.CENTER, 0, 0);	
toast.show();	
Log.e(
Цитирования: 0,01%	id: 376
".",	
scheduleJson.getString(
Цитирования: 0,01%	id: 377
"message"	

```

));
}

} catch (JSONException e) {

Log.e(
❏ Цитирования: 0,01% id: 378
"e:",
Log.getStackTraceString(e));
}

if (UserType.equals(
❏ Цитирования: 0,01% id: 379
" "
)) {

RequestBody formBodyToGetStudentList = new

FormBody.Builder().add(
❏ Цитирования: 0,01% id: 380
"ID",
UserID).build(); JSONObject jsonStudentList = null; JSONObject class_subjects = null; try {

jsonStudentList = jsonParser.postRequest(url_GetStudents, formBodyToGetStudentList);

class_subjects = jsonParser.postRequest(url_GetClassSubjects,
formBodyToGetStudentList);
} catch (IOException e) {

Log.e(
❏ Цитирования: 0,01% id: 381
"e:",
Log.getStackTraceString(e));
}

try {

if (jsonStudentList.getInt(TAG_SUCCESS) == 1) {

final String studentList = jsonStudentList.getString(
❏ Цитирования: 0,01% id: 382
"Classes"
);

sPref = MainActivity.this.getSharedPreferences(
❏ Цитирования: 0,01% id: 383
"com.studentattendance",
Context.MODE_PRIVATE);

Editor ed = sPref.edit(); ed.putString(
❏ Цитирования: 0,01% id: 384
"StudentList",
studentList); ed.commit();

} else {

final String message = jsonStudentList.getString(
❏ Цитирования: 0,01% id: 385
"message"
); MainActivity.this.runOnUiThread(new Runnable() {

public void run() {

toast = Toast.makeText(MainActivity.this, message,
Toast.LENGTH_LONG);

}

});

toast.setGravity(Gravity.CENTER, 0, 0);

toast.show();

Log.e(
❏ Цитирования: 0,01% id: 386
"e:",
jsonStudentList.getString(
❏ Цитирования: 0,01% id: 387
"message"
));
}

```

```

}

if (class_subjects.getInt(TAG_SUCCESS) == 1) {

final String classSubjects = class_subjects.getString(
❏ Цитирования: 0,01% id: 388
"ClassSubjects"
);

sPref = MainActivity.this.getSharedPreferences(
❏ Цитирования: 0,01% id: 389
"com.studentattendance",
Context.MODE_PRIVATE);

Editor ed = sPref.edit(); ed.putString(
❏ Цитирования: 0,01% id: 390
"ClassSubjects",
classSubjects); ed.commit();

} else {

final String message = class_subjects.getString(
❏ Цитирования: 0,01% id: 391
"message"
); MainActivity.this.runOnUiThread(new Runnable() {

public void run() {

toast = Toast.makeText(MainActivity.this, message,
Toast.LENGTH_LONG);

}

});

toast.setGravity(Gravity.CENTER, 0, 0);

toast.show();

Log.e(
❏ Цитирования: 0,01% id: 392
": ",
jsonStudentList.getString(
❏ Цитирования: 0,01% id: 393
"message"
));

}

}

} catch (JSONException e) {

Log.e(
❏ Цитирования: 0,01% id: 394
"Error !:",
Log.getStackTraceString(e));

}

}

return null;

}

protected void onPostExecute(String file_url) {

pDialog.dismiss();

}

}

// starting the service to register with GCM

private void registerGCM() {

Intent intent = new Intent(this, GcmIntentService.class); intent.putExtra(
❏ Цитирования: 0,01% id: 395
"key",
❏ Цитирования: 0,01% id: 396
"register"
);

startService(intent);

}

```

```

private boolean checkPlayServices() { GoogleApiAvailability apiAvailability =
GoogleApiAvailability.getInstance\(\);

int resultCode = apiAvailability.isGooglePlayServicesAvailable(this);

if (resultCode != ConnectionResult.SUCCESS) {

if (apiAvailability.isUserResolvableError(resultCode)) { apiAvailability.getErrorDialog(this,
resultCode,

PLAY_SERVICES_RESOLUTION_REQUEST)

.show();

} else {

Log.i(TAG,



Цитирования: 0,05% id: 397



"This device is not supported. Google Play Services not installed!"



);



Toast.makeText(getApplicationContext(),



Цитирования: 0,05% id: 398



"This device is not supported. Google Play Services not installed!",



Toast.LENGTH\_LONG).show();



finish();



}



return false;



}



return true;



}



@Override



protected void onResume() {



super.onResume\(\);



// register GCM registration complete receiver



LocalBroadcastManager.getInstance(this).registerReceiver(mRegistrationBroadcastReceiver,



new IntentFilter(Config.REGISTRATION\_COMPLETE));



// register new push message receiver



// by doing this, the activity will be notified each time a new message arrives



LocalBroadcastManager.getInstance(this).registerReceiver(mRegistrationBroadcastReceiver,



new IntentFilter(Config.PUSH\_NOTIFICATION));



}



@Override



protected void onPause() {



LocalBroadcastManager.getInstance(this).unregisterReceiver(mRegistrationBroadcastReceiver);



super.onPause\(\);



}



}



MessagesListAdapter.java



package com.studentattendance;



import java.util.ArrayList; import java.util.Collections; import java.util.Comparator; import



java.util.HashMap;



import android.content.Context; import android.graphics.Typeface; import



android.view.LayoutInflater; import android.view.View;



import android.view.ViewGroup; import android.widget.BaseAdapter; import



android.widget.ImageView; import android.widget.ListAdapter; import android.widget.TextView;



public class MessagesListAdapter extends BaseAdapter implements ListAdapter {



private ArrayList HashMap String, String list = new



ArrayList HashMap String, String ();



private Context context;


```



```

public MessagesListAdapter(ArrayList<HashMap<String, String> list, Context context) {
    this.list = list;
    this.context = context;
}

@Override
public int getCount() {
    return list.size();
}

@Override
public Object getItem(int pos) {
    return list.get(pos);
}

@Override
public long getItemId(int pos) {
    // return list.get(pos).getId();
    return 0;
    // just return 0 if your list items do not have an Id variable.
}

class HashMapComparator implements Comparator<HashMap<String, String> > {
    private final String key;

    public HashMapComparator(String key) {
        this.key = key;
    }

    public int compare(HashMap<String, String> first, HashMap<String, String> second) {
        // TODO: Null checking, both for maps and values
        return second.get(key).compareTo(first.get(key));
    }
}

public void loadData() {
}

public void saveData() {
}

@Override
public View getView(final int position, View convertView, ViewGroup parent) {
    View view =
        convertView;

    if (view == null) {
        LayoutInflater inflater = (LayoutInflater)
            context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);

        view = inflater.inflate(R.layout.messages_list_item, null);
    }

    Collections.sort(list, new HashMapComparator(
        "priority"
    ));

    Typeface mssgNameFont = Typeface.createFromAsset(context.getAssets(),
        "fonts/comic_sans_ms.ttf"
    );

    TextView mssgName = (TextView) view.findViewById(R.id.messageSubject);
    mssgName.setTypeface(mssgNameFont); mssgName.setText(list.get(position).get(
        "subject"
    ));
}

```

id: 399

id: 400

id: 401

```

"messageSubject"
));

ImageView mssgImg = (ImageView) view.findViewById(R.id.message_img); TextView mssgBody
= (TextView) view.findViewById(R.id.messageBody); mssgBody.setText(list.get(position).get(
Цитирования: 0,01% id: 402
"messageBody"
).length() 20
? list.get(position).get(
Цитирования: 0,01% id: 403
"messageBody"
).substring(0, 20).concat(
Цитирования: 0% id: 404
"...")
)
: list.get(position).get(
Цитирования: 0,01% id: 405
"messageBody"
));

TextView dateAndTime = (TextView) view.findViewById(R.id.dateAndTime);
dateAndTime.setText(list.get(position).get(
Цитирования: 0,01% id: 406
"date"
).concat(
Цитирования: 0,01% id: 407
"\ "
).concat(list.get(position).get(
Цитирования: 0,01% id: 408
"time"
)));

ImageView priorityImg = (ImageView) view.findViewById(R.id.priority_img);
if (list.get(position).get(
Цитирования: 0,01% id: 409
"status"
) ==
Цитирования: 0,01% id: 410
"new"
) mssgImg.setImageResource(R.drawable.ic_action_email);
else
mssgImg.setImageResource(R.drawable.ic_action_read);
if (list.get(position).get(
Цитирования: 0,01% id: 411
"priority"
) ==
Цитирования: 0,01% id: 412
"2"
) priorityImg.setImageResource(R.drawable.ic_action_important);
else if (list.get(position).get(
Цитирования: 0,01% id: 413
"priority"
) ==
Цитирования: 0,01% id: 414
"1"
) priorityImg.setImageResource(R.drawable.ic_action_half_important);
else
priorityImg.setImageResource(R.drawable.ic_action_not_important);
return view;
}
}

MyGcmPushReceiver.java
package com.studentattendance;
import android.content.Context; import android.content.Intent; import android.os.Bundle;
import android.support.v4.content.LocalBroadcastManager;
import android.text.TextUtils;

```

```

import android.util.Log;

import com.google.android.gms.gcm.GcmListenerService;

/**
 * Created by underhand on 25.05.2016.
 */

public class MyGcmPushReceiver extends GcmListenerService {

    private static final String TAG = MyGcmPushReceiver.class.getSimpleName();

    private NotificationUtils notificationUtils;

    /**
     Called when message is received.
     *
     @param fromSenderID of the sender.
     @param bundle Data bundle containing message data as key/value pairs.
     For Set of keys use data.keySet().
     */

    @Override
    public void onMessageReceived(String from, Bundle bundle) { String title = bundle.getString(
        "title" id: 415
    );

    String message = bundle.getString(
        "message" id: 416
    ); String image = bundle.getString(
        "image" id: 417
    );

    String timestamp = bundle.getString(
        "created_at" id: 418
    ); Log.e(TAG,
        "from: " id: 419
    + from);

    Log.e(TAG,
        "Title: " id: 420
    + title); Log.e(TAG,
        "message: " id: 421
    + message); Log.e(TAG,
        "image: " id: 422
    + image); Log.e(TAG,
        "timestamp: " id: 423
    + timestamp);

    if (!NotificationUtils.isAppInBackground(getApplicationContext()))
    {

        // app is in foreground, broadcast the push message

        Intent pushNotification = new Intent(Config.PUSH_NOTIFICATION); pushNotification.putExtra(
            "message", id: 424
        message);

        LocalBroadcastManager.getInstance(this).sendBroadcast(pushNotification);

        // play notification sound

        NotificationUtils notificationUtils = new NotificationUtils();
        notificationUtils.playNotificationSound();

    } else {

```

```

Intent resultIntent = new Intent(getApplicationContext(), MainActivity.class);

resultIntent.putExtra(
    "message",
    message);

if (TextUtils.isEmpty(image)) { showNotificationMessage(getApplicationContext(), title,
    message, timestamp, resultIntent);
} else {

showNotificationMessageWithBigImage(getApplicationContext(), title, message, timestamp,
resultIntent, image);

}

}

}

/**
 * Showing notification with text only
 */

private void showNotificationMessage(Context context, String title, String message, String
timeStamp, Intent intent) {

notificationUtils = new NotificationUtils(context);
intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK |

Intent.FLAG_ACTIVITY_CLEAR_TASK);

notificationUtils.showNotificationMessage(title, message, timeStamp,

intent);

}

/**
 * Showing notification with text and image
 */

private void showNotificationMessageWithBigImage(Context context, String title, String message,
String timeStamp, Intent intent, String imageUrl) {

notificationUtils = new NotificationUtils(context);
intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK |

Intent.FLAG_ACTIVITY_CLEAR_TASK);

notificationUtils.showNotificationMessage(title, message, timeStamp, intent, imageUrl);

}

}

NotificationUtils.java

package com.studentattendance;

import android.app.ActivityManager;

import android.app.Notification;

import android.app.NotificationManager; import android.app.PendingIntent; import
android.content.ComponentName; import android.content.Context;

import android.content.Intent;

import android.graphics.Bitmap;

import android.graphics.BitmapFactory;

import android.media.Ringtone;

import android.media.RingtoneManager;

import android.net.Uri;

import android.os.Build;

import android.support.v7.app.NotificationCompat;

import android.text.Html; import android.text.TextUtils; import android.util.Patterns;

import java.io.IOException;

import java.io.InputStream;

```

Цитирования: 0,01%

id: 425

```
import java.net.HttpURLConnection;

import java.net.URL;

import java.text.ParseException; import java.text.SimpleDateFormat; import java.util.Arrays;

import java.util.Date;

import java.util.List;

/**
 * Created by underhand on 21.03.2019.
 */

public class NotificationUtils {

    private static String TAG = NotificationUtils.class.getSimpleName();

    private Context mContext; public NotificationUtils() {

    }

    public NotificationUtils(Context mContext) {

        this.mContext = mContext;

    }

    public void showNotificationMessage(String title, String message, String timeStamp, Intent intent)
    {

        showNotificationMessage(title, message, timeStamp, intent, null);

    }

    public void showNotificationMessage(final String title, final String message, final String
    timeStamp, Intent intent, String imageUrl) {

        // Check for empty push message

        if (TextUtils.isEmpty(message))

            return;

        // notification icon

        final int icon = R.drawable.ic_launcher;

        intent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP | Intent.FLAG_ACTIVITY_SINGLE_TOP);

        final PendingIntent resultPendingIntent = PendingIntent.getActivity(

            mContext, 0,

            intent, PendingIntent.FLAG_CANCEL_CURRENT

        );

        final NotificationCompat.Builder mBuilder = new

        NotificationCompat.Builder(

            mContext);

        final Uri alarmSound = RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);

        if (!TextUtils.isEmpty(imageUrl)) {

            if (imageUrl != null && imageUrl.length() > 4 && Patterns.WEB_URL.matcher(imageUrl).matches())

            {

                Bitmap bitmap = getBitmapFromURL(imageUrl);

                if (bitmap != null) {

                    showBigNotification(bitmap, mBuilder, icon, title, message, timeStamp, resultPendingIntent,

                    alarmSound);

                } else {

                    showSmallNotification(mBuilder, icon, title, message, timeStamp, resultPendingIntent,

                    alarmSound);

                }

            }

        } else {

            showSmallNotification(mBuilder, icon, title, message, timeStamp, resultPendingIntent,

            alarmSound);

            playNotificationSound();

        }

    }

}
```

```
}  
}  
  
private void showSmallNotification(NotificationCompat.Builder mBuilder, int icon, String title,  
String message, String timeStamp, PendingIntent resultPendingIntent, Uri alarmSound) {  
  
    NotificationCompat.InboxStyle inboxStyle = new  
    NotificationCompat.InboxStyle();  
  
    if (Config.appendNotificationMessages) {  
        // store the notification in shared pref first  
  
        MyApplication.getInstance().getPrefManager().addNotification(message);  
  
        // get the notifications from shared preferences  
  
        String oldNotification = MyApplication.getInstance().getPrefManager().getNotifications();  
  
        List<String> messages = Arrays.asList(oldNotification.split(  
        "Cитирования: 0,01% id: 426  
"\\|"  
));  
  
        for (int i = messages.size() - 1; i = 0; i--) { inboxStyle.addLine(messages.get(i));  
        }  
    } else {  
  
        inboxStyle.addLine(message);  
    }  
  
    Notification notification;  
  
    notification = mBuilder.setSmallIcon(icon).setTicker(title).setWhen(0)  
  
    .setAutoCancel(true)  
  
    .setContentTitle(title)  
  
    .setContentIntent(resultPendingIntent)  
  
    .setSound(alarmSound)  
  
    .setStyle(inboxStyle)  
  
    .setWhen(getTimeMilliSec(timeStamp))  
  
    .setSmallIcon(R.drawable.ic_launcher)  
  
    .setLargeIcon(BitmapFactory.decodeResource(mContext.getResources(), icon))  
  
    .setContentText(message)  
  
    .build();  
  
    NotificationManager notificationManager = (NotificationManager)  
  
    mContext.getSystemService(Context.NOTIFICATION_SERVICE);  
    notificationManager.notify(Config.NOTIFICATION_ID, notification);  
}  
  
private void showBigNotification(Bitmap bitmap, NotificationCompat.Builder  
mBuilder, int icon, String title, String message, String timeStamp, PendingIntent  
resultPendingIntent, Uri alarmSound) {  
  
    NotificationCompat.BigPictureStyle bigPictureStyle = new  
    NotificationCompat.BigPictureStyle(); bigPictureStyle.setBigContentTitle(title);  
    bigPictureStyle.setSummaryText(Html.fromHtml(message).toString());  
    bigPictureStyle.bigPicture(bitmap);  
  
    Notification notification;  
  
    notification = mBuilder.setSmallIcon(icon).setTicker(title).setWhen(0)  
  
    .setAutoCancel(true)  
  
    .setContentTitle(title)  
  
    .setContentIntent(resultPendingIntent)  
  
    .setSound(alarmSound)  
  
    .setStyle(bigPictureStyle)  
  
    .setWhen(getTimeMilliSec(timeStamp))
```

```
.setSmallIcon(R.drawable.ic_launcher)

.setLargeIcon(BitmapFactory.decodeResource(mContext.getResources(), icon))

.setContentText(message)

.build();

NotificationManager notificationManager = (NotificationManager)
mContext.getSystemService(Context.NOTIFICATION_SERVICE);
notificationManager.notify(Config.NOTIFICATION_ID_BIG_IMAGE,
notification);
}

/**
Downloading push notification image before displaying it in
the notification tray
*/

public Bitmap getBitmapFromURL(String strURL) {

try {

URL url = new URL(strURL);

URLConnection connection = (URLConnection) url.openConnection();

connection.setDoInput(true); connection.connect();

InputStream input = connection.getInputStream(); Bitmap myBitmap =
BitmapFactory.decodeStream(input); return myBitmap;

} catch (IOException e) { e.printStackTrace(); return null;

}

}

// Playing notification sound

public void playNotificationSound() {

try {

Uri notification = RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);

Ringtone r = RingtoneManager.getRingtone(MyApplication.getInstance().getApplicationContext(),
notification);

r.play();

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

}

}

/**
* Method checks if the app is in background or not
*/

public static boolean isAppIsInBackground(Context context) {

boolean isInBackground = true; ActivityManager am = (ActivityManager)
context.getSystemService(Context.ACTIVITY_SERVICE);

if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.KITKAT_WATCH) {

List<ActivityManager.RunningAppProcessInfo> runningProcesses = am.getRunningAppProcesses();

for (ActivityManager.RunningAppProcessInfo processInfo : runningProcesses) {

if (processInfo.importance ==
ActivityManager.RunningAppProcessInfo.IMPORTANCE_FOREGROUND) {

for (String activeProcess : processInfo.pkgList) {

if (activeProcess.equals(context.getPackageName())) { isInBackground = false;

}

}

}

}

}

}
```

```

    } else {

List<ActivityManager.RunningTaskInfo> taskInfo = am.getRunningTasks(1);

ComponentName componentInfo = taskInfo.get(0).topActivity;

if (componentInfo.getPackageName().equals(context.getPackageName()))

{

isInBackground = false;

}

}

return isInBackground;

}

// Clears notification tray messages

public static void clearNotifications() {

NotificationManager notificationManager = (NotificationManager)
MyApplication.getInstance().getSystemService(Context.NOTIFICATION_SERVICE);

notificationManager.cancelAll();

}

public static long getTimeMilliSec(String timeStamp) {

SimpleDateFormat format = new SimpleDateFormat(
); try {

Date date = format.parse(timeStamp);

return date.getTime();

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

}

return 0;

}

}

package com.studentattendance;

import android.content.Context;

import android.view.GestureDetector;

import android.view.GestureDetector.SimpleOnGestureListener;

import android.view.MotionEvent;

import android.view.View;

import android.view.View.OnTouchListener;

public class OnSwipeTouchListener implements OnTouchListener {

private final GestureDetector gestureDetector;

public OnSwipeTouchListener(Context ctx) {

gestureDetector = new GestureDetector(ctx, new GestureListener());

}

@Override

public boolean onTouch(View v, MotionEvent event) {

return gestureDetector.onTouchEvent(event);

}

private final class GestureListener extends SimpleOnGestureListener {

private static final int SWIPE_THRESHOLD = 100;

private static final int SWIPE_VELOCITY_THRESHOLD = 100;

@Override

public boolean onDown(MotionEvent e) {

return true;

```

Цитирования: 0,01%

id: 427

"xxx-MM-dd HH:mm:ss"


```
}

@Override

public boolean onFling(MotionEvent e1, MotionEvent e2, float velocityX,
float velocityY) {

boolean result = false; try {

float diffY = e2.getY() - e1.getY();

float diffX = e2.getX() - e1.getX();

if (Math.abs(diffX) > Math.abs(diffY)) {

if (Math.abs(diffX) > SWIPE_THRESHOLD && Math.abs(velocityX)
> SWIPE_VELOCITY_THRESHOLD) {

if (diffX > 0) { onSwipeRight();

} else {

onSwipeLeft();

}

}

} else if (Math.abs(diffY) > SWIPE_THRESHOLD && Math.abs(velocityY)
> SWIPE_VELOCITY_THRESHOLD) {

if (diffY > 0) { onSwipeBottom();

} else {

onSwipeTop();

}

}

} catch (Exception exception) { exception.printStackTrace();

}

return result;

}

}

public void onSwipeRight() {

}

public void onSwipeLeft() {

}

public void onSwipeTop() {

}

public void onSwipeBottom() {

}

}

ScheduleListAdapter.java

package com.studentattendance;

import android.content.Context; import android.graphics.Typeface; import
android.view.LayoutInflater; import android.view.View;

import android.view.ViewGroup; import android.widget.BaseAdapter; import
android.widget.ListAdapter; import android.widget.TextView;

import java.util.ArrayList;

import java.util.HashMap;

public class ScheduleListAdapter extends BaseAdapter implements ListAdapter { private ArrayList
HashMap<String, String> list = new ArrayList<>(); private Context context;

public ScheduleListAdapter(ArrayList<HashMap<String, String>> list, Context context) {
```

```
this.list = list;

this.context = context;

}

@Override

public int getCount() {

return list.size();

}

@Override

public Object getItem(int pos) {

return list.get(pos);

}

@Override

public long getItemId(int pos) {

// return list.get(pos).getId();

return 0;

}

@Override

public View getView(final int position, View convertView, ViewGroup parent) { View view =
convertView;

if (view == null) {

LayoutInflater inflater = (LayoutInflater)

context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);

view = inflater.inflate(R.layout.schedule_item, null);

}

Typeface subjectFont = Typeface.createFromAsset(context.getAssets(),



» Цитирования: 0,01% id: 428



"font/segoesch.3"



);

TextView subjectName = (TextView) view.findViewById(R.id.name);
subjectName.setTypeface(subjectFont); subjectName.setText(list.get(position).get(



» Цитирования: 0,01% id: 429



"Name"



));

TextView classroom = (TextView) view.findViewById(R.id.classroom);
classroom.setTypeface(subjectFont); classroom.setText(list.get(position).get(



» Цитирования: 0,01% id: 430



"Classroom"



));

TextView time = (TextView) view.findViewById(R.id.time); time.setTypeface(subjectFont);
time.setText(list.get(position).get(



» Цитирования: 0,01% id: 431



"Time"



));

return view;

}

}

SpinnerAdapterForStatActivity.java

package com.studentattendance;

import android.content.Context; import android.graphics.Typeface; import android.view.View;

import android.view.ViewGroup; import android.widget.AdapterView; import
android.widget.TextView;

public class SpinnerAdapterForStatActivity extends ArrayAdapter String { Typeface font;

SpinnerAdapterForStatActivity(Context context, int resource, String[] spinnerArray) {

super(context, resource, spinnerArray);


```

```
font = Typeface.createFromAsset(context.getAssets(),
    "fonts/serif_print_bold.ttf"
);
}

@Override

public View getView(int position, View convertView, ViewGroup parent) { TextView view =
(TextView) super.getView(position, convertView, parent); return redesignTextView(view);
}

@Override

public View getDropDownView(int position, View convertView, ViewGroup parent)

{

TextView view = (TextView) super.getDropDownView(position, convertView,
parent);

return redesignTextView(view);

}

private TextView redesignTextView(TextView view) { view.setTextSize(20);
view.setSingleLine(false); view.setTypeface(font);

return view;

}

}

StatisticActivity.java

package com.studentattendance;

import android.app.ActionBar; import android.app.Activity; import android.graphics.Color;
import android.graphics.drawable.ColorDrawable;

import android.os.Bundle; import android.text.Html; import android.view.View; import
android.view.Window;

import android.widget.LinearLayout;

import android.widget.LinearLayout.LayoutParams;

import android.widget.Toast;

import org.achartengine.ChartFactory;

import org.achartengine.GraphicalView;

import org.achartengine.model.CategorySeries;

import org.achartengine.model.SeriesSelection;

import org.achartengine.renderer.DefaultRenderer;

import org.achartengine.renderer.SimpleSeriesRenderer;

public class StatisticActivity extends Activity {

private static int[] COLORS = new int[] { Color.RED, Color.GREEN, Color.YELLOW };

private static double[] VALUES = new double[] { 75, 25, 15 };

private static String[] NAME_LIST = new String[] {

Цитирования: 0,01% id: 432
"Відсутні",
Цитирования: 0,01% id: 434
"Присутні",
Цитирования: 0,01% id: 435
"Запізнилися"
};

private CategorySeries mSeries = new CategorySeries(

Цитирования: 0% id: 436
""
); private DefaultRenderer mRenderer = new DefaultRenderer(); private GraphicalView
mChartView;


@Override

protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState);
```

```

getWindow().requestFeature(Window.FEATURE_ACTION_BAR);
setContentView(R.layout.activity_statistic_all_subjects); ActionBar actionBar = getActionBar();
actionBar.setBackgroundDrawable(new

ColorDrawable(Color.parseColor(



Цитирования: 0,01%



id: 437


```

"# 71ac"



Цитирования: **0,01%**

id: 438


" font_color='#0bf748' /font "


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**


id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**


id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**


id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**


id: 438


Цитирования: **0,01%**


id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**


id: 438


Цитирования: **0,01%**


id: 438


Цитирования: **0,01%**


id: 438


Цитирования: **0,01%**


id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**


id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**


id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**


id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**

id: 438


Цитирования: **0,01%**

id: 438

```

mChartView.setOnLongClickListener(new View.OnLongClickListener() {

@Override

public boolean onLongClick(View v) { SeriesSelection seriesSelection =

mChartView.getCurrentSeriesAndPoint();

if (seriesSelection == null) {

// Toast.makeText(StatisticActivity.this,

❗ Цитирования: 0,03% id: 439
"Na chart

// element was long pressed",

Toast.LENGTH_SHORT);

return false;

} else {

// Toast.makeText(StatisticActivity.this, "Chart element

// data point index "

// + seriesSelection.getPointIndex() + " was long

// pressed", Toast.LENGTH_SHORT);

return true;

}

}

});

layout.addView(mChartView, new LayoutParams(LayoutParams.MATCH_PARENT,

LayoutParams.MATCH_PARENT));

} else {

mChartView.repaint();

}

}

}

StatisticTypeActivity.java

package com.studentattendance;

import android.app.ActionBar; import android.content.Context; import android.content.Intent;

import android.content.SharedPreferences;

import android.graphics.Color;

import android.graphics.Typeface;

import android.graphics.drawable.ColorDrawable;

import android.os.Bundle; import android.text.Html; import android.view.Menu;

import android.view.MenuInflater;

import android.view.MenuItem;

import android.view.View;

import android.view.Window;

import android.widget.AdapterView;

import android.widget.AdapterView.OnItemClickListener;

import android.widget.Button; import android.widget.Spinner; import android.widget.TextView;

import com.google.gson.Gson;

import java.util.ArrayList; import java.util.Arrays; import java.util.HashMap; import

java.util.HashSet; import java.util.Set;

public class StatisticTypeActivity extends ILocalScheduleLoaderActivity

implements View.OnClickListener {

SharedPreferences sPref; Button Ok;

Spinner subjectSpinner; Spinner groupeSpinner;

ArrayList HashMap String, String scheduleList, groupsList, filteredGroupsList;

```

```

String statData = "
❏ Цитирования: 0,14% id: 440
";

@Override

protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState);
getWindow().requestFeature(Window.FEATURE_ACTION_BAR);
setContentView(R.layout.activity_statistic_type); ActionBar actionBar = getSupportActionBar();
actionBar.setBackgroundDrawable(new
ColorDrawable(Color.parseColor("#a71aca

# a71aca
❏ Цитирования: 0,02% id: 441
")); actionBar.setTitle(Html.fromHtml("
font color='#0bf748' /font
❏ Цитирования: 0,05% id: 442
")); actionBar.show();

Typeface myTypeface = Typeface.createFromAsset(getAssets(),
"
fonts/v_CCforkedTongue.ttf
❏ Цитирования: 0,13% id: 443
");

TextView selectFiltersTV = (TextView) findViewById(R.id.selectFiltersTV);
selectFiltersTV.setTypeface(myTypeface);

subjectSpinner = (Spinner) findViewById(R.id.subjectsSpinner);
setOnItemSelectedListener(subjectSpinner);

sPref = this.getSharedPreferences("
com.studentattendance
❏ Цитирования: 0,03% id: 444
", Context.MODE_PRIVATE);

if (!sPref.getString("
UserType
❏ Цитирования: 0,01% id: 445
", "").equals("
t
❏ Цитирования: 0,19% id: 446
")) { groupsSpinner = (Spinner) findViewById(R.id.groupsSpinner);
groupsSpinner.setEnabled(true); groupsSpinner.setVisibility(View.VISIBLE);
}

Ok = (Button) findViewById(R.id.ok);

Ok.setOnClickListener(this);

new LocalScheduleLoader(StatisticTypeActivity.this, this, "").execute(); new
LocalGroupListLoader(StatisticTypeActivity.this, this, "
All Subjects
❏ Цитирования: 0,01% id: 447
",
"
").execute();
}

public void onClick(View v) {

new GetStatisticData(this, this, getSelectedSubjectsAndGroupsJson()).execute();
}

public void setOnItemSelectedListener(Spinner spinner) { spinner.setOnItemSelectedListener(new
OnItemSelectedListener() {

@Override

public void onItemSelected(AdapterView ? parentView, View selectedItemView, int position, long
id) {

refreshGroupsSpinnerOnSubjectChange();
}

@Override

public void onNothingSelected(AdapterView ? parent) {

// TODO Auto-generated method stub
}
}
}

```

```
});
}

@Override
public boolean onCreateOptionsMenu(Menu menu) { MenuInflater inflater = getMenuInflater();
inflater.inflate(R.menu.statistic_type_activity_actions, menu);
return super.onCreateOptionsMenu(menu);
}

@Override
public boolean onOptionsItemSelected(int featureId, MenuItem item) {
int itemId = item.getItemId();

switch (itemId) {

case R.id.action_help:

Intent helpActivity = new Intent(getApplicationContext(), HelpActivity.class);

startActivity(helpActivity);

break;

}

return true;
}

void CallActivityForSelectedFilters(String statData) { startActivity(new
Intent(getApplicationContext(),
StatisticActivity.class)).putExtra("stat
Цитирования: 0,11% id: 448
", statData));
}

String getSelectedSubjectsAndGroups() {
return new Gson().toJson(getSelectedSubjectsAndGroupsList());
}

ArrayList<HashMap<String, String>> getSelectedSubjectsAndGroupsList() { ArrayList<HashMap<String,
String>> subjectAndGroupsPairs = new
ArrayList<>();

if (subjectSpinner.getSelectedItem().equals("
Всі предмети
Цитирования: 0,16% id: 449
"))

for (int i = 0; i < scheduleList.size(); i++) addAllGroupsForSubject(subjectAndGroupsPairs,
scheduleList.get(i));

size

for (int i = 0; i < scheduleList.size(); i++)

if

(scheduleList.get(i).get("
Name
Цитирования: 0,18% id: 450
").equals(subjectSpinner.getSelectedItem())) { addAllGroupsForSubject(subjectAndGroupsPairs,
scheduleList.get(i));
}

return subjectAndGroupsPairs;
}

void addAllGroupsForSubject(ArrayList<HashMap<String, String>> subjectAndGroupsPairs,
HashMap<String, String> subject) {
for (int i = 0; i < groupsList.size(); i++) {

if (groupsList.get(i).get("
SubjectID
Цитирования: 0,02% id: 451
```

<pre> ").equals(subject.get(" </pre>	
ID	
<pre> Цитирования: 0,05% id: 452 ")) { HashMap String, String hm = new HashMap String, String (); hm.put(" </pre>	
SubjectID	
<pre> Цитирования: 0,01% id: 453 ", subject.get(" </pre>	
ID	
<pre> Цитирования: 0,02% id: 454 ").toString()); hm.put(" </pre>	
ClassID	
<pre> Цитирования: 0,02% id: 455 ", groupsList.get().get(" ClassID").toString()); subjectAndGroupePairs.add(hm); } } } @Override void getScheduleData(ArrayList HashMap String, String scheduleList) { if (!scheduleList.isEmpty()) { this.scheduleList = scheduleList; setSubjectSpinnerAdapter(scheduleList); } } void setGroupeSpinnerData(ArrayList HashMap String, String groupsList) { this.groupsList = groupsList; setGroupeSpinnerAdapter(groupsList); } void refreshGroupsSpinnerOnSubjectChange() { if (!subjectSpinner.getSelectedItem().equals("Всі предмети </pre>	
<pre> Цитирования: 0,07% id: 456 ")) { getGroupsForSpecificSubject(); groupsList = filteredGroupsList; setGroupeSpinnerAdapter(filteredGroupsList); } else { new LocalGroupListLoader(StatisticTypeActivity.this, this, " </pre>	
All_Subjects	
<pre> Цитирования: 0,13% id: 457 ", "").execute(); setGroupeSpinnerAdapter(groupsList); } } void getGroupsForSpecificSubject() { filteredGroupsList = new ArrayList (); for (int i = 0; i < scheduleList.size(); i++) if (scheduleList.get().get(" </pre>	
Name	
<pre> Цитирования: 0,06% id: 458 ").equals(subjectSpinner.getSelectedItem())) { new LocalGroupListLoader(StatisticTypeActivity.this, this, scheduleList.get().get(" </pre>	
ID	
<pre> Цитирования: 0,11% id: 459 ", "").execute(); filteredGroupsList.addAll(groupsList); } } void setSubjectSpinnerAdapter(ArrayList HashMap String, String subjectList) </pre>	


```

{
    SpinnerAdapterForStatActivity adapter = new
    SpinnerAdapterForStatActivity(StatisticTypeActivity.this, android.R.layout.simple_spinner_item,
    getArrayWithFilteredDataForSpinnerAdapter(subjectList, "
Bci
предмети"));

    adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
    subjectSpinner.setAdapter(adapter);
}

void setGroupeSpinnerAdapter(ArrayList HashMap String, String groupsList) {
    SpinnerAdapterForStatActivity adapter = new
    SpinnerAdapterForStatActivity(StatisticTypeActivity.this, android.R.layout.simple_spinner_item,
    getArrayWithFilteredDataForSpinnerAdapter(groupsList, "Bci групи
Цитирования: 0,22% id: 460
"));

    adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
    groupeSpinner.setAdapter(adapter);
}

String[] getArrayWithFilteredDataForSpinnerAdapter(ArrayList HashMap String, String list, String
title) {
    Set String hs = new HashSet(); hs.add(title);
    for (int i = 0; i < list.size(); ++i) { hs.add(list.get(i).get("Name"));
    }
    String[]
    spinnerArray = new String[hs.size()]; hs.toArray(spinnerArray); Arrays.sort(spinnerArray);
    return spinnerArray;
}
}

StudentMainScheduleActivity.java
package com.studentattendance;

import android.app.ActionBar; import android.app.Activity; import android.app.ProgressDialog;
import android.content.Context;

import android.content.SharedPreferences;

import android.graphics.Color;

import android.graphics.Typeface;

import android.graphics.drawable.ColorDrawable;

import android.os.AsyncTask; import android.os.Bundle; import android.text.Html; import
android.util.Log;

import android.widget.ImageView; import android.widget.ListView; import
android.widget.RelativeLayout; import android.widget.TextView;

import org.json.JSONArray; import org.json.JSONException; import org.json.JSONObject;

import java.text.SimpleDateFormat;

import java.util.ArrayList; import java.util.Calendar; import java.util.HashMap; import
java.util.Locale;

public class StudentMainScheduleActivity extends Activity {

    SharedPreferences sPref; private ProgressDialog pDialog; JSONObject jsonSchedule;
    ArrayList HashMap String, String scheduleList; JSONArray days = null;

    ListView lv; String dayOfWeek;

    static Calendar dateChosen; TextView dayAndDateStatus; ImageView youFreeImg;

    @Override

    protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState);
    setContentView(R.layout.student_main_schedule_activity); ActionBar actionBar = getActionBar();

    actionBar.setBackgroundDrawable(new ColorDrawable(Color.parseColor("#a71aca
Цитирования: 0,02% id: 461

```

```

    ""));
    actionBar.setTitle(Html.fromHtml("
font color='#0bf748' //font
Цитирования: 0,12% id: 462
    "")); actionBar.show();

    youFreeimg = (ImageView) findViewById(R.id.youFreeimage); dayAndDateStatus = (TextView)
    findViewById(R.id.day_of_week); Typeface myTypeface =
    Typeface.createFromAssets(getAssets(),
    "
fonts/v_CCforkedTongue.ttf"); dayAndDateStatus.setTypeface(myTypeface); dayOfWeek =
    Calendar.getInstance(Locale.US).getDisplayName(Calendar.DAY_OF_WEEK, Calendar.LONG,
    Locale.US);

    dateChosen = Calendar.getInstance();

    dayAndDateStatus
    .setText(Calendar.getInstance(Locale.US).getDisplayName(Calendar.DAY_OF_WEEK,
    Calendar.LONG, Locale.getDefault())
    + ", " + new
    SimpleDateFormat(
    Цитирования: 0,02% id: 463
    "dd.MM.yyyy"
    ).format(dateChosen.getTime()));

    sPref = this.getSharedPreferences(
    Цитирования: 0,01% id: 464
    "com.studentattendance",
    Context.MODE_PRIVATE);

    try {

    days = new JSONArray(sPref.getString(
    Цитирования: 0,01% id: 465
    "Schedule",
    Цитирования: 0% id: 466
    ""
    ));

    } catch (JSONException e) {

    Log.e(
    Цитирования: 0,01% id: 467
    "E:",
    Log.getStackTraceString(e));

    }

    scheduleList = new ArrayList<HashMap<String, String>>();

    lv = (ListView) findViewById(R.id.schedule);

    new LoadSchedule().execute();

    RelativeLayout rLayout = (RelativeLayout) findViewById(R.id.relLayout);
    rLayout.setOnTouchListener(new
    OnSwipeTouchListener(StudentMainScheduleActivity.this) {

    public void onSwipeTop() { DateDayChanger(7);

    }

    public void onSwipeRight() { DateDayChanger(-1);

    }

    public void onSwipeLeft() { DateDayChanger(1);

    }

    public void onSwipeBottom() { DateDayChanger(-7);

    }

    });

    lv.setOnTouchListener(new
    OnSwipeTouchListener(StudentMainScheduleActivity.this) {

    public void onSwipeRight() { DateDayChanger(-1);

    }

```

```

public void onSwipeLeft() { DateDayChanger(1);
}

});
}

public void DateDayChanger(int daysToChange) { youFreeImg.setBackgroundResource(0);
dateChosen.add(Calendar.DATE, daysToChange);

dayOfWeek = dateChosen.getDisplayName(Calendar.DAY_OF_WEEK, Calendar.LONG, Locale.US);

dayAndDateStatus.setText(dateChosen.getDisplayName(Calendar.DAY_OF_WEEK,
Calendar.LONG, Locale.getDefault()) +
Цитирования: 0% id: 468
", "

+ new SimpleDateFormat(
Цитирования: 0,02% id: 469
"dd.MM.yyyy"
).format(dateChosen.getTime());

lv.setAdapter(null); scheduleList.clear();

new LoadSchedule().execute();
}

class LoadSchedule extends AsyncTask<String, String, String> {

@Override

protected void onPreExecute() {

super.onPreExecute();

pDialog = new ProgressDialog(StudentMainScheduleActivity.this); pDialog.setMessage(
Цитирования: 0,03% id: 470
"Завантаження даних, зачекайте будь ласка..."
); pDialog.setIndeterminate(false);

pDialog.setCancelable(false); pDialog.show();
}

protected String doInBackground(String... args) {

try {

for (int i = 0; i < days.length(); i++) { JSONObject day = days.getJSONObject(i); Log.e(
Цитирования: 0,01% id: 471
"e:",
day.toString());

JSONArray d = day.getJSONArray(dayOfWeek);

for (int j = 0; j < d.length(); j++) { JSONObject subject = d.getJSONObject(j);

String id = subject.getString(
Цитирования: 0,01% id: 472
"id"
); String name = subject.getString(
Цитирования: 0,01% id: 473
"Name"
);

String dayOfWeek = subject.getString(
Цитирования: 0,01% id: 474
"DayOfWeek"
); String classroom =
Цитирования: 0,01% id: 475
"Classroom:"
+
subject.getString(
Цитирования: 0,01% id: 476
"Classroom"
);

String time = subject.getString(
Цитирования: 0,01% id: 477
"Time"
);
}
}
}

```

```

HashMap<String, String> map = new HashMap<String, String> (); map.put(
Цитирования: 0,01% id: 478
"id",
id);

map.put(
Цитирования: 0,01% id: 479
"name",
name); map.put(
Цитирования: 0,01% id: 480
"classroom",
classroom); map.put(
Цитирования: 0,01% id: 481
"dayOfWeek",
dayOfWeek); map.put(
Цитирования: 0,01% id: 482
"time",
time);

scheduleList.add(map);
}
}

} catch (JSONException e) {

Log.e(
Цитирования: 0,01% id: 483
"Er.",
Log.getStackTraceString(e));

}

return null;
}

protected void onPostExecute(String file_url) {
pDialog.dismiss(); runOnUiThread(new Runnable() {

public void run() {

if (!scheduleList.isEmpty()) { ScheduleListAdapter scheduleAdapter = new
ScheduleListAdapter(scheduleList, StudentMainScheduleActivity.this);

lv.setAdapter(scheduleAdapter);

} else {

dayAndDateStatus.append(
Цитирования: 0,01% id: 484
"\ \ "
+
Цитирования: 0,02% id: 485
"Занять немає!\ "
);

youFreelmg.setBackgroundResource(R.drawable.ic_green_happy_man);

}

}

});

}

}

}

}

TeacherMainScheduleActivity.java

package com.studentattendance;

import android.annotation.SuppressLint;

import android.app.ActionBar;

import android.app.DatePickerDialog;

import android.app.Dialog;

import android.app.DialogFragment; import android.app.ProgressDialog; import
android.content.Context; import android.content.Intent;

```

```

import android.content.SharedPreferences;

import android.graphics.Color;

import android.graphics.Typeface;

import android.graphics.drawable.ColorDrawable;

import android.os.Bundle; import android.text.Html; import android.util.Log; import
android.view.Menu;

import android.view.MenuInflater; import android.view.MenuItem; import android.view.View;

import android.view.Window;

import android.widget.AdapterView;

import android.widget.AdapterView.OnItemClickListener;

import android.widget.DatePicker; import android.widget.ImageView; import
android.widget.ListView; import android.widget.RelativeLayout; import android.widget.TextView;

import org.json.JSONArray; import org.json.JSONException; import org.json.JSONObject;

import java.text.SimpleDateFormat;

import java.util.ArrayList; import java.util.Calendar; import java.util.Date; import
java.util.HashMap; import java.util.Locale;

public class TeacherMainScheduleActivity extends LocalScheduleLoaderActivity {

    SharedPreferences sPref; private ProgressDialog pDialog; JSONObject jsonSchedule;

    ArrayList<HashMap<String, String>> scheduleList; JSONArray days = null;

    ListView lv; String dayOfWeek;

    static Calendar dateChosen; TextView dayAndDateStatus; ImageView youFreeImg;

    @Override

    protected void onCreate(Bundle savedInstanceState) { super.onCreate(savedInstanceState);
    getWindow().requestFeature(Window.FEATURE_ACTION_BAR);
    setContentView(R.layout.teacher_main_schedule_activity); ActionBar actionBar = getSupportActionBar();
    actionBar.setBackgroundDrawable(new

    ColorDrawable(Color.parseColor(
    Цитирования: 0,01% id: 486
    "# 71acc"
    )); actionBar.setTitle(Html.fromHtml(
    Цитирования: 0,02% id: 487
    "font color='#0bf748' /font "
    )); actionBar.show();

    youFreeImg = (ImageView) findViewById(R.id.youFreeImage); dayAndDateStatus = (TextView)
    findViewById(R.id.day_of_week); Typeface myTypeface = Typeface.createFromAsset(getAssets(),

    Цитирования: 0,01% id: 488
    "font/_CCTorkedTongue.9f"
    ); dayAndDateStatus.setTypeface(myTypeface); dayOfWeek =

    Calendar.getInstance(Locale.US).getDisplayName(Calendar.DAY_OF_WEEK,
    Calendar.LONG, Locale.US);

    dateChosen = Calendar.getInstance();

    dayAndDateStatus

    .setText(Calendar.getInstance(Locale.US).getDisplayName(Calendar.DAY_OF_WEEK,
    Calendar.LONG, Locale.getDefault()));

    +
    Цитирования: 0% id: 489
    " "
    + new

    SimpleDateFormat(
    Цитирования: 0,02% id: 490
    "dd.MM.yyyy"
    ).format(dateChosen.getTime());

    sPref = this.getSharedPreferences(
    Цитирования: 0,01% id: 491
    "com.studentattendance",
    Context.MODE_PRIVATE);

    try {

    days = new JSONArray(sPref.getString(
    Цитирования: 0,01% id: 492

```

```

"Schedule",
Цитирования: 0% id: 493
""
));

} catch (JSONException e) {

Log.e(
Цитирования: 0,01% id: 494
"e:",
Log.getStackTraceString(e));

}

scheduleList = new ArrayList<HashMap<String, String>>();

lv = (ListView) findViewById(R.id.schedule);

new LocalScheduleLoader(TeacherMainScheduleActivity.this, this, dayOfWeek).execute();

lv.setOnItemClickListener(new OnItemClickListener() {

public void onItemClick(AdapterView ? parent, View view, int position,

long id) {

String pid = scheduleList.get(position).get(
Цитирования: 0,01% id: 495
"id"
); String dayAndTime =

scheduleList.get(position).get("DayOfWeek").concat("
Цитирования: 0,02% id: 496
")

.concat(new SimpleDateFormat("

yyyy.MM.dd
Цитирования: 0,05% id: 497
").format(dateChosen.getTime()));

.concat(scheduleList.get(position).get("

Time
Цитирования: 0,05% id: 498
")); Intent in = new Intent(getApplicationContext(),

AttendanceTaker.class);

in.putExtra("

id
Цитирования: 0,01% id: 499
", pid.concat("

").concat(dayAndTime));

startActivityForResult(in, 100);

}

});

RelativeLayout rLayout = (RelativeLayout) findViewById(R.id.relLayout);

rLayout.setOnClickListener(new

OnSwipeTouchListener(TeacherMainScheduleActivity.this) {

public void onSwipeTop() { DateDayChanger(7);

}

public void onSwipeRight() { DateDayChanger(-1);

}

public void onSwipeLeft() { DateDayChanger(1);

}

public void onSwipeBottom() { DateDayChanger(-7);

}

});

lv.setOnClickListener(new

OnSwipeTouchListener(TeacherMainScheduleActivity.this) {

public void onSwipeRight() { DateDayChanger(-1);

}

}

```

```

public void onSwipeLeft() { DateDayChanger(1);
}
});
}

@Override

public boolean onCreateOptionsMenu(Menu menu) { MenuInflater inflater = getMenuInflater();
inflater.inflate(R.menu.teacher_main_actions, menu); return super.onCreateOptionsMenu(menu);
}

@Override

public boolean onOptionsItemSelected(int featureId, MenuItem item) {

int itemId = item.getItemId();

switch (itemId) {

case R.id.previous: DateDayChanger(-1); break;

case R.id.next: DateDayChanger(1); break;

case R.id.calendar:

Calendar c1 = Calendar.getInstance(Locale.US); c1.setTime(new Date());

if (dateChosen.get(Calendar.DAY_OF_YEAR) != c1.get(Calendar.DAY_OF_YEAR))

|| dateChosen.get(Calendar.YEAR) != c1.get(Calendar.YEAR)) {

dateChosen = c1; DateDayChanger(0);

} else {

showDatePickerDialog(this.findViewById(android.R.id.content));

}

break;

case R.id.action_messages:

Intent messageActivity = new Intent(getApplicationContext(), MessagesActivity.class);

startActivity(messageActivity);

break;

case R.id.action_help:

Intent helpActivity = new Intent(getApplicationContext(), HelpActivity.class);

startActivity(helpActivity);

break;

case R.id.action_statistic:

Intent statisticActivity = new Intent(getApplicationContext(), StatisticTypeActivity.class);

startActivity(statisticActivity);

break;

}

return true;

}

public void showDatePickerDialog(View v) { DialogFragment newFragment = new
DatePickerFragment(); newFragment.show(getFragmentManager(), "datePicker

");
}

public void DateDayChanger(int daysToChange) { youfreemg.setBackgroundResource(0);
dateChosen.add(Calendar.DATE, daysToChange);

dayOfWeek = dateChosen.getDisplayName(Calendar.DAY_OF_WEEK, Calendar.LONG,
Locale.US);

dayAndDateStatus.setText(dateChosen.getDisplayName(Calendar.DAY_OF_WEEK,
Calendar.LONG, Locale.getDefault()) + ",

```



Цитирования: 0,18%

id: 500



Цитирования: 0,02%

id: 501

```

+ new SimpleDateFormat("
dd.MM.yyyy
Цитирования: 0,15% id: 502
").format(dateChosen.getTime());

lv.setAdapter(null); scheduleList.clear();

new LocalScheduleLoader(TeacherMainScheduleActivity.this, TeacherMainScheduleActivity.this,
dayOfWeek)
.execute();
}

@Override
void getScheduleData(ArrayList<HashMap<String, String> scheduleList) {

this.scheduleList = scheduleList; OnScheduleLoading(); Log.i("
Name:
Цитирования: 0% id: 503
", "
12131313");
}

void OnScheduleLoading() {

if (!scheduleList.isEmpty()) { ScheduleListAdapter scheduleAdapter = new
ScheduleListAdapter(scheduleList,
TeacherMainScheduleActivity.this);

lv.setAdapter(scheduleAdapter);
} else {

dayAndDateStatus.append("\n\n" +
Цитирования: 0,02% id: 504
"Занять немає!\n");
}

youFreeImg.setBackgroundResource(R.drawable.ic_green_happy_man);
}
}

@SuppressWarnings(
Цитирования: 0,01% id: 505
"ValidFragmen")
)

public class DatePickerFragment extends DialogFragment implements
DatePickerDialog.OnDateSetListener {

@Override
public Dialog onCreateDialog(Bundle savedInstanceState) {

final Calendar c = Calendar.getInstance();

int year = c.get(Calendar.YEAR);

int month = c.get(Calendar.MONTH);

int day = c.get(Calendar.DAY_OF_MONTH);

return new DatePickerDialog(getActivity(), this, year, month, day);
}

public void onDateSet(DatePicker view, int year, int month, int day) {

dateChosen.set(year, month, day); DateDayChanger(0);
}
}

}

?php
Schedule.php
header('Content-type: text/plain; charset=utf-8');

if (isset($_POST['ID']) and isset($_POST['UserType'])){

```



```

$response = array();

// include db connect class
require_once DIR. '/connector.php';

// connecting to db

$con=connect();

// echo $con; if ($con!=null) {

// get allfromtable mysqli_set_charset($con,
❏ Цитирования: 0,01% id: 506
"utf8"
);

$D=mysqli_real_escape_string($con,$_POST['ID']); if($_POST['UserType']==
❏ Цитирования: 0,01% id: 507
"1"
)

{

$stmt = mysqli_prepare($con, 'CALL GetStudentSubjects(?)'); mysqli_stmt_bind_param($stmt, 'i',
$D); mysqli_stmt_execute($stmt);

$result = $stmt->get_result();

}

else

{

$stmt = mysqli_prepare($con, 'SELECT * FROM Subject WHERE TeacherID like
? ORDER BY Subject.DayOfWeek'); mysqli_stmt_bind_param($stmt, 'i', $D);
mysqli_stmt_execute($stmt);

$result = $stmt->get_result();

}

close($con);

// check for empty result
if (mysqli_num_rows($result) 0) {

$response[
❏ Цитирования: 0,01% id: 508
"Schedule"
] = array();

$dayName=null;

while ($row = mysqli_fetch_array($result)) { if($dayName==null)

{

$dayName=$row[
❏ Цитирования: 0,02% id: 509
"DayOfWeek"];

$day[$dayName]
=array();

}

else if($dayName!=$row[
❏ Цитирования: 0,01% id: 510
"DayOfWeek"]
)

{

$dayName=$row[
❏ Цитирования: 0,02% id: 511
"DayOfWeek"];

$day[$dayName]
=array();

}

$subject=array();

$subject[
❏ Цитирования: 0,07% id: 512

```

```
"id"=$row["id"];

$subject["Name"]=$row["Name"];

$subject["Type"]=$row["Type"]

;

$subject[
  "Цитирования: 0,02%" id: 513
  "Classroom"=$row["Classroom"]
;

$subject[
  "Цитирования: 0,05%" id: 514
  "DayOfWeek"=$row["DayOfWeek"];
  $subject["Time"]=$row["Time"]
;

$subject[
  "Цитирования: 0,02%" id: 515
  "WeekParty"=$row["WeekParty"]
;

$subject[
  "Цитирования: 0,02%" id: 516
  "Durability"=$row["Durability"]
;

$subject[
  "Цитирования: 0,02%" id: 517
  "TeacherID"=$row["TeacherID"]
; array_push($day[$dayName] , $subject);
}

array_push($response[
  "Цитирования: 0,01%" id: 518
  "Schedule"
], $day);

// success

$response[
  "Цитирования: 0,01%" id: 519
  "success"
] = 1;

// echoing JSON response

echo json_encode($response,JSON_UNESCAPED_UNICODE);

} else {

// no products found

$response[
  "Цитирования: 0,01%" id: 520
  "success"
] = 0;

$response[
  "Цитирования: 0,01%" id: 521
  "message"
] =

  "Цитирования: 0,02%" id: 522
  "User not exist"
;

// echo no users JSON

echo json_encode($response,JSON_UNESCAPED_UNICODE);

}

}

else {

// required field is missing

$response[
  "Цитирования: 0,01%" id: 523
  "success"
] = 0;
```

```
$response[
  "Цитирования: 0,01%" id: 524
  "message"
] =
  "Цитирования: 0,01%" id: 525
  "Connection failed"
;

// echoing JSON response
echo json_encode($response,JSON_UNESCAPED_UNICODE);
}}

else {
// required field is missing

$response[
  "Цитирования: 0,01%" id: 526
  "success"]
= 0;

$response[
  "Цитирования: 0,01%" id: 527
  "message"
] =
  "Цитирования: 0,03%" id: 528
  "Required field ( ) is missing"
;

// echoing JSON response
echo json_encode($response,JSON_UNESCAPED_UNICODE);
}

?

?php
studentList.php

header('Content-type: text/plain; charset=utf-8'); if (isset($_POST['ID'])){

$response = array();

// include db connect class
require_once DIR. '/connector.php';

// connecting to db
$con=connect();

// echo $con; if ($con!=null) {

// get allfromtable mysqli_set_charset($con,
  "Цитирования: 0,01%" id: 529
  "utf8"
);

$ID=mysqli_real_escape_string($con,$_POST['ID']);

$stmt = mysqli_prepare($con, 'CALL GetStudents(?)'); mysqli_stmt_bind_param($stmt, 'i', $ID);
mysqli_stmt_execute($stmt);

$result = $stmt->get_result(); close($con);

// check for empty result
if (mysqli_num_rows($result) 0) {

$response[
  "Цитирования: 0,01%" id: 530
  "Classes"
] = array();

$className=null;

while ($row = mysqli_fetch_array($result)) { if($className==null)

{

$className=$row[
  "Цитирования: 0,02%" id: 531
  "ClassID"];
}
```

```

$Class[$className]
=array();
}

else if($className!=$row[
Цитирования: 0,01% id: 532
"ClassID"]
)
{
$className=$row[
Цитирования: 0,02% id: 533
"ClassID"];
$Class[$className]
=array();
}

$student=array();

$student[
Цитирования: 0,1% id: 534
"ID"]=$row["ID"];

$student["Name"]=$row["Name"];

$student["Surname"]=$row["Surname"];

$student["Email"]=$row["Email"]
; array_push($Class[$className] , $student );
}

array_push($response[
Цитирования: 0,01% id: 535
"Classes"
] , $Class);

// success

$response[
Цитирования: 0,01% id: 536
"success"
]= 1;

// echoing JSON response

echo json_encode($response,JSON_UNESCAPED_UNICODE);

} else {

// no found

$response[
Цитирования: 0,01% id: 537
"success"
]= 0;

$response[
Цитирования: 0,01% id: 538
"message"
]=
Цитирования: 0,02% id: 539
"User not exist"
;

// echo no users JSON

echo json_encode($response,JSON_UNESCAPED_UNICODE);

}

}

else {

// required field is missing

$response[
Цитирования: 0,01% id: 540
"success"
]= 0;

$response[

```

```

"    Цитирования: 0,01%                                     id: 541
] =
"    Цитирования: 0,01%                                     id: 542
"connection failed"
;

// echoing JSON response
echo json_encode($response,JSON_UNESCAPED_UNICODE);
}}
else {
// required field is missing

$response[
"    Цитирования: 0,01%                                     id: 543
"success"]
= 0;

$response[
"    Цитирования: 0,01%                                     id: 544
"message"
] =
"    Цитирования: 0,03%                                     id: 545
"required field() is missing"
;

// echoing JSON response
echo json_encode($response,JSON_UNESCAPED_UNICODE);
}
?
?php
Class_subject.php

header('Content-type: text/plain; charset=utf-8'); if (isset($_POST['ID'])){
$response = array();

// include db connect class
require_once DIR. '/connector.php';

// connecting to db

$con=connect();

// echo $con; if ($con!=null) {

// get allfromtable

mysqli_set_charset($con,
"    Цитирования: 0,01%                                     id: 546
"utf8"
);

$ID=mysqli_real_escape_string($con,$_POST['ID']);

$stmt = mysqli_prepare($con, 'CALL GetClassSubjects(?)'); mysqli_stmt_bind_param($stmt, 'i',
$ID); mysqli_stmt_execute($stmt);

$result = $stmt- get_result(); close($con);

// check for empty result

if (mysqli_num_rows($result) 0) {

$response[
"    Цитирования: 0,01%                                     id: 547
"ClassSubjects"
] = array();

while ($row = mysqli_fetch_array($result)) {

$classSubject=array();

$classSubject[
"    Цитирования: 0,02%                                     id: 548
"classID"]=$row["classID"]
;

$classSubject[

```

```
"Citirovaniya": 0,02% id: 549
    "SubjectID"=$obj["SubjectID"]
;

$ClassSubject[
    "Citirovaniya": 0,02% id: 550
    "SubjectID"=$obj["SubjectID"]
; array_push($response[
    "Citirovaniya": 0,01% id: 551
    "ClassSubjects"
] , $ClassSubject);
}

// success

$response[
    "Citirovaniya": 0,01% id: 552
    "success"
] = 1;

// echoing JSON response

echo json_encode($response,JSON_UNESCAPED_UNICODE);

} else {

// no found

$response[
    "Citirovaniya": 0,01% id: 553
    "success"
] = 0;

$response[
    "Citirovaniya": 0,01% id: 554
    "message"
] =
    "Citirovaniya": 0,02% id: 555
    "User not exist"
;

// echo no users JSON

echo json_encode($response,JSON_UNESCAPED_UNICODE);

}

}

else {

// required field is missing

$response[
    "Citirovaniya": 0,01% id: 556
    "success"
] = 0;

$response[
    "Citirovaniya": 0,01% id: 557
    "message"
] =
    "Citirovaniya": 0,01% id: 558
    "Connection failed"
;

// echoing JSON response

echo json_encode($response,JSON_UNESCAPED_UNICODE);

}}

else {

// required field is missing

$response[
    "Citirovaniya": 0,01% id: 559
    "success"
] = 0;

$response[
    "Citirovaniya": 0,01% id: 560
    "message"
] =
```

```

Цитирования: 0,03% id: 561
"Required data is missing"
;

// echoing JSON response
echo json_encode($response,JSON_UNESCAPED_UNICODE);
}
?
?php
statisticForTeacher.php
header('Content-type: text/plain; charset=utf-8'); if (isset($_POST['SubjectsAndGroupsList'])){
$response = array();
$sonData = array();
$SubjectsAndGroupsList=json_decode($_POST['SubjectsAndGroupsList'], true);
require_once DIR.'/connector.php';
$con=connect();
$absent=0;
$present=0;
$late=0;
if ($con!=null) { mysqli_set_charset($con,
Цитирования: 0,01% id: 562
"utf8"
);
foreach($SubjectsAndGroupsList as $subjectAndGroupe)
{
$stmt= mysqli_prepare($con, 'CALL GetStatForTeacher(?,?)'); mysqli_stmt_bind_param($stmt,
'ii',
$subjectAndGroupe[
Цитирования: 0,01% id: 563
"SubjectID"]
,$subjectAndGroupe[
Цитирования: 0,01% id: 564
"ClassID"
]);
$stmt->execute();
$result = $stmt->get_result();
$row=mysqli_fetch_assoc($result);
$absent+=$row['Absent'];
$present+=$row['Present'];
$late+=$row['Late'];
$stmt->close();
}
close($con);
$stat=$absent.
Цитирования: 0% id: 565
",."
$present.
Цитирования: 0% id: 566
",."
$late;
$response[
Цитирования: 0,01% id: 567
"success"]
= 1;
$response[
Цитирования: 0,01% id: 568
"message"
] =

```

```
Цитирования: 0,03% id: 569
"Transaction successful";

$response["stat"]

=$stat;

echo json_encode($response,JSON_UNESCAPED_UNICODE);

}

else {

$response[
Цитирования: 0,01% id: 570
"success"]
= 0;

$response[
Цитирования: 0,01% id: 571
"message"
] =
Цитирования: 0,01% id: 572
"Connection failed "
;

echo json_encode($response,JSON_UNESCAPED_UNICODE);

}}

else {

$response[
Цитирования: 0,01% id: 573
"success"]
= 0;

$response[
Цитирования: 0,01% id: 574
"message"
] =
Цитирования: 0,03% id: 575
"Required field (.) is missing"
;

echo json_encode($response,JSON_UNESCAPED_UNICODE);

}

?




Help_activity.xml

RelativeLayout xmlns:android=
Цитирования: 0,02% id: 576
"http://schemas.android.com/apk/res/android"
xmlns:tools=
Цитирования: 0,02% id: 577
"http://schemas.android.com/tools"
android:layout_width=
Цитирования: 0,01% id: 578
"match_parent"
android:layout_height=
Цитирования: 0,01% id: 579
"match_parent"
tools:context=
Цитирования: 0,01% id: 580
"${relativePackage}.${activityClass}"

ScrollView

xmlns:android=
Цитирования: 0,02% id: 581
"http://schemas.android.com/apk/res/android"
android:layout_width=
Цитирования: 0,01% id: 582
"fill_parent"
android:layout_height=
Цитирования: 0,01% id: 583
"fill_parent"

android:fillViewport=
```


 Цитирования: 0,01%	id: 584
"true"	
!--IMPORTANT otherwise backgrnd img. will not fill the whole screen --	
RelativeLayout android:layout_width=	
 Цитирования: 0,01%	id: 585
"match_parent"	
android:layout_height=	
 Цитирования: 0,01%	id: 586
"match_parent"	
tools:context=	
 Цитирования: 0,01%	id: 587
"\${relativePackage}.\${activityClass}"	
TextView	
android:id=	
 Цитирования: 0,01%	id: 588
"@+id/mainWindowTv"	
android:layout_width=	
 Цитирования: 0,01%	id: 589
"wrap_content"	
android:layout_height=	
 Цитирования: 0,01%	id: 590
"wrap_content"	
android:layout_centerHorizontal=	
 Цитирования: 0,01%	id: 591
"true"	
android:gravity=	
 Цитирования: 0,01%	id: 592
"center"	
android:text=	
 Цитирования: 0,01%	id: 593
"@string/mainWindowHelpTitle"	
android:textSize=	
 Цитирования: 0,01%	id: 594
"22sp"	
/	
TextView	
android:id=	
 Цитирования: 0,01%	id: 595
"@+id/mainWindowTex1"	
android:layout_width=	
 Цитирования: 0,01%	id: 596
"wrap_content"	
android:layout_height=	
 Цитирования: 0,01%	id: 597
"wrap_content"	
android:layout_below=	
 Цитирования: 0,01%	id: 598
"@+id/mainWindowTv"	
android:layout_centerHorizontal=	
 Цитирования: 0,01%	id: 599
"true"	
android:layout_marginLeft=	
 Цитирования: 0,01%	id: 600
"20sp"	
android:layout_marginRight=	
 Цитирования: 0,01%	id: 601
"20sp"	
android:gravity=	
 Цитирования: 0,01%	id: 602
"fill_horizontal"	
android:text=	
 Цитирования: 0,01%	id: 603
"@string/mainWindowTex1"	
android:textSize=	
 Цитирования: 0,01%	id: 604
"18sp"	
/	

ImageView		
android:id=		
Цитирования: 0,01%		id: 605
@+id/mainWindowimg1		
android:layout_width=		
Цитирования: 0,01%		id: 606
match_parent		
android:layout_height=		
Цитирования: 0,01%		id: 607
wrap_content		
android:layout_below=		
Цитирования: 0,01%		id: 608
@+id/mainWindowTex1		
android:src=		
Цитирования: 0,01%		id: 609
@drawable/main_window_schedule		
/		
TextView		
android:id=		
Цитирования: 0,01%		id: 610
@+id/mainWindowTex2		
android:layout_width=		
Цитирования: 0,01%		id: 611
wrap_content		
android:layout_height=		
Цитирования: 0,01%		id: 612
wrap_content		
android:layout_below=		
Цитирования: 0,01%		id: 613
@+id/mainWindowimg1		
android:layout_marginLeft=		
Цитирования: 0,01%		id: 614
20dp		
android:layout_marginRight=		
Цитирования: 0,01%		id: 615
20dp		
android:layout_centerHorizontal=		
Цитирования: 0,01%		id: 616
true		
android:gravity=		
Цитирования: 0,01%		id: 617
fill_horizontal		
android:text=		
Цитирования: 0,01%		id: 618
@string/mainWindowTex2		
android:textSize=		
Цитирования: 0,01%		id: 619
18sp		
/		
ImageView		
android:id=		
Цитирования: 0,01%		id: 620
@+id/mainWindowimg2		
android:layout_width=		
Цитирования: 0,01%		id: 621
match_parent		
android:layout_height=		
Цитирования: 0,01%		id: 622
wrap_content		
android:layout_below=		
Цитирования: 0,01%		id: 623
@+id/mainWindowTex2		
android:src=		
Цитирования: 0,01%		id: 624
@drawable/main_window_schedule_navigation		
/		
/RelativeLayout		
/ScrollView		
/RelativeLayout		

List_loader_for_statistic.xml	
RelativeLayout xmlns:android=	
Цитирования: 0,02%	id: 625
"http://schemas.android.com/apk/res/android"	
xmlns:tools=	
Цитирования: 0,02%	id: 626
"http://schemas.android.com/tools"	
android:layout_width=	
Цитирования: 0,01%	id: 627
"match_parent"	
android:layout_height=	
Цитирования: 0,01%	id: 628
"match_parent"	
tools:context=	
Цитирования: 0,01%	id: 629
"\${relativePackage}.\${activityClass}"	
ListView	
android:id=	
Цитирования: 0,01%	id: 630
"@+id/list"	
android:background=	
Цитирования: 0,01%	id: 631
"@drawable/lv_background"	
android:layout_width=	
Цитирования: 0,01%	id: 632
"fill_parent"	
android:layout_height=	
Цитирования: 0,01%	id: 633
"wrap_content"	
android:layout_marginTop=	
Цитирования: 0,01%	id: 634
"16"	
android:layout_marginLeft=	
Цитирования: 0,01%	id: 635
"16"	
android:layout_marginRight=	
Цитирования: 0,01%	id: 636
"16"	
/	
Button	
android:id=	
Цитирования: 0,01%	id: 637
"@+id/bt"	
android:onClick=	
Цитирования: 0,01%	id: 638
"onOKStart"	
android:text=	
Цитирования: 0,01%	id: 639
"Close"	
android:textSize=	
Цитирования: 0,01%	id: 640
"16"	
android:layout_alignParentBottom=	
Цитирования: 0,01%	id: 641
"true"	
android:layout_centerHorizontal=	
Цитирования: 0,01%	id: 642
"true"	
android:layout_width=	
Цитирования: 0,01%	id: 643
"fill_parent"	
android:layout_height=	
Цитирования: 0,01%	id: 644
"wrap_content"	
/	
/RelativeLayout	

activity_login.xml		
RelativeLayout xmlns:android=		
Цитирования: 0,02%	id: 645	
"http://schemas.android.com/apk/res/android"		
xmlns:tools=		
Цитирования: 0,02%	id: 646	
"http://schemas.android.com/tools"		
android:layout_width=		
Цитирования: 0,01%	id: 647	
"match_parent"		
android:layout_height=		
Цитирования: 0,01%	id: 648	
"match_parent"		
android:background=		
Цитирования: 0,01%	id: 649	
"@drawable/bg"		
tools:context=		
Цитирования: 0,01%	id: 650	
"\${relativePackage}.\${activityClass}"		
Button		
android:id=		
Цитирования: 0,01%	id: 651	
"@+id/btn"		
android:layout_width=		
Цитирования: 0,01%	id: 652	
"50dp"		
android:layout_height=		
Цитирования: 0,01%	id: 653	
"60dp"		
android:layout_alignParentRight=		
Цитирования: 0,01%	id: 654	
"true"		
android:background=		
Цитирования: 0,01%	id: 655	
"@android:color/transparent"		
/		
TextView		
android:id=		
Цитирования: 0,01%	id: 656	
"@+id/enter_email"		
android:layout_width=		
Цитирования: 0,01%	id: 657	
"wrap_content"		
android:layout_height=		
Цитирования: 0,01%	id: 658	
"wrap_content"		
android:layout_centerHorizontal=		
Цитирования: 0,01%	id: 659	
"true"		
android:gravity=		
Цитирования: 0,01%	id: 660	
"center"		
android:text=		
Цитирования: 0,04%	id: 661	
"Введіть Ваш емейл, будь ласка :)"		
android:textSize=		
Цитирования: 0,01%	id: 662	
"32dp"		
android:layout_marginTop=		
Цитирования: 0,01%	id: 663	
"170dp"		
android:layout_marginBottom=		
Цитирования: 0,01%	id: 664	
"20dp"		
/		

EditText		
android:id=	Цитирования: 0,01%	id: 665
@+id/		
android:layout_above=	Цитирования: 0,01%	id: 666
@+id/		
android:layout_marginBottom=	Цитирования: 0,01%	id: 667
120dp		
android:layout_width=	Цитирования: 0,01%	id: 668
match_parent		
android:layout_height=	Цитирования: 0,01%	id: 669
wrap_content		
android:hint=	Цитирования: 0,01%	id: 670
@string/enterID		
android:inputType=	Цитирования: 0,01%	id: 671
textEmailAddress		
/		
Button		
android:id=	Цитирования: 0,01%	id: 672
@+id/		
android:layout_alignParentBottom=	Цитирования: 0,01%	id: 673
true		
android:layout_centerHorizontal=	Цитирования: 0,01%	id: 674
true		
android:layout_width=	Цитирования: 0,01%	id: 675
110dp		
android:layout_height=	Цитирования: 0,01%	id: 676
114dp		
android:background=	Цитирования: 0,01%	id: 677
@android:color/transparent		
/		
/RelativeLayout		
Teacher_main_schedule_activity.xml		
RelativeLayout xmlns:android=	Цитирования: 0,02%	id: 678
http://schemas.android.com/apk/res/android		
xmlns:tools=	Цитирования: 0,02%	id: 679
http://schemas.android.com/tools		
android:id=	Цитирования: 0,01%	id: 680
@+id/rllayout		
android:layout_width=	Цитирования: 0,01%	id: 681
match_parent		
android:layout_height=	Цитирования: 0,01%	id: 682
match_parent		
android:background=	Цитирования: 0,01%	id: 683
@drawable/lotopay		
tools:context=	Цитирования: 0,01%	id: 684
\${relativePackage}.\${activityClass}		
RelativeLayout android:layout_width=	Цитирования: 0,01%	id: 685

"match_parent"	
android:layout_height=	
Цитирования: 0,01%	id: 686
"wrap_content"	
android:orientation=	
Цитирования: 0,01%	id: 687
"horizontal"	
android:layout_marginLeft=	
Цитирования: 0,01%	id: 688
"35dp"	
android:layout_marginTop=	
Цитирования: 0,01%	id: 689
"30dp"	
android:layout_marginRight=	
Цитирования: 0,01%	id: 690
"3dp"	
TextView android:id=	
Цитирования: 0,01%	id: 691
"@+id/day_of_week"	
android:layout_width=	
Цитирования: 0,01%	id: 692
"wrap_content"	
android:layout_height=	
Цитирования: 0,01%	id: 693
"wrap_content"	
android:gravity=	
Цитирования: 0,01%	id: 694
"center"	
android:layout_centerHorizontal=	
Цитирования: 0,01%	id: 695
"true"	
android:layout_marginBottom=	
Цитирования: 0,01%	id: 696
"35dp"	
android:textSize=	
Цитирования: 0,01%	id: 697
"28dp"	
/TextView	
ListView	
android:id=	
Цитирования: 0,01%	id: 698
"@+id/schedule"	
android:background=	
Цитирования: 0,01%	id: 699
"@drawable/ic_background"	
android:layout_below=	
Цитирования: 0,01%	id: 700
"@+id/day_of_week"	
android:layout_width=	
Цитирования: 0,01%	id: 701
"fill_parent"	
android:layout_height=	
Цитирования: 0,01%	id: 702
"wrap_content"	
android:layout_marginRight=	
Цитирования: 0,01%	id: 703
"5dp"	
/	
ImageView	
android:id=	
Цитирования: 0,01%	id: 704
"@+id/coufreimage"	
android:layout_below=	
Цитирования: 0,01%	id: 705
"@+id/day_of_week"	
android:layout_width=	

<div><div>»</div><div>Цитирования: 0,01%</div><div>id: 706</div></div>
<div><div>"wrap_content"</div></div>
<div>android:gravity=</div>
<div><div>»</div><div>Цитирования: 0,01%</div><div>id: 707</div></div>
<div><div>"center"</div></div>
<div>android:layout_centerHorizontal=</div>
<div><div>»</div><div>Цитирования: 0,01%</div><div>id: 708</div></div>
<div><div>"true"</div></div>
<div>android:layout_height=</div>
<div><div>»</div><div>Цитирования: 0,01%</div><div>id: 709</div></div>
<div><div>"wrap_content"</div></div>
<div>android:layout_marginLeft=</div>
<div><div>»</div><div>Цитирования: 0,01%</div><div>id: 710</div></div>
<div><div>"35dp"</div></div>
<div>android:layout_marginRight=</div>
<div><div>»</div><div>Цитирования: 0,01%</div><div>id: 711</div></div>
<div><div>"5dp"</div></div>
<div>/</div>
<div>/RelativeLayout</div>
<div>/RelativeLayout</div>
<div>1</div>
<div>110</div>
<div>Збір даних про відвідуваність</div>
<div>Обробка та збереження даних в локальному сховищі даних</div>
<div>Збереження даних в глобальній БД</div>
<div>Процес збору даних про відвідуваність</div>
<div>Перетворення даних у потрібний для аналізу вигляд</div>
<div>Аналіз даних за потрібним критерієм</div>
<div>Збереження результатів аналізу</div>
<div>Процес обробки та аналізу даних</div>
<div>Виявлення проблеми з відвідуваністю</div>
<div>Внесення до списку студентів в яких проблеми з відвідувністю</div>
<div>Інформування студентів про те що вони у списках</div>
<div>Процес повідомлення про проблеми з відвідуваністю</div>
<div>Додаток</div>
<div><div>»</div><div>Цитирования: 0,01%</div><div>id: 712</div></div>
<div><div>"Student Attendance"</div></div>
<div><div>RESTful API</div></div>
<div><div>Android ОС</div></div>
<div><div>GCM API</div></div>
<div><div>MySQL БД</div></div>
<div><div>GCM</div></div>
<div><div>Веб-сервер</div></div>
<div><div>GetStatisticData.java</div></div>
<div><div>GetStatisticData</div></div>
<div><div>MainActivity.java</div></div>
<div><div>OnSwipeTouchListener.java</div></div>