ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

ІС – інформаційна система;

ІТ – інформаційні технології;

БД – база даних;

ПЗ – програмне забезпечення;

ЦОД – центр обробки даних;

API – (Application Programming Interface) – прикладний програмний інтерфейс;

ASP – (Active Server Pages) – активні серверні сторінки;

CLI – (Command Line Interface) – інтерфейс командного рядка;

CSS – (Cascading Style Sheets) – каскадна таблиця стилів;

DOM – (Document Object Model) – об’єктна модель документа;

IaaS – Infrastructure as a Service;

JS – Java Script;

JSON – (Java Script Object Notation) – запис об’єктів Java Script;

HTML – (Hyper Text Markup Language) – мова гіпертекстової розмітки;

PaaS – Platform as a Service;

PHP – (Hypertext Preprocessor) – гіпертекстовий препроцесор;

SaaS – Software as a Service;

SOA – Service-Oriented Architecture;

SQL – (Structured Query Language) – мова структурованих запитів;

XML – (Extensible Markup Language) – розширювана мова розмітки;

URL – (Uniform Resource Locator) – єдиний вказівник на ресур.

ВСТУП

Суть концепції хмарних технологій полягає в наданні кінцевим користувачам віддаленого динамічного доступу до послуг, обчислювальним ресурсів і додатків через Інтернет. Розвиток сфери хостингу було обумовлено виниклою потребою в програмному забезпеченні і цифрових послугах, якими можна було б управляти зсередини, але які були б при цьому більш економічними і ефективними за рахунок економії на масштабі. Більшість сервіс-провайдерів пропонують хмарні обчислення у формі VPS-хостингу, віртуального хостингу, і ПЗ-як-послуга (SaaS). Хмарні послуги довгий час надавалися у формі SaaS, наприклад, Microsoft Hosted Exchange і SharePoint. Не можна не визнати, що технології хмарних обчислень мають величезний потенціал, тому що всі сучасні комп'ютерні продукти постійно збільшують свої вимоги до технічного оснащення комп'ютера користувача, що неминуче веде до значних витрат на апгрейд. Ця технологія дозволяє вирішити проблему надмірної вимогливості додатків до ресурсів кінцевого користувача. Концепція хмарних обчислень значно змінила традиційний підхід до доставки, управління та інтеграції додатків. У порівнянні з традиційним підходом, хмарні обчислення дозволяють управляти більшими інфраструктурами, обслуговувати різні групи користувачів в межах однієї хмари, а також означають повну залежність від провайдера хмарних послуг. Однак дана залежність є такою лише в теорії, адже якщо компанія-провайдер допустить хоч один прецедент крадіжки інформації, це стане колосальним ударом по всій індустрії надання віддалених потужностей [1][1)].

Хмарні обчислення – це ефективний інструмент підвищення прибутку і розширення каналів продажів для незалежних виробників програмного забезпечення (ISV), операторів зв'язку і VAR-посередників (у формі SaaS). Цей підхід дозволяє організувати динамічне надання послуг, коли користувачі

[1)] [1] Облачные вычисления: что же это такое? URL: https://www.itweek.ru/its/article/detail.php?ID=135408 (дата звернення: 25.08.2019).

можуть проводити оплату за фактом і регулювати обсяг своїх ресурсів залежно від реальних потреб без довгострокових зобов'язань. Найголовнішою перевагою застосування є відсутність необхідності мати потужну систему у кінцевого користувача, що однозначно веде до вагомого зниження витрат для користувача. Другим плюсом можна назвати неможливість використання піратського контенту, адже весь вхідний трафік буде виходити від сертифікованих провайдерів. Таким чином можна вирішити одну з глобальних проблем комп'ютерної сучасності – піратство [2][1)].

Роль спорту в житті людини є найголовнішим атрибутом в способі життя людей, які стежать за здоров'ям і хочуть зберегти свою красу і привабливість тіла на довгі роки. Займаючись спортом людина не тільки отримаєте стрункy і красиву фігуру, але й колосальне здоров'я. Одні дуже прохолодно ставляться до спорту і деякі навіть вважають його марною тратою часу. Інші люди бачать сенс спорту і їх, на щастя, більше ніж перших. Причому кожен з них може мати різне ставлення до спорту: хтось воліє дивитися його по телевізору, хтось віддає перевагу просто займатися яким-небудь видом спорту або загальнофізичною підготовкою, ну а для когось спорт – це засіб існування. Серед останніх можуть бути діючі спортсмени, тренери, лікарі, директори різних спортивних товариств, піклувальники спорту та ін. Спорт має дивовижні властивості. Він може об'єднувати людей, знайомити їх між собою, в більшості випадків спорт зміцнює здоров'я, характер і навіть розумові здібності людей [3][2)].

В сучасному світі розвитку мобільних і бездротових технологій для інформаційної підтримки занять спортом кожної зацікавленої людини доцільно використання додатків, що забезпечать організацію тренувань, надання консультацій професійним тренером що до навантаження тренувань. Використання таких додатків значно допоможуть людині займатися спортом і під-

---

[1)] [2] Облачные вычисления: тенденции развития и основные «игроки». Часть 2. URL: https://npsod.ru/analytics/1956.html (дата звернення: 05.09.2019).

[2)] [3] Про фізичну культуру і спорт від 24.12.1993 № 3808-XII URL: https://zakon.rada.gov.ua/laws/main/3808-12 (дата звернення: 07.09.2019).

тримувати спортивну форму без очного відвідування фітнес-центрів. Тому завдання магістерської роботи що до розробки та проектування хмарного сервісу «Фітнес-студія» є актуальним завданням, яке спрямовано на забезпечення користувачів зручним додатком для занять спортом.

Метою магістерської роботи є створення хмарного сервісу «Фітнес-студія», який реалізує як локальні функції з організації тренувань на базі OS Android, так і можливість реалізації серверних функцій зберігання інформації, вибору та участі в фітнес групах, отримання консультацій тренера.

Для досягнення мети необхідно вирішити наступні завдання:

- провести огляд сучасних тенденцій розвитку хмарних технологій;
- провести аналіз та дослідження застосування моделей хмарних технологій;
- розробити структуру хмарного сервісу «Фітнес-студія», обравши відповідний сервіс для реалізації;
- розглянути аналогічні мобільні та мережеві додатки, визначити їх переваги і недоліки, сформулювати вимоги до функціональних можливостей сервісу;
- здійснити вибір середовища та інструментів розробки сервісу, визначити механізми реалізації клієнт-серверного спілкування;
- здійснити проектування та розробку графічного інтерфейсу та функцій розділів мобільного додатку сервісу «Фітнес-студія» та реалізувати функції підсистеми тренера в середовищі Eclipse з доповненням Android SDK, на мові Java.

Практична цінність магістерської роботи полягає в тому, що розроблений хмарний сервіс «Фітнес-студія» може бути використаний для проведення занять спортом як тренерами так і всіма зацікавленими у тренуваннях клієнтами з можливістю отримання всієї необхідної інформації що до занять.

Магістерська робота містить 84 сторінки, 1 таблицю, 38 рисунків, 12 посилань.

1 ДОСЛІДЖЕННЯ ТА АНАЛІЗ ЗАСТОСУВАННЯ МОДЕЛЕЙ ХМАРНИХ ТЕХНОЛОГІЙ

1.1 Основні поняття хмарних обчислень

Хмарні обчислення є комплексним рішенням, яке надає IT-ресурси у вигляді сервісів. Це рішення ґрунтоване на Інтернет-технологіях. У хмарі комп'ютери налаштовані на спільну роботу, а різні застосування використовують загальну обчислювальну потужність, неначе виконуються в єдиній системі. Ідея хмарних обчислень полягає в тому, що компанії, що використовують IT-послуги, можуть купувати ці послуги як сервіси. Замість того щоб купувати сервери для підтримки внутрішніх або зовнішніх сервісів і придбавати ліцензії на програмне забезпечення, компанія може купити їх як сервіс. Для переходу до хмарних обчислень є вагомі причини. На користь вибору хмарних технологій виступають такі аргументи, як зменшення витрат, оптимальність використання персоналу, надійність при масштабованості. Специфіка хмарних технологій полягає в тому, що їх застосування створює додаткові риски і уразливості [2][1)]:

- зберігання конфіденційних даних на стороні провайдера послуг;
- залежність від швидкості і якості з'єднання з мережею Інтернет;
- складність прогнозування робочого навантаження в пікові години з боку клієнтів і, як наслідок, небезпеку виникнення мережевих штормів і сервісної відмови в обслуговуванні.

Модель хмарних технологій складається з внутрішньої і зовнішньої частин. Вони сполучені в мережі, як правило, через Інтернет. Самою хмарою виступає внутрішня частина. Зовнішня частина включає клієнтський комп'ютер або мережі комп'ютерів організації і додатків, які використовуються для доступу в хмару. Внутрішня частина надає комп'ютери, додатки, сховища даних і сервери, що створюють хмарні сервіси.

[1)] [2] Облачные вычисления: тенденции развития и основные «игроки». Часть 2. URL: https://npsod.ru/analytics/1956.html (дата звернення: 05.09.2019).

Концепція «хмари» характеризується рівнями, які мають свою певну функціональність Хмара надає наступні рівні. Перший рівень – інфраструктура (Infrastructure as a Service – IaaS), яка є основою хмарних обчислень. Рівень складається з фізичних активів – мережевих пристроїв, серверів, дисків і т. д. Користувач насправді не управляє базовою інфраструктурою при взаємодії з IaaS, проте управляє сховищами даних, операційною системою, розгортаними застосуваннями і вибраними мережевими компонентами. Проміжним рівнем є платформа (Platform as a Service – PaaS). Вона є інфраструктурою додатків. PaaS дозволяє надавати доступ до операційної системи і відповідних сервісів і розгортати додатки в хмарі за допомогою інструментальних засобів. Верхній рівень – рівень додатків (Software as a Service – SaaS), при якому постачальник розробляє веб-застосування і самостійно управляє ним, надаючи замовникові доступ до програмного забезпечення через Інтернет [4][1)]. Ці усі рівні представлені на рис. 1.1.

Рисунок 1.1 – Рівні хмарних обчислень

Технології рівнів при спільному використанні дозволяють користувачам хмарних обчислень скористатися обчислювальними потужностями і схо-

[1)] [4] Разработка ПО (мировой рынок). URL: http://www.tadviser.ru/index.php/Статья:Разработка_ПО_%28мировой_рынок%29 (дата звернення: 10.10.2019).

вищами даних, які за допомогою певних технологій віртуалізації і високого рівня абстракції надаються ним як послуги. На перший план серед аргументів переходу в хмари для клієнтів виходить забезпечення надійної підтримки і хостингу додатка – з відповідним рівнем відповідальності по SLA (Service Layer Agreement). Окрім різних рівнів надання сервісів розрізняють декілька варіантів розгортання хмарних систем. Приватна хмара (private cloud) – для надання сервісів усередині однієї компанії, яка являється одночасно і замовником і постачальником послуг. Публічна хмара (public cloud) – використовується хмарними провайдерами для надання сервісів зовнішнім замовникам. Змішана хмара (hybrid cloud) – спільне використання двох вищеперелічених моделей розгортання. Це поєднання приватної і публічної хмари, в якій використовуються сервіси, розташовані як в закритому, так і у відкритому просторі [4][1)]. За управління такими сервісами відповідальність розподіляється між підприємством і провайдером відкритої хмари (рис .1.2).

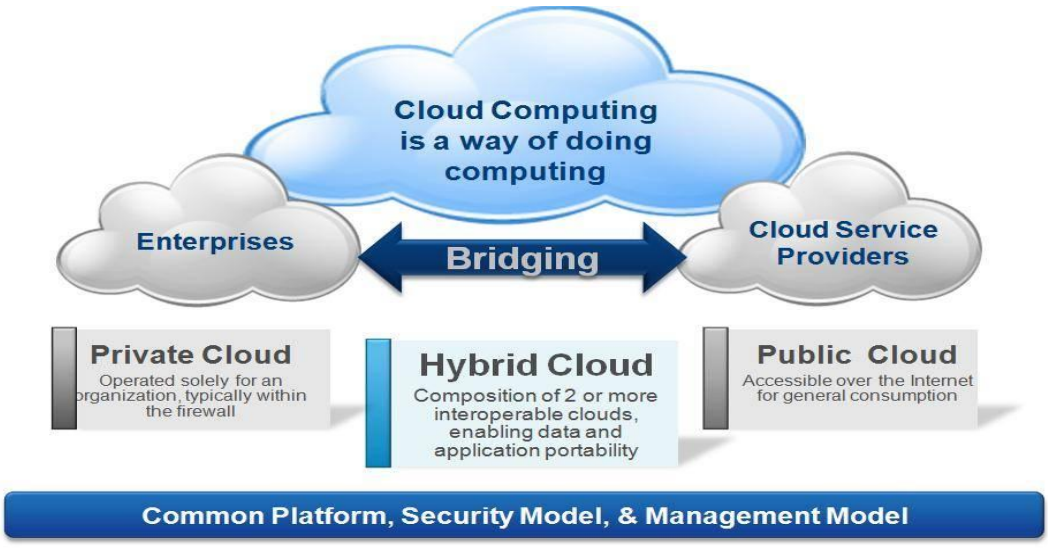


Рисунок 1.2 – Структура гібридної хмари

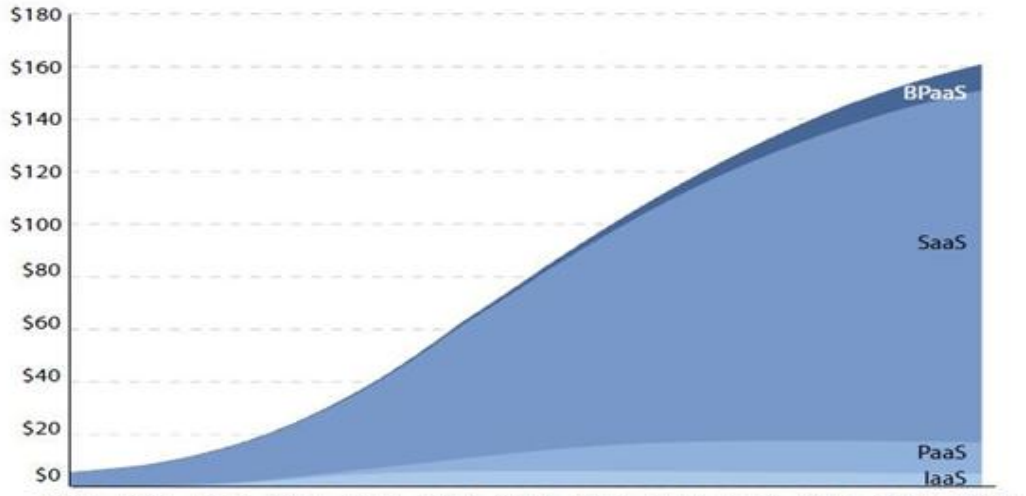
При використанні гібридної хмари підприємства визначають цілі і вимоги до хмарних сервісів, вибираючи більше відповідний варіант. Одна з

[1)] [4] Разработка ПО (мировой рынок). URL: http://www.tadviser.ru/index.php/Статья:Разработка_ПО_%28мировой_рынок%29 (дата звернення: 10.10.2019).

ключових ідей хмарних технологій полягає якраз в тому, щоб з технологічної точки зору різниці між внутрішніми і зовнішніми хмарами не було і замовник міг гнучко переміщати свої завдання між власною і орендованою IT-інфраструктурою. Хмарні обчислення є ефективним інструментом для підвищення прибутковості компанії та розширення продажу для виробників ПЗ.

1.2 Огляд та аналіз сучасних тенденцій розвитку ринку хмарних послуг

Аналітична компанія Forrester Research опублікувала прогноз розвитку ринку публічних хмарних обчислень до 2020 року. Згідно зі звітом, до 2020 року об'єм ринку хмарних послуг складе $160 млрд [4][1)]. На рис. 1.3 представлені прогноз зростання об'єму ринку cloud computing.



	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020
BPaaS ($)	0.15	0.23	0.35	0.53	0.80	1.26	1.95	2.93	4.28	6.00	7.66	9.08	10.02
SaaS ($)	5.56	8.09	13.40	21.21	33.09	47.22	63.19	78.43	92.75	105.49	116.39	125.52	132.57
PaaS ($)	0.05	0.12	0.31	0.82	2.08	4.38	7.39	9.80	11.26	11.94	12.15	12.10	11.91

Рисунок 1.3 – Прогноз зростання об'єму ринку cloud computing по сегментах

[1)] Разработка ПО (мировой рынок). URL: http://www.tadviser.ru/index.php/Статья:Разработка_ПО_%28мировой_рынок%29 (дата звернення: 10.10.2019).

За даними компанії International Data Corporation (IDC), що займається аналітикою, во всьому світі 94% компаній активно готуються до сучасних реалій, орієнтації на хмарні технології: вони займаються або переглядом і трансформацією, або модернізацією своїх мережевих інфраструктур, щоб спростити доставку додатків. В ході дослідження аналітики IDC провели опитування представників 900 великих компаній по всьому світу і сформулювали наступні тези [4][1)]:

- тільки 24% організацій упевнені, що їх мережі повністю задовольняють вимогам, необхідним для перенесення ПЗ в гібридну хмару;
- 82% опитаних компаній вважають, що їх поточна архітектура мережі занадто складна для перенесення додатків в хмару;
- 76% респондентів повідомили, що їх мережі повністю не готові (36%) або частково не готові (40%);
- 70% учасників опитування мають 2-3 способи віддаленого доступу до своїх дата-центрів і хмар.

Аналітики з'ясували, що основні цілі переходу від локальної моделі до концепції доставки додатків на призначені для користувача пристрої через хмару – підвищення безпеки і зниження витрат, у тому числі за рахунок оптимізації роботи IT-відділів і фахівців. Поступово гібридні і мультихмарні технології стають «новою нормою» для компаній, які впроваджують хмарні середовища. Проте сам процес адаптації наявної архітектури під доставку додатків може виявитися досить складним і дорогим. 12 квітня 2018 року аналітики Gartner оприлюднили результати дослідження світового ринку публічних хмар. Витрати на ці рішення підвищилися більш ніж на 30% завдяки інфраструктурним сервісам (IaaS) і програмному забезпеченню, що надається в якості послуги (SaaS) [2][2)].

---

1) [4] Разработка ПО (мировой рынок). URL: http://www.tadviser.ru/index.php/Статья:Разработка_ПО_%28мировой_рынок%29 (дата звернення: 10.10.2019).

2) [2] Облачные вычисления: тенденции развития и основные «игроки». Часть 2. URL: https://npsod.ru/analytics/1956.html (дата звернення: 05.09.2019)..

У 2017 році сумарні витрати споживачів і компаній на хмарні сервіси, що надаються у рамках публічного доступу, склали $153, 5 млрд. проти $118 млрд. роком раніше (табл.1.1).

Таблиця 1.1 – Сегменти ринку публічних хмар, дані Gartner (прогноз)

	2017	2018	2019	2020	2021
Cloud Business Process Services (BPaaS)	42,6	46,4	50,1	54,1	58,4
Cloud Application Infrastructure Services (PaaS)	11,9	15,0	18,6	22,7	27,3
Cloud Application Services(SaaS)	60,2	73,6	87,2	101,9	117,1
Cloud Management and Security Services	8,7	10,5	12,3	14,1	16,1
Cloud System Infrastructure Services (IaaS)	30,0	40,8	52,9	67,4	83,5
Total Market	153,5	186,4	221,1	260,2	302,5

Продажі рішень IaaS підскочили з 25,3 до 30 млрд. доларів. Сегменти SaaS і PaaS показали підйом з $38, 6 млрд. і $7, 2 млрд. до $60, 2 млрд. і $11, 9 млрд. відповідно. Витрати на BPaaS-сервіси (бізнес-процеси як послуга) і послуги хмарного управління і безпеки в 2017 році досягли 42,6 і 8,7 млрд. відповідно. Найбільшим сегментом ринку публічних хмар залишається SaaS, який, за прогнозами Gartner, в 2018 році підвищиться на 22,2%, а до 2021-го року на ці рішення доводитиметься 45% софтверних витрат у світі. Самим швидкорослим сегментом аналітики називають IaaS, об'єм якого в 2018 році збільшиться майже на 40%. Український ринок хмарних сервісів росте дуже високими темпами. Згідно з даними дослідження, яке було проведено аналітиками «СиБ» [5][1)], у 2016 році об'єм цього сегменту досяг $13,6 млн., що майже на 35% більше, ніж результат 2015-го року (рис.1.4).

[1)] [5] Ринок хмарних послуг в Україні зріс на 70% за минулий рік – дослідження. Mind.ua. URL: https://mind.ua/news/20202735-rinok-hmarnih-poslug-v-ukrayini (дата звернення: 16.10.2019).

Рисунок 1.4 – Об'єм українського ринку хмарних сервісів в 2012-2018 роках

При цьому зростання по різних видах сервісів було нерівномірним. Якщо сегмент IaaS виріс на 30% (до $8, 7 млн.), то SaaS – на 72% ($4, 3 млн.), доля PaaS скоротилася, але такі рішення не роблять істотного впливу на місцевий ринок, оскільки займають сьогодні не більше 3,5%. Якщо розглядати сегмент в контексті розподілу різних типів сервісів, то за результатами року домінує IaaS, на який припало 64% ринку. SaaS, усупереч глобальним тенденціям, поки що відстає, хоча і росте випереджаючими темпами. Як прогнозують учасники вітчизняного хмарного ринку, до 2020 року долі IaaS і SaaS в Україні повинні порівнятися [5][1)].

Компанія NetApp оприлюднила у березні 2017 року результати свого першого галузевого опитування про перехід на хмарні технології в Європі. Аналіз відповідей семисот п'ятдесяти респондентів з числа IT-директорів і

[1)] [5] Ринок хмарних послуг в Україні зріс на 70% за минулий рік – дослідження. Mind.ua. URL: https://mind.ua/news/20202735-rinok-hmarnih-poslug-v-ukrayini (дата звернення: 16.10.2019).

менеджерів у Франції, Німеччині і Великобританії показав, що найбільшою популярністю під час переходу компаній на хмарні технології користується гібридна хмара – більше половини усіх учасників опитування в кожній країні відзначили, що зробили вибір на користь комбінації приватної і публічної хмари (69 % в Німеччині; 61 % у Франції; 58 % у Великобританії). Крім того, більше половини респондентів виділили безпеку в якості головної причини переходу на хмару, що вказує на те, що довіра до провайдерів таких послуг продовжує зміцнюватися. Основними напрямами використання хмари в усіх включених в дослідження країнах були зберігання і резервне копіювання [4][1)].

Проведений огляд підтверджує, що на даний момент ринок IT в цілому скорочується, а використання хмарних технологій росте. Очевидно, що це відбувається за рахунок того, що зараз більше вкладають засобів в хмарні технології. Головною тенденцією розвитку хмарних технологій стає гібридна хмара і мультисервісні системи.

Останнім часом велику популярність отримали хмарні технології, які мають величезний потенціал для істотного підвищення ефективності підприємства. Вже зараз багато провідних аналітичних компаній відмічають інтенсивне зростання світового ринку хмарних рішень і послуг.

1.3 Дослідження та аналіз застосування аналогічних мобільних і мережевих додатків

Для здійснення проектування та розробки хмарного сервісу «Фітнес-студія» необхідно здійснити дослідження та аналіз аналогічних мобільних і мережевих додатків, представлених у мережі Інтернет, що надасть можливість визначити ключові вимоги та функції хмарного сервісу.

[1)] [4] Разработка ПО (мировой рынок). URL: http://www.tadviser.ru/index.php/Статья:Разработка_ПО_%28мировой_рынок%29 (дата звернення: 10.10.2019).

Першим було розглянуто мобільний додаток «Nike Training Club» [6][1)], який реалізовано для платформи ПК, iOC (для iPhone і iPad) і Android. Додаток забезпечує виконання тренувань в виді покрокових переходів до фото- і відео- інструкціям домашніх вправ (рис 1.5).

Рисунок 1.5 – Програма тренувань у мобільному додатку Nike TClub

Спочатку відбувається реєстрація, обирається мета тренувань(рис 1.6).

Рисунок 1.6 – Огляд програми тренувань в мобільному додатку Nike TClub

[1)] [6] Приложение Nike+ Training Club для iOS и Android. Nike.com (RU). URL: https://www.nike.com/ru/ru_ru/c/nike-plus/training-app (дата звернення: 16.10.2019).

Наступний крок: встановлюєте рівень: початковий – 4-5 тренувань на місяць, середній – 2-3 тренування на тиждень, просунутий – 3-5 тренувань на тиждень. Сервіс видає вам програму занять, які залежно від рівня та мети тривають 15, 30 або 45 хвилин. Кожне тренування важить приблизно 12-15 Мб: слід враховувати це, якщо у вас обмежений Інтернет-трафік.

Сервіс фіксує кількість запущених вами тренувань, витрачений час, калорії і т. п. За активні заняття користувач отримуєте нагороди (красиві значки) і бонуси (тренування з відомим тренерами). Результатами можна ділитися на сайті Nike, в соціальних мережах і блогах (рис 1.7).

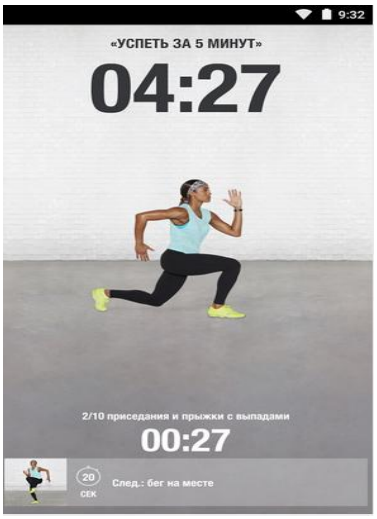


Рисунок 1.7 – Хід тренувань в мобільному додатку Nike TClub

До переваг цього додатка можна виділити те, що додаток є відмінним вибором для активної молоді без особливих проблем зі здоров'ям. Малорухливим, з поганою координацією, з обмеженнями за станом здоров’я, деякі вправи будуть спочатку не по силах, а деякі можуть навіть виявитися травмонебезпечними .

До недоліків цього додатка можна навести те, що програма включає ті вправи, які не можна робити зі сколіозом, болями в колінах і попереку, варикозом, остеохондрозом. Відсутній медичний довідник, спочатку інтерфейс програми здається складним: буває непросто знайти ту чи іншу функцію. Але

при здійсненні аналізу та огляду у цьому додатку корисними виявляються наступні функціональні можливості: створення програм тренувань різних рівнів складності; графічний опис вправ, що може буди використано для реалізації у хмарному сервісі «Фітнес-студія»

Наступним було досліджено та проаналізовано мобільний фітнес-додаток «Pump Up» [7][1] – фітнес-ком'юніті, тренер і бібл)іотека вправ (рис. 1.8).

Рисунок 1.8 – Енциклопедія вправ в мобільному додатку PumpUp

Додаток скачується безкоштовно, до того ж багата частина функцій і вправ віддається безоплатно. При реєстрації в додатку за допомогою електронної пошти або аккаунта на Facebook Pump Up автоматично створює профіль нового користувача. Таким чином ви вже через хвилину стаєте учасником соцмережі, в якій об'єднані спортсмени-любителі. Тут ви можете спостерігати за успіхами людей з усього світу, а також ділитися своїми знімками і досягненнями. Соцмережа Pump Up служить відмінним джерелом мотивації, але головна функція програми – допомогти користувачу досягти певних фіт-

[1)] [7] PumpUp – Fitness Community 6.0.1 Загрузить APK для Android – Aptoide. URL: https://pumpup.ru.aptoide.com/ (дата звернення: 16.10.2019).

нес-цілей. Мобільний додаток «PumpUp» реалізовано для платформи iOC і Android. У цьому додатку більше 500 вправ з покроковими інструкціями і фото: з них формуються силові або інтервальні тренування. Обираєте свою мету (схуднути, стати сильнішими і т.д.), рівень підготовки, улюблене обладнання (гантелі, штанга, або без обладнання взагалі) і отримуєте готову програму тренінгу. Сервіс фіксує час заняття, витрачені калорії, виконані мети. Результатами можливо ділитися в блогах на сайті PumpUp. До переваг розглянутого додатку можливо виділити те, що сервіс задовольняє потреби тих, хто хоче при виборі будь-якої мети користувач будете опрацьовувати окремо різні групи м'язів, як у тренажерному залі. Також сервіс набирає користувачу вправи в залежності від фітнес-цілі і устаткування, якому віддаєте перевагу. У додатку є голосові інструкції. До недоліків цього додатку можливо навести те, що додаток немає російської мови. Також додатком можна користуватися тільки з мобільного пристрою. Але при розгляді функціональних можливостей цього додатку можливо відзначити цікавими наступні функції: створення тренувань за певним списком вправ та покроковий опис вправ.

Наступним був розлянений сервіс (рис. 1.9) «Jillian Michaels» [8][1)].

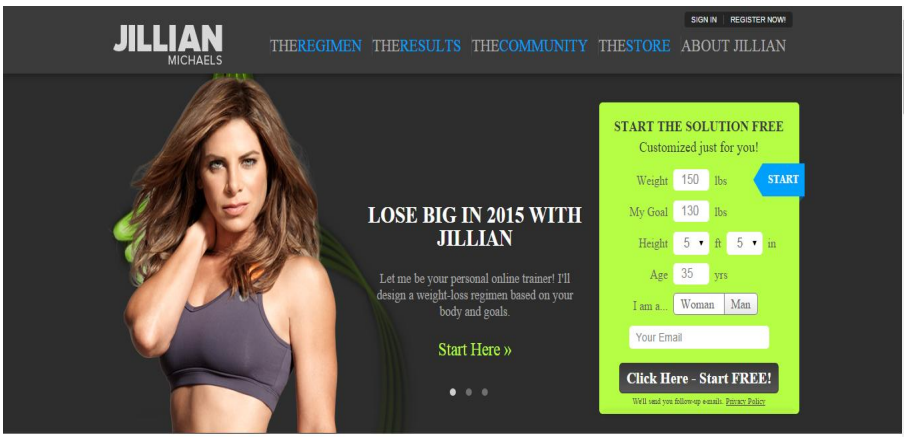


Рисунок 1.9 – Реєстрація в Інтернет-сервісі Jillian Michaels

[1)] [8] App Store: Jillian Michaels Fitness App. URL: https://apps.apple.com/ru/app/jillian-michaels-fitness-app/id1190148494 (дата звернення: 18.10.2019).

Jillian Michaels – це безкоштовний додаток для занять спортом, який включає в себе понад 170 вправ з докладними відео-інструкціями. Користувач зможете скласти персональний план вправ і тренувань, виходячи з типу статури і поставлених цілей, а також змагатися з іншими учасниками програми. Jillian Michaels є Інтернет-сервісом і доступний для різних платформ з підключеним Інтернет з'єднанням.

У сервісі надаються відео-уроки фітнесу з демонстрацією вправ і поясненнями від модного американського тренера Джилліан Майклс. Деякі програми тривають всього 20-30 хвилин. Завантажити їх можна, зареєструвавшись на сайті, обравши програму: зазвичай новачки починають з «плоский живіт за 6 тижнів» або «стрункa фігура за 30 днів». До переваг даного сервісу можна виділити те, що тренування не займають багато часу.

Сервіс дає безкоштовно зразок дієти і план тренувань на місяць. А також присутній багато статей про принципи здорового харчування. До недоліків сервісу можливо навести те, що немає додатку для мобільного пристрою для любої ОС. При огляді і дослідженні цього додатку цікавими для подальшої реалізації у хмарному сервісі «Фітнес-студія» можуть бути загальна ідея та архітектура Інтернет-сервісу з серверної базою даних.

Результатом огляду, дослідженню та аналізу існуючих аналогічних мобільних і мережевих фітнес-додатків у мережі Інтернет стало виявлення функціональних можливостей подібних систем, переваг і недоліків додатків, що дозволило здійснити постановку завдання та виявити набір функцій для реалізації хмарного сервісу «Фітнес-студія».

1.4 Постановка завдання

Дослідження та аналіз розглянутих мобільних і мережевих додатків, їх функціональних особливостей дозволяє визначити завдання для реалізації хмарного сервісу «Фітнес-студія». В сервісі повинні бути реалізовані наступні функціональні можливості:

- зберігання енциклопедичної інформації про вправи. Зберігання всіх даних Енциклопедії в віддаленому хмарному сервісі. Перехід по посиланнях в Інтернет з докладним описом вправи і на пошук у відео сервіс YouTube з відео-виконанням вправ;
- формування інформації про проведення та виконанні силових тренувань, бігових тренувань і вело- тренувань;
- формування і зберігання інформації про споживані продукти харчування та їх калорійності;
- формування статистичної інформації про проведені тренування;
- можливість вступу користувача до складу різних тренувальних груп, виконання групових програм тренувань, коментування програм тренувань;
- створення сервісу обробки та зберігання даних в хмарі;
- створення віртуальних фітнес груп тренером;
- додавання нових тренувань в список створених груп;
- перегляд статистичної інформації по веденої групі;
- коментарі та рекомендації щодо проведеного тренування в ведених групах.

Реалізація хмарного сервісу «Фітнес-студія» керується терміналогією, що використовується при проведенні силових тренувань. Основні терміни наведені нижче.

Силове тренування – в бодібілдингу, важкої атлетики, пауерліфтингу та інших видах спорту має багато різних методик, застосовуваних для досягнення конкретних результатів, цілей: адаптації тіла атлета до короткочасної максимальному навантаженні (пауерліфтинг), до тривалої середньо інтенсивному навантаженні – витривалості – і великому обсягу м'язів (бодібілдинг).

Вправа – різні способи виконання вправи на задану групу м'язів порізному впливають на м'язи, змушуючи їх безперервно розвиватися.

Повторення – повторення виконання вправи одного повного циклу – піднесення і опускання ваги – з контролем траєкторії руху.

Сет (підхід) – складається з декількох повторень, які виконуються безперервно одне за іншим без перерви між ними.

Інтенсивність відноситься до кількості роботи, необхідної для досягнення мети і пропорційності м'язової маси і підйому ваг.

2 ВИБІР АРХІТЕКТУРИ ТА ВИДУ ХМАРНОГО СЕРВІСУ ДЛЯ РЕАЛІЗАЦІЇ

2.1 Визначення структури хмарного сервісу «Фітнес-студія»

Розглянувши різні види хмарних сервісів у першому розділі магістерської роботи доцільно використовувати для реалізації системи «Фітнес-студія» інфраструктуру як сервіс, т. к необхідно використовувати цей сервіс, як платформу віртуалізації. Користувач насправді не буде здійснювати управління базовою інфраструктурою при взаємодії з IaaS, проте він управляє сховищами даних, операційною системою, розгортаними застосуваннями і вибраними мережевими компонентами.

Інформаційна система хмарної фітнес-студії розділена на дві частини:

– перша – це мобільний клієнт під операційну систему Android;

– друга частина – хмарний сервіс.

Структура ІС «Фітнес-студія» приведена на рис. 2.1.

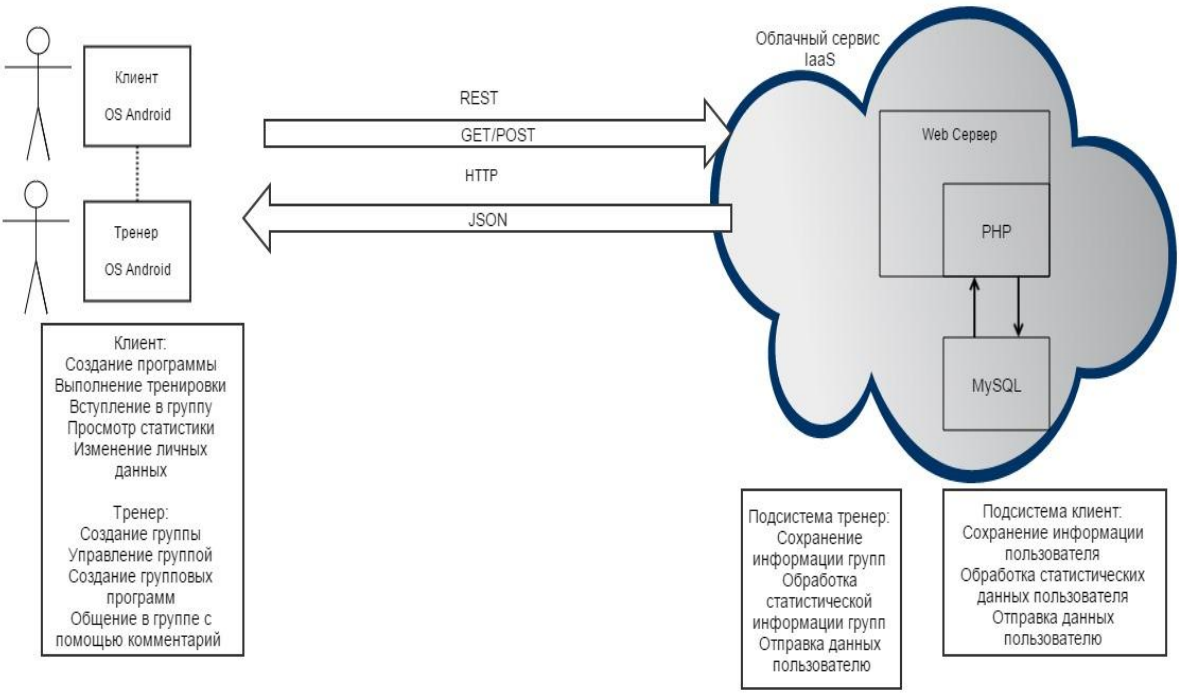


Рисунок 2.1 – Структура хмарного сервісу «Фітнес-студія»

Система є багатокористувацькою і має два типи користувачів: клієнт і тренер. Хмарний сервіс містить базу даних MySQL, веб-сервер, а також PHP модулі. Зв'язок між клієнтом під ОС Android і хмарним сервісом відбувається за допомогою комунікаційного протоколу REST. Мобільний клієнт формує HTTP запит і відправляє його на сервер за допомогою методів GET і POST. Сервер обробляє запит за допомогою PHP скриптів і залежно від завдання виконує обчислювальні дії з даними з бази даних, зберігає дані в базу даних і відправляє дані мобільному клієнту. Так само на мобільному клієнті виконуються локальні функції роботи з даними.

2.2 Дослідження механізмів реалізації в системі клієнт-серверного спілкування

Для здійснення подальшого проектування хмарного сервісу «Фітнес-студія» необхідно провести дослідження і обрати найбільш відповідний спосіб реалізації клієнт-серверного спілкування. Розглянемо можливі варіанти.

Перший підхід – на основі сокетів (Socket API). Часто використовується в додатках, де важлива швидкість доставки повідомлення, важливий порядок доставки повідомлень і необхідно тримати стабільне з'єднання з сервером. Такий спосіб реалізується в месенджерах та іграх (рис. 2.2).

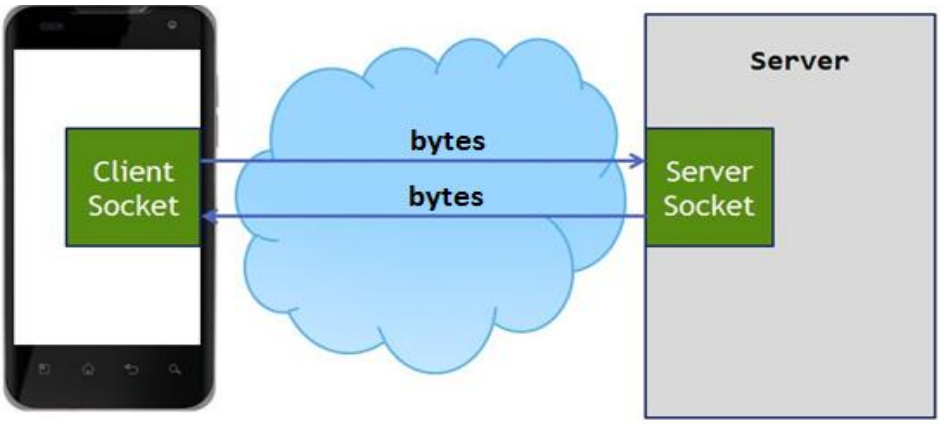


Рисунок 2.2 – Сокет з'єднання

Другий підхід – це часті опитування (polling): клієнт посилає запит на сервер і повідомляє йому: «дай мені свіжі дані». Сервер відповідає на запит клієнта і віддає все, що у нього накопичилося до цього моменту. Механізм з'єднання при цьому підході наведено на рис. 2.3.

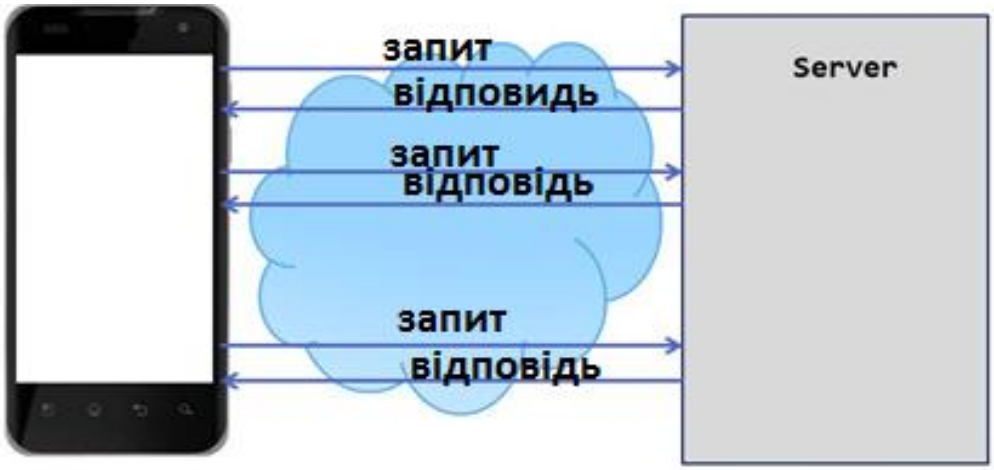


Рисунок 3.3 – З'єднання типу часті опитування (polling)

Недоліком такого підходу є те, що клієнт не знає, чи з'явилися свіжі дані на сервері. По мережі зайвий раз вирушають запити, в першу чергу через часті установки з'єднань з сервером.

Третій підхід – довгі опитування (long polling) – полягає в тому, що клієнт посилає запит на сервер. Сервер дивиться, чи є свіжі дані для клієнта, якщо їх немає, то він тримає з'єднання з клієнтом до тих пір, поки ці дані не з'являться. Як тільки дані з'явилися, він «пушит» їх назад клієнтові. Клієнт, отримавши дані від сервера, тут же посилає наступний запит і т.д.

Реалізація цього підходу досить складна на мобільному клієнті в першу чергу через нестабільність мобільного з'єднання. Але при цьому підході трафіку витрачається менше, ніж при звичайному polling'е, тому скорочується кількість установок з'єднань з сервером.

Механізм з'єднання при цьому підході наведено на рис. 2.4.

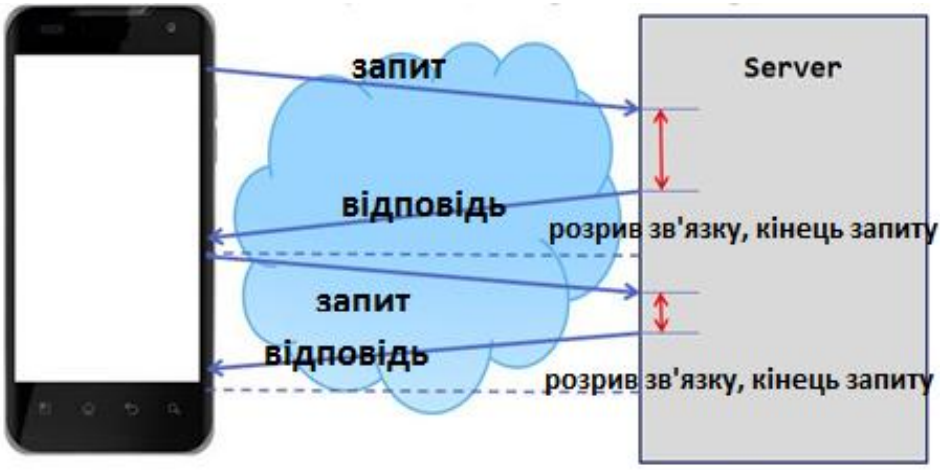


Рисунок 2.4 – З'єднання типу довгі опитування (long polling)

Механізм long polling, або пуш-повідомлень (push notifications), реалізований в самій платформі Android. І, напевно, для більшості завдань буде краще використовувати його, а не реалізовувати самим. Додаток підписується у сервісі Google Cloud Messaging (GCM) на отримання пуш-повідомлень (рис. 2.5).

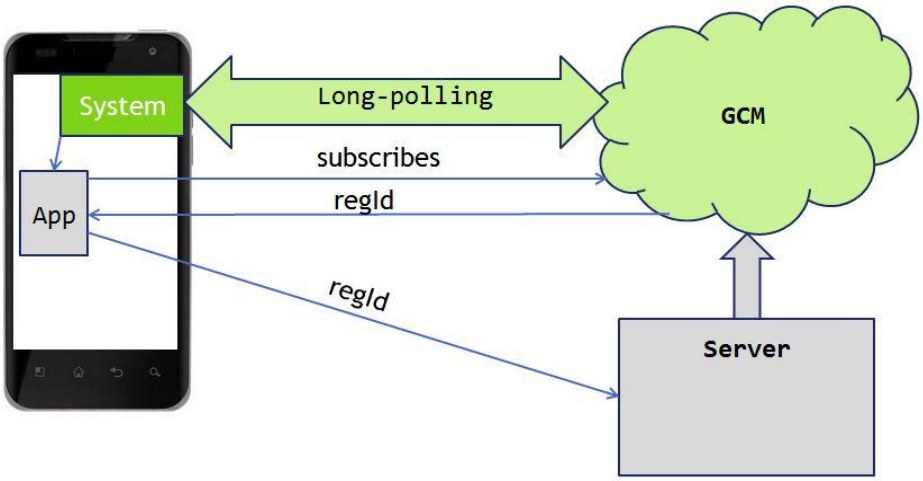


Рисунок 2.5 – Механізм long polling, або пуш-повідомлень (push notifications)

Тим самим розривається зв'язок безпосередньо між сервером і клієнтом за рахунок того, що сервер працює з сервісом GCM і відправляє свіжі дані

завжди на цей сервіс, а він вже в свою чергу реалізує всю логіку доставки цих даних до вашого застосування. Перевагами цього підходу є те, що усувається необхідність частих установок з'єднання з сервером за рахунок того, що ви точно знаєте, що дані з'явилися, і про це вас оповіщає сервіс GCM.

В результаті дослідження з цих чотирьох механізмів (підходів) найбільш популярними при розробці бізнес-додатків є пуш-повідомлення та часті опитування. При реалізації цих підходів нам так чи інакше доведеться встановлювати з'єднання з сервером і передавати дані. Для реалізації хмарного сервісу «Фітнес-студія» краще використовувати довгі запити, тому що користувачу не потрібно завжди тримати зв'язок з сервером.

2.3 Визначення інструментів взаємодії з сервером і відправлення запитів

Далі розглядаються інструменти, які необхідні для роботи з HTTP/ HTTPS-протоколам – HttpUrlConnection та HttpClient. В арсеналі Android-розробника є два класи для роботи за цими протоколами: java.net.HttpURLConnection, org.apache.http.client.HttpClient. Обидві ці бібліотеки включені в Android SDK. HttpURLConnection містить тільки один клас. Це пояснюється тим, що батьківський клас URLConnection був спроектований для роботи не тільки по HTTP-протоколу, а ще за такими протоколами, як file, mailto, ftp і т.п. Схема HttpURLConnection наведена на рис. 2.6

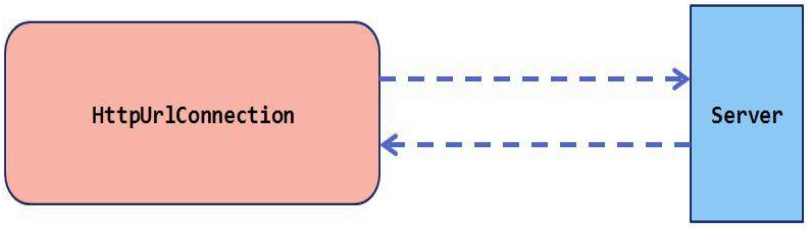


Рисунок 2.6 – Схема HttpURLConnection

HttpClient був спроектований, як більш об'єктно-орієнтований. У ньому є чіткий поділ абстракцій. У самому простому випадку розробник працює з п'ятьма різними інтерфейсами: HttpRequest, HttpResponse, HttpEntity і HttpContext, тому клієнт для Apache набагато великовагові HttpUrlConnection. Як правило, на всі додатки існує всього один екземпляр класу HttpClient, що обумовлено його ваговитістю. Використання окремого примірника на кожен запит буде марнотратним, але можливо зберігати екземпляр HTTP-клієнта в спадкоємця класу Application [9][1)]. Схема HttpClient наведена на рис. 2.7.

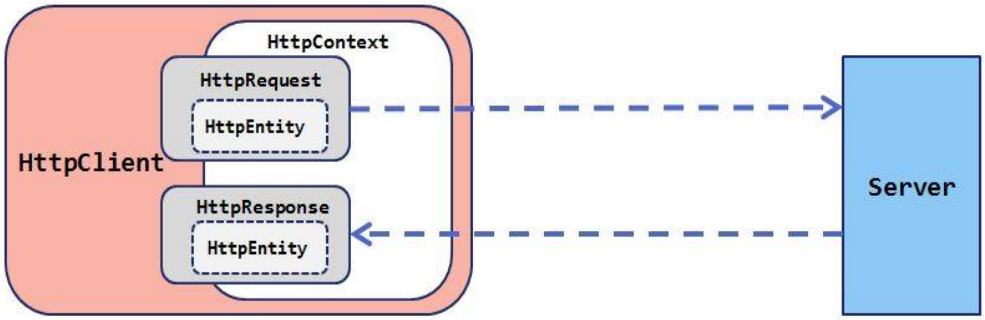


Рисунок 2.7 – Схема HttpClient

Схема взаємодії HttpClient с классом Application наведена на рис. 2.8.

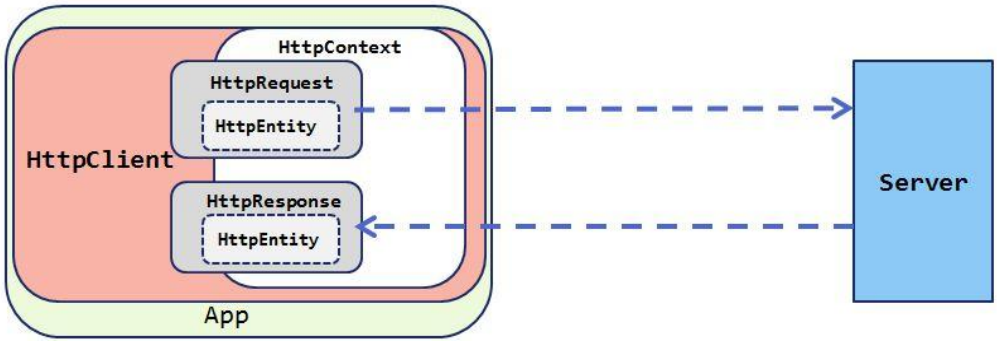


Рисунок 2.8 – Використання HttpClient с классом Application

[1)] [9] Автостопом по галактике «Cloud Computing» | Статьи | Компьютерное Обозрение. URL: https://ko.com.ua/avtostopom_po_galaktike_cloud_computing_102566 (дата звернення: 04.10.2019).

2.4 Вибір середовища та інструментів розробки сервісу

Для реалізації клієнтської частини хмарного сервісу «Фітнес-студія», як додатка для пристроїв під операційну систему Android, була обрана для використання середа Eclipse. Android – операційна система, створена компанією Google на основі ядра Linux. Популярність і відкритість платформи привертає до неї все більше число програмістів оскільки призначені для користувача програми для цієї системи реалізовані на мові Java, у неї сьогодні практично немає реальних конкурентів. Початкова установка і налаштування середовища розробки для Android передбачає використання наступних інструментів [10][1)]:

- Java Development Kit (версія 5 або вище);
- Android SDK;
- середовище розробки Eclipse IDE for Java Developers;
- розширення Android Development Tools (ADT) для IDE Eclipse.

Java Development Kit (JDK) – безкоштовно поширюваний компанією Oracle Corporation, комплект розробника додатків на мові Java, що включає в собі компілятор Java (javac), стандартні бібліотеки класів Java, приклади, документацію, різні утиліти і виконавчу систему Java (JRE). До складу JDK не входить інтегроване середовище розробки на Java. Android SDK включає в себе різноманітні інструменти, які допомагають розробляти мобільні додатки для платформи Android. Найбільш важливі з них – емулятор і плагін для Eclipse, однак до складу SDK входять різні інструменти для налагодження, упаковки та інсталяції ваших додатків на емулятор. Емулятор Android – це віртуальний мобільний пристрій, який запускається на звичайному комп'ютері. Емулятор використовується для проектування, налагодження і тестування програм в реальному середовищі виконання Android.

[1)] [10] З. Медникс. Программирование для Android: Программирование на Java для нового поколения мобильных устройств. Пер. с англ. СПб.: Питер, 2013. 560 с.

Плагін ADT – це потужне розширення для інтегрованого середовища Eclipse, що робить створення й налагодження додатків легше і швидше. Якщо ви використовуєте Eclipse, цей плагін дає неймовірний приріст швидкості розробки додатків для Android. Плагін ADT забезпечує [11][1)]:

- надання доступу до решти інструментам Android всередині Eclipse. Наприклад, ADT дозволяє отримати доступ до багатьох можливостей DDMS (Dalvik Debug Monitor Service) – зняття скріншотів, управління портами, настройка контрольних точок (breakpoints), перегляд інформації про потоки і процеси – безпосередньо з Eclipse;
- надання New Project Wizard, який допоможе швидко створити і налаштувати всі необхідні файли для додатка;
- автоматизацію і спрощення процесу побудови додатків;
- надання редактора коду Android, який допоможе написати правильний файл XML для Android-manifest і файлів ресурсів.

Інтегрований з Dalvik, стандартною віртуальною машиною платформи Android, інструмент ADT дозволяє управляти процесами на емуляторі або пристрої, а також допомагає в налагодженні додатків. Розробник може використовувати цей сервіс для завершення процесів, вибору певного процесу для налагодження, генерування трасувань даних, перегляду інформації про потоки, робить скріншоти емулятора або пристроя і багато іншого. Інструмент ADT дозволяє встановити файли з розширенням ".apk" на емулятор або пристрій з командного рядка, дозволяє створювати файли .apk, що містять бінарні коди і ресурси Android-додаткі. Він також дозволяє генерувати код для міжпроцесорного інтерфейсу .sqlite3 та отримати доступ до файлів даних SQLite, створених з використанням додатків для Android. Платформа Eclipse розроблена і побудована для виконання наступних вимог:

- підтримка конструювання різноманітних інструментів для розробки додатків;

[1)] [11] Обзор платформы Eclipse – как её использовать. Статьи о программном обеспечении. URL: http://hightech.in.ua/content/art-eclipse-platform (дата звернення 28.10.2019)..

- підтримка необмеженого ряду постачальників інструментарію, включаючи незалежних постачальників програмного забезпечення (ISV);
- підтримка інструментів в маніпулюванні довільним типом вмісту (тобто, HTML, Java, C, JSP, EJB, XML і GIF);
- забезпечення «безшовної» інтеграції інструментів с різними типами вмісту і різними постачальниками інструментарію;
- підтримка середовищ розробки додатків як з графічним інтерфейсом користувача (GUI), так і без GUI;
- виконання на широкому спектрі операційних систем, включаючи Windows і Linux;
- використання популярності мови програмування Java для написання інструментарію.

Важлива роль платформи Eclipse полягає в забезпеченні постачальників інструментами, механізмами і правилами, використання яких і дотримання яких призведе до безшовної інтеграції інструментів. Ці механізми представляються через чітко визначені інтерфейси, класи і методи в API. Платформа також забезпечує корисні вбудовані блоки і каркаси, які полегшують розробку нових інструментів. Підключення (plug-in) – найменша одиниця функціональності платформи Eclipse, яка може бути розроблена і поставлена окремо [11][1)].

Серверна частини хмарного сервісу «Фітнес-студія» містить модуль, до складу якого входять база даних, веб-сервер, PHP-модулі.

Для зберігання бази даних серверної частини використовується СУБД MySQL. СУБД MySQL є клієнт-серверної системою, що включає багатопоточний SQL-сервер, що підтримує різні платформи, кілька клієнтських програм і бібліотек, інструменти адміністрування і широкий діапазон програмних інтерфейсів додатків (API-інтерфейсів).

[1)] [11] Обзор платформы Eclipse – как её использовать. Статьи о программном обеспечении. URL: http://hightech.in.ua/content/art-eclipse-platform (дата звернення 28.10.2019).

Зв'язок між клієнтом під ОС Android і хмарним сервісом відбувається за допомогою протоколу REST. Абревіатура REST, яка розшифровується як Representational State Transfer, відповідає компактному, витонченому протоколу web-сервісів. Він став відповіддю на великовагові протоколи web-сервісів, такі як SOAP і XML-RPC, які спираються на готовий формат повідомлень і методи їх передачі між серверами і клієнтами. REST не накладає таких обмежень; ви можете використовувати будь-який формат повідомлень, який подобається (будь то JSON, XML, HTML, впорядковані дані або навіть прямий текст) і працювати зі стандартними дієсловами HTTP GET, DELETE і POST. Правила REST для взаємодії клієнт – сервер можуть цілком визначатися вимогами додатків. Так, якщо визначити інтерфейс REST, призначений для клієнта JavaScript, можна зробити так, щоб дані повертались в форматі JSON, а якщо він призначений для клієнта PHP, то це можуть бути впорядковані дані або XML.

Кожен виклик системи REST – це простий запит HTTP з використанням одного з стандартних дієслів HTTP. Як правило, використовується дієслово, найбільш підходящий для виконуваної дії: GET – для отримання даних від сервера; POST – для передачі даних на сервер; DELETE – для видалення ресурсу на сервері.

3 ПРОЕКТУВАННЯ ХМАРНОГО СЕРВІСУ «ФІТНЕС-СТУДІЯ»

3.1 Проектування функцій тренера

Інформаційний хмарний сервіс «Фітнес-студія» є багатокористувацькою системою. Користувачі сервісу поділяються на три типи: простий користувач, адміністратор і тренер. Також у сервісі передбачено управління, яке здійснюється адміністратором, який матиме доступ до бази даних і управління групами.

Для того щоб приєднатися до системи тренеру, необхідно зареєструватися, обравши тип користувача – користувача-тренер. Нижче наведена діаграма функціональних можливостей тренера сервісу (рис 3.1).

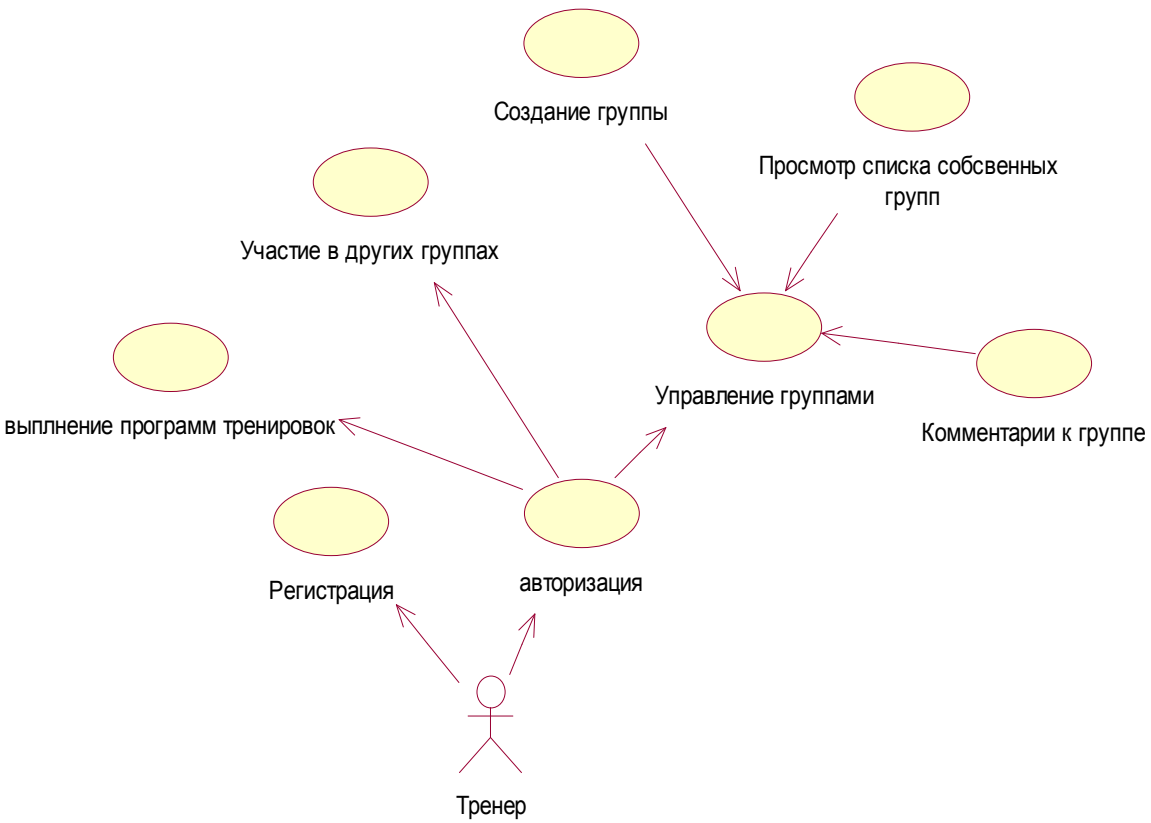


Рисунок 3.1 – Діаграма функціональних можливостей користувача-тренера

Тренеру надається можливість доступу до функцій користувача-клієнта, що дозволяє йому створювати власні програми тренувань і виконувати їх. Але головною особливістю користувача-тренера є реалізація функції створення груп. При створенні групи тренер дає їй назву і опис та рекомендації. Далі тренер створює в групі певну програму силових тренувань. Опис групи необхідно, для того щоб користувачі могли оцінити свої сили, а також щоб виконувати цю програму чи ні. Щоб до групи мали доступ користувачі, група проходить перевірку адміністратором.

Тренер може подивитися статистику групи. Це надає функції контролю над групою. Для спілкування в групі є замітки групи. Учасники групи можуть залишати свої коментарі, ставити питання, а також тренер може додавати свої коментарі до групи. Система також містить віртуального тренера. Віртуальний тренер дає коментарі по статистиці. Тобто якщо у вас погана інтенсивність, то система дає пораду, що краще зробити щоб її збільшити, а також інші види коментарів. Тренер має можливість створити кілька груп, а також брати участь у групах інших тренерів.

3.2 Проектування функцій клієнта

Хмарний сервіс «Фітнес-студія» забезпечує використання сервісу категорією користувачів, які бажають займатися тренуваннями і зацікавлені в отриманні очикуванного результату. Ця категорія користувачів визначена в системі, як користувач-клієнт. При запуску додатку викликається клас, в якому відбувається авторизація і вхід користувача-клієнта в додаток. Користувач вводить свої дані, логін користувача і пароль, після чого відбувається перевірка введених даних з даними в таблиці користувачів і після перевірки правильності перехід в клас головного меню. При натисканні кнопки реєстрація відбувається перехід в клас, де користувач вводить необхідні данні, та вказує до якої категорії користувачів він зараховується: тренером або клієнтом. При натисканні кнопки Реєстрація відбувається введення нових даних в

таблицю користувачів. Функціональні можливості користувача-клієнта сервісу «Фітнес-студія» наведені на рис. 3.2.

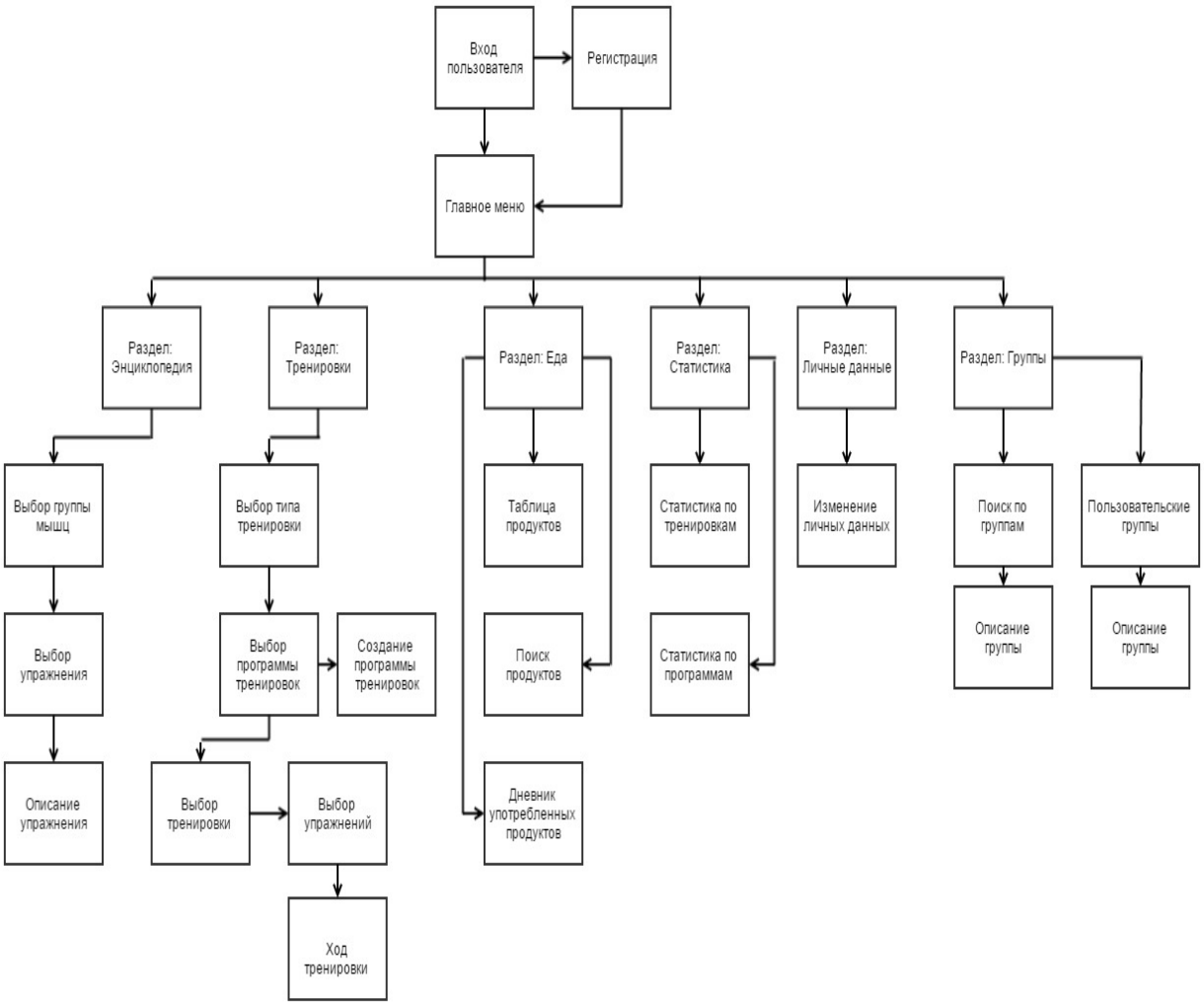


Рисунок 3.2 – Функціональний набір і навігація в підсистемі клієнта сервісу

У головному меню представлені View-елементи, в які виводитися інформація користувача про останні тренування користувача зі статистикою про закінчені тренування. Для переходу в основні розділи системи використовуються елементи Tab. При натисканні на елементи Tab відбувається перехід між розділами.

При переході клієнта в розділ груп, в класі, виконується пошук груп по типу груп. При виборі типу груп в View-елемент виводитися групи певного

типу. При виборі певної групи користувач переходить в новий клас, в якому виводитися інформація про цю групу і можливість вступити в неї. Так само в класі при натисканні на View-елемент «Користувальницькі групи», відбувається перехід в клас, в якому виводитися список груп, в які входить користувач. При натисканні на групу відбувається перехід в клас, в якому виводитися інформація про групу з можливістю залишити цю групу і залишати коментарі.

Створення програми тренувань – це окрема подія, в якій користувач дає назву програми, дату початку та закінчення тренування. В іншій події користувач вибирає створені особисті програми або групові програми. У цьому вікні відображається список програм. Натиснувши на певну програму запускається нове вікно з вибором тренування і кнопкою створення нового тренування. Користувач вибирає вже створене тренування або створює нове, задаючи їй дату і її тип. Вибравши тренування, користувач вибирає вправи, які будуть входити в ці тренування. На даній сторінці розташований список дозволених вправ, натиснувши на вибрану вправу користувач додає її в список тренування. Якщо тренування входить до складу групової програми, то користувач просто вибирає готовє тренування. Натиснувши на кнопку «Почати» запускається нове вікно с початком тренування.

3.3 Проектування бази даних хмарного сервісу

Насамперед в проектуванні бази даних виділимо сутності, які будуть зберігатися в базі даних. Головною сутністю можна визначити програму тренувань, яку користувач буде виконувати, крім того обрана користувачем-клієнтом програма тренувань може бути груповою.

Кожна програма тренувань повинна мати свою назву, а також дату її початку, дату виконання і ідентифікатор групи, до якої вона належить. І так як сервіс є багатокористувальницький, у таблиці програм тренувань повинні

бути ідентифікатор користувача, якщо ця програма тренувань визначена як індивідуальна.

Тренування містить у собі id програми тренувань. Кожне тренування відноситься до певної програми тренувань. Тренування повинне мати дату виконання, тип, а також тривалість виконання. У кожне тренування входять вправи. Вправи повинні мати кількість повторень, і з якою вагою ці повторення виконувалися.

Так само до головної сутності можна віднести користувача. Він характеризується ім'ям, віком, вагою та іншими біометричними даними і полем в якому вказується тип цього користувача, або клієнт має обмежену кількість прав доступу, або це користувач-тренер, який має розширений список прав і можливостей у системі.

Сутність групи містить номер, назву, опис і автора. До сутності групи відноситься сутність учасники групи. Дана сутність містить id користувачів і id групи.

Сутність енциклопедія містить у собі інформацію про вправи для тренувань. У ній знаходиться опис вправи, графічне представлення і посилання в Інтернет з повним описом і відео з демонстрацією правильного виконання.

Сутність їжа зберігає інформацію про продукт. До цієї інформації відноситься енергетична цінність продукту.

Далі, коли визначені основні сутності, необхідно провести проектування реляційної бази даних хмарного сервісу «Фітнес-студія».

У кожної сущності атрибут id є первинним ключем.

В сущності Program зберігається інформація про створені користувачем тренувальні програми або інформацію про групові програми. При створенні програми тренувань, в таблицю записується основна інформація: назва – атрибут name, дата початку програми – атрибут startDate, дата закінчення програми – атрибут finishDate, ідентифікатор користувача – атрибут id_user. Якщо ця програма не є індивідуальною, то цьому атрибуту присвоюється значення 0 і ідентифікатор групи – атрибут id_group, якщо програма є груповою,

якщо програма не є груповий цього атрибуту присвоюється значення 0. Також у таблиці зберігається унікальний номер, атрибут id, який створюється автоматично. Атрибут id є числовим полем. Атрибути: name, startDate finishDate є текстовими полями. Атрибут id_ user та атрибут id_group є числовим полем.

В сутності Trainings зберігається інформація про створені користувачем тренування. При створенні користувачем тренування в сутність записується основна інформація: назва – атрибут date, тип тренування – атрибут type і унікальний номер тренування – атрибут id_program. Також, по суті, зберігається унікальний номер – атрибут _id, який створюється автоматично, атрибут full_time містить час тренування, у атрибуті checked зазначається закінчення тренувань. Атрибут id є числовим полем. Атрибути: date, type, full_time є текстовими полями. Атрибут id_program и checked є числовим полем.

В сутності Approach зберігається інформація про виконані користувачем підходи. При записі підходу в таблицю зберігається інформація: кількість повторень – поле number_of_repetitions, вага – поле weight і унікальний номер вправи – поле id_exercises. Також у таблиці зберігається унікальний номер – поле _id, який створюється автоматично. Поле id, number_of_repetitions є числовими полями. Поле weight є текстовим полем. Поле id_exercises є числовим полем.

В сутності Groups зберігається інформація про створені групи тренувань. При створенні нової групи в таблицю записується інформація: назва групи – поле name, автор групи – поле autor і унікальний номер групи – поле id. Поле id є числовим полем. Поле name є текстовим полем. Поле autor є текстовим полем.

В сутності Users_group зберігается інформація про користувачів, які входять до складу груп. При додаванні користувача в групу в таблицю записується інформація: унікальний номер групи – поле id_group, унікальний

номер користувача – поле id_user і унікальний номер – поле _id. Поле id, id_group є числовим полем.

В сутності Exercises зберігається інформація про виконані користувачем вправи. При додаванні вправи в тренування в таблицю записується інформація: назва вправи – поле Name_EXer, м'язова група – поле muscle_group, номер тренування – поле id_traning і унікальний номер вправи – поле id. Поле id є числовим полем. Поле Name_EXer є текстовим полем. Поле muscle_group є текстовим полем. Поле id_traning є числовим полем.

В сутності Encyclopedia зберігається інформація про вправи та їх виконання. У таблиці міститься інформація про назву вправи – поле name, м'язові групі – поле type, описи вправи – поле descr, назви зображень – поле image, посилання в Інтернет для пошуку вправи – поле link і унікальний номер вправи – поле id. Поле id є числовим полем. Поля: name, type, descr, image, link є текстовими полями.

В сутності Food зберігається інформація про продукти харчування. З цієї таблиці виводиться інформація про продукти харчування, їх калорійності та харчової цінності. У таблиці міститься інформація про назву продукту – поле name, типі продукту – поле type, кількості калорій – поле calories, кількості білків – поле proteins, кількості жирів – поле fats, кількості вуглеводів – поле carbs і унікальний номер продукту – поле id. Поле id є числовим полем. Поля: name, type, calories, proteins, fats, carbs є текстовими полями.

В сутності Max_weight зберігається інформація про максимальні вагови показника вправ. Поля в таблиці відповідають за назву вправи – поле Exercises, вага – поле Weight і унікальний номер вправи – поле id. Поле id є числовим полем. Поле Exercises є текстовим полем. Поле Weight є числовим полем.

В сутності MuscleGroup міститься інформація про назви м'язових груп. В поле name зберігається назва м'язової групи, в поле id записується

унікальний номер м'язової групи. Поле id є числовим полем. Поле name є текстовим полем.

В сутності Users зберігається інформація про користувачів. У таблиці міститься інформація про ім'я користувача – поле name, вік користувача – поле age, вага користувача – поле weight, відсоток жиру – поле percentFat, обсяг біцепсу – поле volumeBiceps, обсяг грудей – поле breastVolume, об'єм стегон – поле hips, розмір талії – поле waist, обсяг квадрицепса – поле volumeQuadriceps, тип користувача – поле pravo і унікальний номер користувача – поле id. Поля: id, age, pravo є числовими полями. Поля: name, percentFat, volumeBiceps, breastVolume, hips, waist, volumeQuadriceps є текстовими полями. Поле weight є числовим полем типу real.

В сутності Zametki зберігається інформація для нотаток користувача під час тренувань. У таблиці міститься інформація про номер тренування – поле id_training, текст замітки – поле koment_text і унікальний номер замітки – поле id. Поле id є числовим полем. Поле koment_text є текстовим полем. Поле id_training є числовим полем.

В сутності Eating зберігається інформація про вжиті користувачем продукти харчування. У таблиці міститься інформація про дату – поле Day, назву продукту – поле Name, кількість продукту – поле Number, код користувача – поле id_user і унікальний номер – поле id. Поле id є числовим полем. Поле Day, Name є текстовим полем. Поле Number и id_user є числовим полем.

Сутність Program використовує зв'язок один до багатьох з сутністю Trainings по полях id і id_program.

Сутність Trainings використовує зв'язок один до багатьох з сутністю Exercises по полях id і id_training.

Сутність Exercises використовує зв'язок один до багатьох з сутністю Approach по полях id і id_exercises.

База даних хмарного сервісу «Фітнес-студія» приведена до третьої нормальної форми, а діаграма «Сутність-Зв'язок» наведена на рис. 3.3.

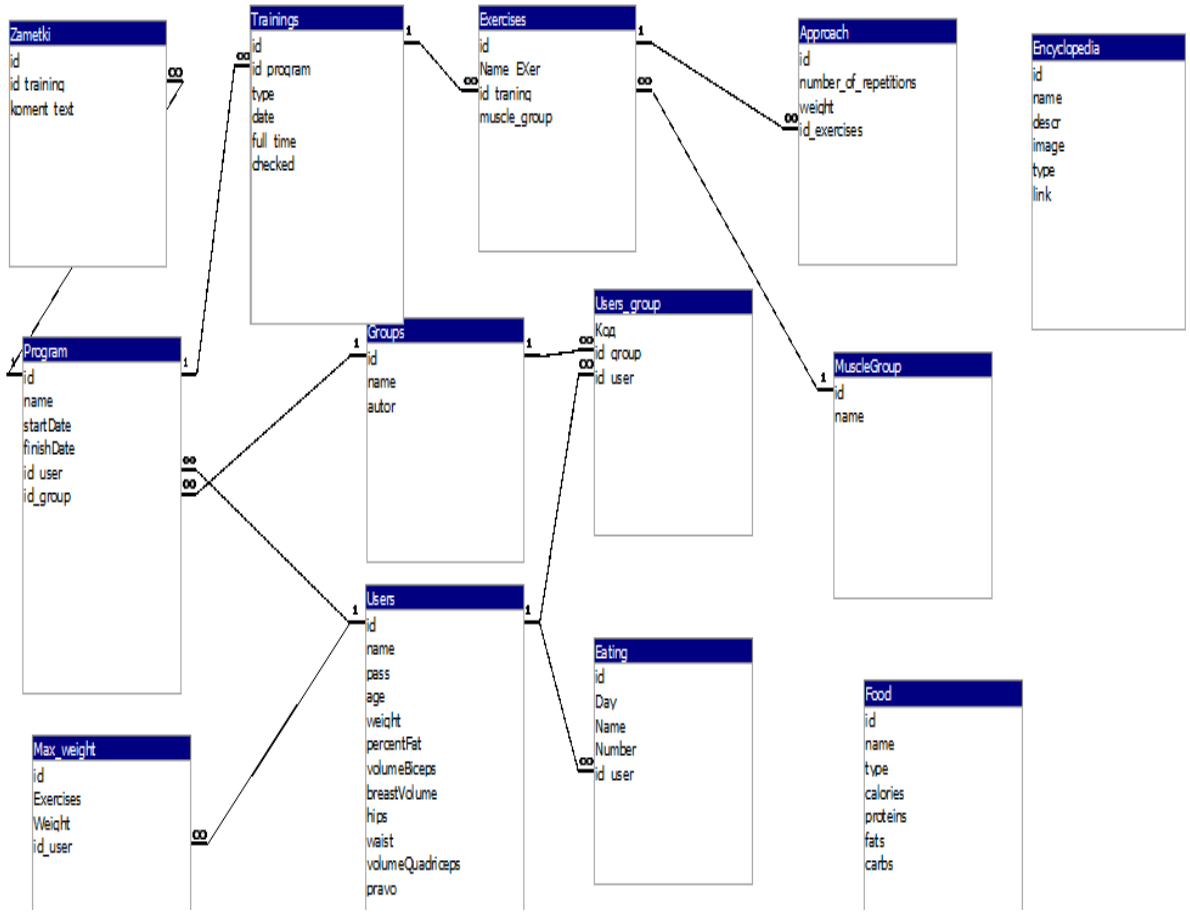


Рисунок 3.3 –Діаграма «Сутність- Зв'язок» бази даних хмарного сервісу

Сутність Exercises використовує зв'язок багато до одного з сутністю MuscleGroup по полях muscle_group і id.

Сутність Program використовує зв'язок багато до одного з сутністю Groups по полях id_group і id.

Сутність Program використовує зв'язок багато до одного з сутністю Users по полях id_user і id.

Сутність Program використовує зв'язок один до багатьох з сутністю Zametki по полях id і id_program.

Сутність Groups використовує зв'язок багато до одного з сутністю Users з використанням проміжної таблици Users_group.

Сутність Groups використовує зв'язок один до багатьох з сутністю Users_group по полях id і id_group.

Сутність Users використовує зв'язок один до багатьох з сутністю Users_group по полях id і id_user.

Сутність Users використовує зв'язок один до багатьох з сутністю Eating по полях id і id_user.

Сутність Users використовує зв'язок один до багатьох з сутністю Max_weight по полях id і id_user.

4 РОЗРОБКА ХМАРНОГО СЕРВІСУ «ФІТНЕС-СТУДІЯ»

4.1 Реалізація бази даних хмарного сервісу

Для створення та зберігання бази даних використовується СУБД MySQL. СУБД MySQL є клієнт-серверної системою, що включає багатопоточний SQL-сервер, що підтримує різні платформи, кілька клієнтських програм і бібліотек, інструменти адміністрування і широкий діапазон програмних інтерфейсів додатків (API-інтерфейсів). Сервер MySQL існує також і у формі вбудовуваної багатопотокової бібліотеки, яку можна пов'язувати з додатками, що розроблюються, щоб отримати більш компактні, швидкі і легкокеровані продукти. Для створення таблиць використовується веб-додаток PHPMyAdmin. PHPMyAdmin – веб-додаток з відкритим кодом, написаний на мові PHP і представляє собою веб-інтерфейс для адміністрування СУБД MySQL. Додаток користується популярністю у веб-розробників, оскільки дозволяє управляти СУБД MySQL без безпосереднього через введення SQL-команд, надаючи дружній інтерфейс [12][1)].

У базі даних хмарного сервісу «Фітнес-студія» зберігаються наступні таблиці: Groups, Users_group, Program, Trainings, Approach, Exercises, Encyclopedia, Food, Max_weight, MuscleGroup, Users, Zametki.

Для створення таблиці Groups використовувався наступний запит:

```
CREATE TABLE Groups (_id integer PRIMARY KEY AUTOINCREMENT,
name text,
autor text)
```

У таблиці Groups містяться поля: name, autor, _id. Поле _id має тип integer, є первинним ключем та має властивість autoincrement. Поле name є текстовим полем типу text. Поле autor є текстовим полем типу text.

Для створення таблиці Users_group використовувався наступний запит:

[1)] [12] Люк Веллинг, Лора Томсон. Разработка веб-приложений с помощью PHP и MySQL. М.: Вильямс, 2010. 848 с.

```
    CREATE TABLE Users_group (_id integer PRIMARY KEY
AUTOINCREMENT,
    id_group integer,
    id_user integer)
```

У таблиці Users_group містяться поля: id_group, id_user, _id. Поле _id має тип integer, є первинним ключем та має властивість autoincrement. Поле id_group є числовим полем типу integer. Поле id_user є числовим полем типу integer.Для створення таблиці Program використовувавсянаступний запит:

```
    CREATE TABLE Program (_id integer PRIMARY KEY
AUTOINCREMENT,
    name text,
    startDate text,
    finishDate text,
    id_user integer,
    id_group integer)
```

У таблиці Program містяться поля: name, startDate, finishDate, _id. Поле _id має тип integer, є первинним ключем і має властивість autoincrement. Поле name є текстовим полем типу text. Поле startDate є текстовим полем типу text. Поле finishDate є текстовим полем типу text. Поле id_group є числовим полем типу integer. Поле id_user є числовим полем типу integer.

Для створення таблиці Trainings використовувався запит:

```
    CREATE TABLE Trainings (_id integer PRIMARY KEY
AUTOINCREMENT,
    id_program integer,
    type text,
    date text,
    full_time text,
    checked integer)
```

У таблиці Trainings містяться поля: date, type, id_program, _id, full_time, checked. Поле _id має тип integer, є первинним ключем і має властивість autoincrement. Поле date є текстовим полем типу text. Поле type є текстовим полем типу text. Поле full_time є текстовим полем типу text. Поле id_program є числовим полем типу integer. Поле checked є числовим полем типу integer. Для створення таблиці Approach використовувався запит:

```
    CREATE TABLE Approach (id integer PRIMARY KEY
AUTOINCREMENT,
    number_of_repetitions integer,
    weight real,
    id_exercises integer)
```

У таблиці Approach містяться поля: number_of_repetitions, weight, id_exercises, _id. Поле _id має тип integer, є первинним ключем і має властивість autoincrement. Поле number_of_repetitions є числовим полем типу integer. Поле weight є числовим полем типу real. Поле id_exercises є числовим полем типу integer. Для створення таблиці Exercises використовувався запит:

```
    CREATE TABLE Exercises (_id integer PRIMARY KEY
AUTOINCREMENT,
    Name_EXer text,
    id_traning integer,
    muscle_group text)
```

У таблиці Exercises містяться поля: Name_EXer, muscle_group, id_traning, _id. Поле _id має тип integer, є первинним ключем і має властивість autoincrement. Поле Name_EXer є текстовим полем типу text. Поле muscle_group є текстовим полем. Поле id_traning є числовим полем типу integer. Для створення таблиці Encyclopedia використовувався запит:

```
    CREATE TABLE Encyclopedia (_id integer PRIMARY KEY
AUTOINCREMENT,
    name text,
    descr text,
    image text,
    type text,
    link text)
```

У таблиці Encyclopedia містяться поля: name, type, descr, image, link, _id. Поле _id має тип integer, є первинним ключем і має властивість autoincrement. Поле name є текстовим полем типу text. Поле type є текстовим полем типу text. Поле descr є текстовим полем типу text. Поле image є текстовим полем типу text. Поле link є текстовим полем типу text. Для створення таблиці Food використовувався запит:

```
CREATE TABLE Food (_id integer PRIMARY KEY AUTOINCREMENT,
name text,
type text,
calories real,
proteins real,
fats real,
carbs real)
```

У таблиці Food містяться поля: name, type, calories, proteins, fats, carbs, _id. Поле _id має тип integer, є первинним ключем і має властивість autoincrement. Поле name є текстовим полем типу text. Поле type є текстовим полем типу text. Поле calories є числовим полем типу real. Поле proteins є числовим полем типу real. Поле fats є числовим полем типу real. Поле carbs є числовим полем типу real. Для створення таблиці Max_weight використовувався запит:

```
CREATE TABLE Max_weight (_id integer PRIMARY KEY
AUTOINCREMENT,
Exercises text,
Weight integer)
```

Для створення таблиці MuscleGroup використовувався запит:

```
CREATE TABLE MuscleGroup (_id integer PRIMARY KEY
AUTOINCREMENT,
name text)
```

У таблиці MuscleGroup містяться поля: name, _id. Поле _id має тип integer, є первинним ключем і має властивість autoincrement. Поле name є текстовим полем типу text. Для створення таблиці Users зроблений запит:

```
CREATE TABLE Users (_id integer PRIMARY KEY AUTOINCREMENT,
name text,
age integer,
weight real,
percentFat real,
volumeBiceps real,
breastVolume real,
hips real,
waist real,
volumeQuadriceps real,
pravo text)
```

У таблиці Users містяться поля: name, age, weight, percentFat, volumeBiceps, breastVolume, hips, waist, volumeQuadriceps, _id. Поле _id має тип integer, є первинним ключем і має властивість autoincrement. Поле name є текстовим полем типу text. Поле age є числовим полем типу integer. Поле weight є числовим полем типу real. Поле percentFat є числовим полем типу real. Поле volumeBiceps є числовим полем типу real. Поле breastVolume є числовим полем типу real. Поле hips є числовим полем типу real. Поле waist є числовим полем типу real. Поле waist є числовим полем типу volumeQuadriceps. Поле pravo є текстовим полем типу text. Для створення таблиці Zametki використовувався запит:

```
    CREATE TABLE Zametki (_id integer PRIMARY KEY
AUTOINCREMENT, id_program integer,
    koment_text text)
```

У таблиці Zametki містяться поля: id_training, koment_text, _id. Поле _id має тип integer, є первинним ключем і має властивість autoincrement. Поле koment_text є текстовим полем типу text. Поле id_training є числовим полем типу integer.

4.2 Розробка інтерфейсу користувачів сервісу «Фітнес-студія»

За графічне відображення у хмарному сервісі відповідають файли xml. Кожен файл відноситься до свого класу. В Android SDK реалізація елементів можлива за допомогою тексту або перетягування на форму елементу.

Інтерфейс Android, або оболонка, які описані в документації XML, можуть бути знайдені в папці res/layouts. Код шаблону, вже згенерований Eclipse, оголошений в res/layouts/main.xml.

Eclipse має у своєму розпорядженні власний інструментарій для проектування макета, який дозволяє створювати інтерфейс методом перетягування в межах екрану. Проте, часом легше написати інтерфейс в XML і використовувати графічний макет для попереднього перегляду результатів.

Одним з найбільш важливим елементом інтерфейсу в Android є контейнери Layout, такі як LinearLayout. Ці елементи невидимі для користувача, але виступають в якості контейнерів для інших елементів, таких як Buttons і TextViews. Структурно в xml знаходиться xml-опис всіх View layout-файлу. Назви xml-елементів – це класи View-елементів, xml-атрибути – це параметри View-елементів, тобто всі ці параметри, змінюються через вкладку Properties. У файлі login.xml описується інтерфейс входу в сервісний додаток сервісу «Фітнес-студія», вигляд якого наведено на рис. 4.1.

Рисунок 4.1 – Вхід користувача в систему

Розмітка LinearLayout вирівнює всі дочірні об'єкти в одному напрямку – вертикально чи горизонтально. Усі дочірні елементи поміщаються в стек один за іншим, так що вертикальний список уявлень буде мати тільки один дочірній елемент в рядку незалежно від того, наскільки широким він є. Горизонтальне розташування списку буде розміщувати елементи в один рядок з висотою, яка дорівнює висоті найвищого дочірнього елемента списку.

```
    <LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="match_parent"
```

```
        android:layout_height="match_parent"
        android:orientation="vertical" >
```

Далі описуються View-елементи Button. Xml-атрибути описують розташування кнопки, розмір тексту і унікальне ім'я.

```
    <Button
         android:id="@+id/encycl"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:layout_above="@+id/button2"
         android:layout_centerHorizontal="true"
         android:layout_marginBottom="26dp"
         android:text="Энциклопедия" />
```

Кожна кнопка, яка описується в цьому файлі, використовується для переходу з головного меню в певний розділ.

```
    <Button
         android:id="@+id/personal"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:layout_below="@+id/button4"
         android:layout_centerHorizontal="true"
         android:layout_marginTop="18dp"
         android:text="Личные данные" />
```

У файлі activity_main.xml описується інтерфейс головного меню:

```
    <RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:tools="http://schemas.android.com/tools"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:paddingBottom="@dimen/activity_vertical_margin"
        android:paddingLeft="@dimen/activity_horizontal_margin"
        an-
droid:paddingRight="@dimen/activity_horizontal_margin"
        android:paddingTop="@dimen/activity_vertical_margin"
        tools:context=".MainActivity" >
```

Вигляд інтерфейсу головного меню програми, наведено на рис. 4.2., меню відображає необхідні функції для клієнта-користувача і клієнта- тренера.

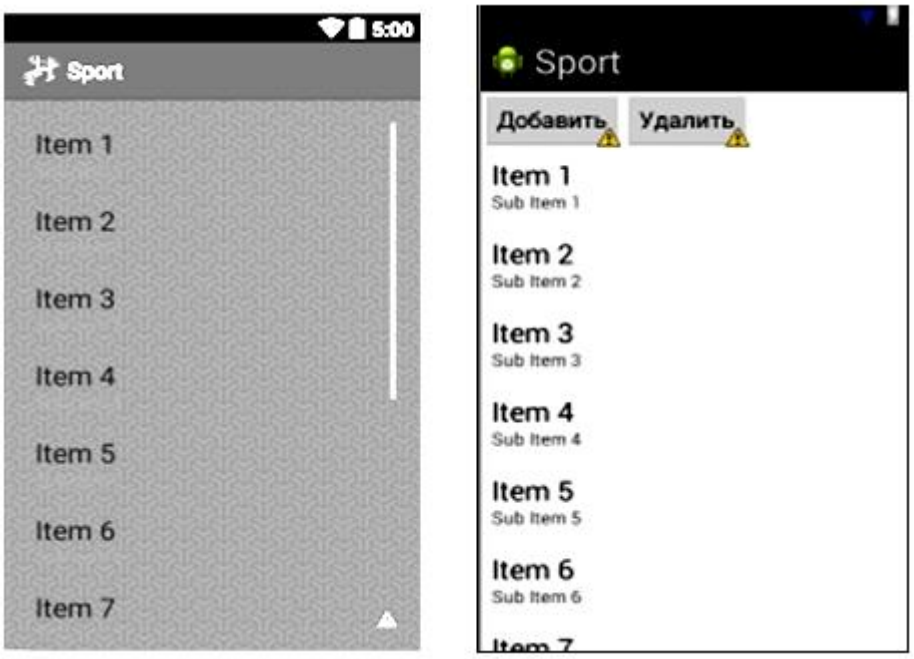


Рисунок 4.2 – Головне меню сервісу користувача і тренера

ReiativeLayout (відносна розмітка) дозволяє дочірнім уявленням визначати свою позицію щодо батьківського подання, або щодо сусідніх дочірніх елементів (ідентифікатора елемента). У RelativeLayout дочірні елементи розташовані так, що якщо перший елемент розташований по центру екрана, інші елементи, вирівняні щодо першого елемента, будуть вирівняні відносно центру екрана. При такому розташуванні, при оголошенні розмітки в XML-файлі, елемент, на який будуть посилатися для позиціонування інші об'єкти уявлення, повинен бути оголошений раніше, ніж інші елементи, які звертаються до нього за його ідентифікатором. Далі описується елемент ListView який являє собою список елементів. Xml-атрибути описують розмір списку і унікальне ім'я:

```
<ListView
    android:id="@+id/lvMain"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" >
</ListView>
```

У файлі encycl.xml описується інтерфейс користувача для здійснення вибору групи м'язів і вправи в розділі Енциклопедія. Далі описується елемент ListView який являє собою список елементів. Xml-атрибути описую розмір списку і унікальне ім'я.

```
    <ListView
        android:id="@+id/lvMain"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >
    </ListView>
</LinearLayout>
```

У файлі specification.xml описується користувальницький інтерфейс опису тренувальної вправи в розділі енциклопедія, яка наведена на рис. 4.3.

Рисунок 4.3 – Опис тренувальної вправи в додатку

У контейнери ScrollView і HorizontalScrollView можна розміщувати тільки один дочірній елемент (зазвичай LinearLayout), який у свою чергу може бути контейнером для інших елементів. Віджет ScrollView, незважаючи на свою назву, підтримує тільки вертикальну прокрутку, тому для створення вертикальної і горизонтальної прокрутки необхідно використовувати ScrollView в поєднанні з HorizontalScrollView. Зазвичай ScrollView використовують як кореневого елемента, aHorizontalScrollView в якості дочірнього.

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

ImageView є базовим елементом-контейнером для використання графіки. Можливо завантажувати зображення з різних джерел, наприклад, з ресурсів програми, контент-провайдерів.

```
   <ImageView
android:id="@+id/imageView1"
        android:layout_width="226dp"
        android:layout_height="285dp" />
```

Віджет TextView призначений для відображення тексту без можливості редагування його користувачем, що видно з його назви (Text – текст, view – перегляд). TextView – один з найбільш використовуваних віджетів. З його допомогою користувачеві зручніше орієнтуватися в програмі. TextView знаходиться в папці Form Widgets під чотирма варіантами: TextView, Large Text, Medium Text, Small Text.

```
        <TextView
            android:id="@+id/textView1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="TextView" />
```

При великій кількості інформації, яку потрібно помістити на екрані доводиться використовувати смуги прокрутки.

Кнопка – один з найпоширеніших елементів управління в програмуванні. Успадковується від Textview і є базовим класом для класу CompoundButton. Від класу CompoundButton в свою чергу успадковуються такі елементи какCheckBox, ToggleButton і RadioButton. В Android для кноп-

ки використовується клас android.widget.Button. На кнопці розташовується текст і на кнопку треба натиснути, щоб отримати результат.

```
    <Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Описание в интернете" />
    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Видео в You Tube" />
</LinearLayout>
</ScrollView>
```

Вибір тренування тренером відбувається у файлі training1, інтерфейс надання цього функціоналу наведено на рис 4.4.

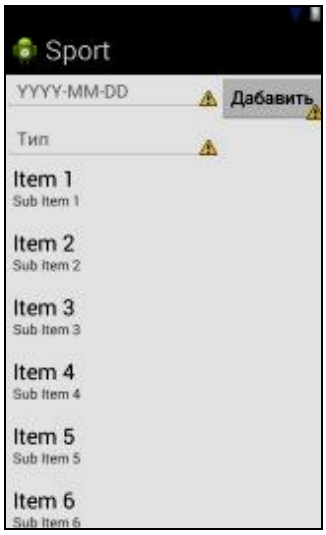


Рисунок 4.4 – Вибір тренування

В даному інтерфейсі розташовуються поля для введення назви нового тренування, тип тренування, кнопка додавання, а також список допустимих тренувань.

Елемент введення – це елемент editText, так елемент кнопки має назву button і список listView. Приклад коду наведено нижче:

```
<EditText
        android:id="@+id/editText2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/button1"
        android:ems="10"
         android:hint="Тип"/>
    <ListView
        android:id="@+id/lvMain"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_below="@+id/editText2" >
```

Вибравши тренування необхідно додати вправи, які будуть в нього входити. Це відбувається в файлі training2. Інтерфейс додавання вправ до обраного тренування наведено на рис 4.5.

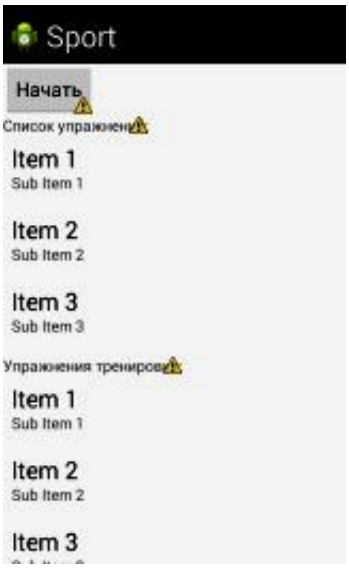


Рисунок 4.5 – Вибір вправ для тренування

На даній activity розташовано два списку: перший список допустимих вправ, другий список обраних вправ, присутня кнопка «Розпочати» тренування. Приклад реалізації цієї функції наведено нижче:

```
<Button
        android:id="@+id/button1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
    android:text="Начать" />
<TextView
    android:id="@+id/textView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Список упражнений:" />
<ListView
    android:id="@+id/listView1"
    android:layout_width="match_parent"
    android:layout_height="200dp" >
</ListView>
```

На найголовнішому activity відбувається виконання тренування, де наявні кнопки «Start», «Stop», «Reset», «Заметки», «Энциклопедия», «Записать» (рис. 4.6).

Рисунок 4.6 – Виконання тренування

В інтерфейсі тренування присутній загальний час проведення тренування, кнопки паузи / старту і скидання часу. Випадає список вправ, які входять до тренування. Кнопка заміток, що дозволяє записати текст для свого тренування. Поля для введення кількості повторень і вага, кнопку для їх запису, текстове поле записаних підходів. Для зручності передбачена кнопка

для переходу на інше actvity з енциклопедією по обраній вправі тренування. А також передбачено кнопку завершення тренування. Для створення списку використовується елемент spinner. Для відображення часу chronometr. Нижче наведено програмний код реалізації інтерфейсу.

```
<Chronometer
        android:id="@+id/chronometer1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:text="Chronometer"
        android:textSize="40sp"
        />
    <Button
        android:id="@+id/btnStart"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBaseline="@+id/btnStop"
        android:layout_alignBottom="@+id/btnStop"
        android:layout_alignParentLeft="true"
        android:text="Start" />
    <EditText
        android:id="@+id/editText2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignBaseline="@+id/editText1"
        android:layout_alignBottom="@+id/editText1"
        android:layout_alignParentRight="true"
        android:layout_toRightOf="@+id/btnStop"
        android:ems="10"
        android:inputType="numberDecimal"
        android:hint="Вес"/>
```

4.3 Реалізація функцій користувача сервісу «Фітнес-студія»

Хмарний сервіс складається з веб-сервера і MySQL сервера. Веб-сервер містить PHP-скрипти, які реалізують зв'язок між веб-сервером і сервером MySQL. Кожен PHP-скрипт відповідає за певну дію: створення, збереження, видалення, обчислення та інші подібні дії.

Для реалізації функціонального набору було створено 30 класів, призначення яких визначено функціональними вимогами клієнтів-користувачів та тренерів. При запуску програми викликається Activity клас Login, в якому відбувається вхід користувача в додаток, користувач вводить свої дані, логін

користувача і пароль. Дані відправляються на сервер за допомогою протоколу REST, де сервер перевіряє правильно ввів дані користувач чи ні. Сервер віддає id користувача, який необхідний для прив'язки тренувань, груп та інших даних. Після чого відбувається перехід в Activity клас MainActivity. Нижче наведено код відправки даних на сервер:

```
     List<NameValuePair> params = new Array-
List<NameValuePair>();
     params.add(new BasicNameValuePair("login", login.getText()
          .toString()));
     params.add(new BasicNameValuePair("pass",
pass.getText().toString()));
     // получаем JSON строк с URL
     JSONObject json = jParser.makeHttpRequest(url.getUrl()
     + url_enterUser, "GET", params);
     try {
     // получаем SUCCESS тег для проверки статуса ответа сервера
     success = Integer.parseInt(json.getString(TAG_SUCCESS));
     Log.d(LOG_TAG, "success  " + success);
     if (success == 1) {
     // данные введены верно
     products = json.getJSONArray(TAG_PRODUCTS);
     JSONObject c = products.getJSONObject(0);
     saveId(c.getString(TAG_ID));
     url.setId(c.getString(TAG_ID));
     saveName(c.getString(TAG_LOGIN));
     url.setName(c.getString(TAG_LOGIN));
     savePass(c.getString(TAG_PASS));
     url.setPass(c.getString(TAG_PASS));
```

Реалізація функції, що відповідає за здійснення перевірки сервера на правильність введення даних наведена нижче:

```
     if (isset($_GET['login']) && isset($_GET['pass'])) {
     $login = $_GET['login'];
     $pass = $_GET['pass'];
     require 'db_connect.php';
     $db = new DB_CONNECT();
     mysql_query("SET NAMES 'utf8'");
     $result = mysql_query("SELECT _id, name, pass from `Users`
where name='$login' and pass='$pass'");
     if ($result) {
     // successfully updated
     $response["User"] = array();
     while ($row = mysql_fetch_array($result)) {
     $User = array();
     $User["_id"] = $row["_id"];
     $User["name"] = $row["name"];
     $User["pass"] = $row["pass"];
     array_push($response["User"], $User);
```

Графічне відображення інтерфейсів для здійснення авторизації та реєстрації наведено на рис. 4.7.

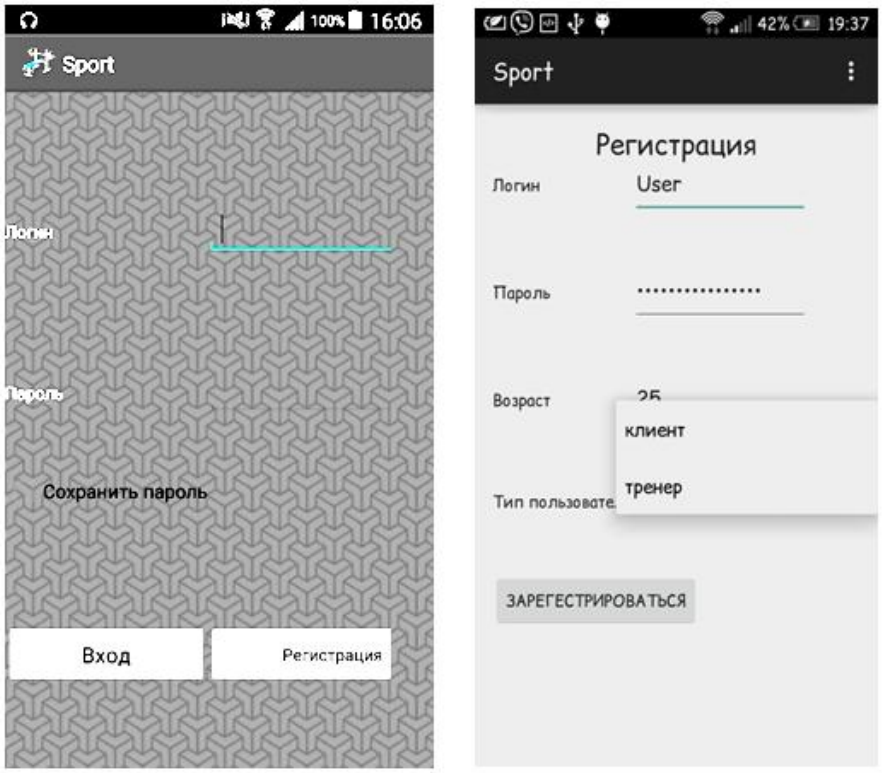


Рисунок 4.7 – Авторизація та реєстрація користувачів в системі

При натисканні кнопки реєстрація відбувається перехід в Activity клас Registration, де користувач вводить нові дані, вказує яким користувачем він хоче бути, тренером або клієнтом і при натисканні кнопки «Реєстрація» відбувається введення нових даних в таблицю користувачів.

TabActivity клас MainActivity відображає інтерфейс користувача де відображено головне меню програми. У головному меню представлені View-елементи Listview, в які виводитися інформація користувача про останні тренування зі статистикою про завершені тренування. Для переходу в основні розділи меню використовуються елементи Tab. При натисканні на елементи Tab відбувається перехід між розділами.

При переході користувача-клієнта в розділ груп тренування, в TabActivity Groups_menu класі виконується пошук груп по типу, при виборі типу груп в View-елемент Listview виводитися групи певного типу. При виборі певної групи, користувач переходить в новий Activity клас Group_specific, в якому виводитися інформація про цю групу і можливість вступити до неї (рис. 4.8).

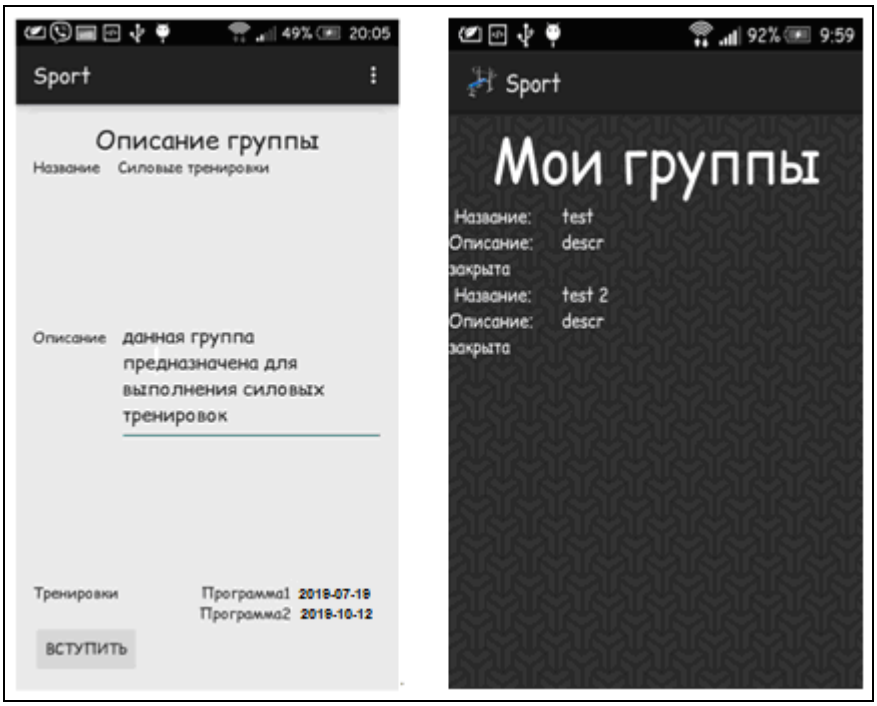


Рисунок 4.8 – Опис групи тренувань та список групп тренувань користувача

Так само в TabActivity класі при натисканні на View-елемент Button, відбувається перехід в Activity клас Users_groups, в якому виводитися список груп, в які входить користувач. При натисканні на групу відбувається перехід в Activity клас Group_specific, в якому виводитися інформація про групу з можливістю залишити цю групу і записати коментарі. Кожен користувач може спілкуватися с тренером групи та іншими учасниками групи за домогою коментарів групи. Користувач може спитати ради у тренера чи поділитися якоюсь новиною з іншими. Графічне представлення коментарів для групи представлено на рис. 4.9.

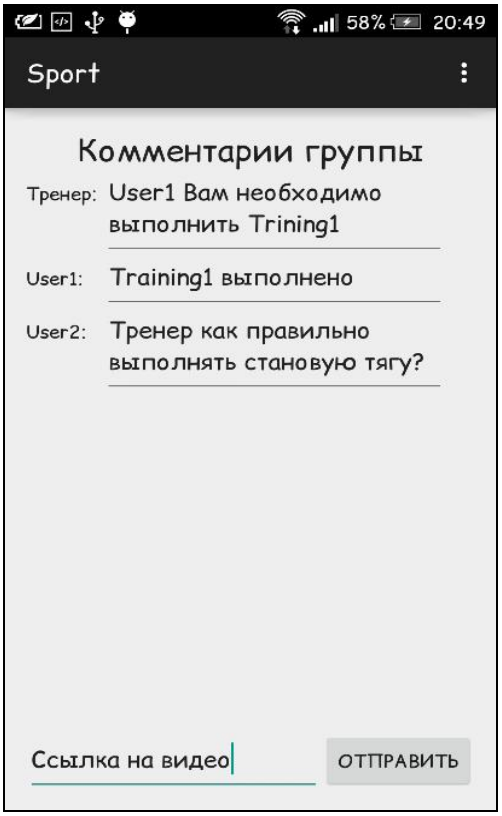


Рисунок 4.9 – Коментарі групи

Створення програми тренувань – це окрема подія, в якої користувач-тренер надає: назву програмі, дату початку та дату закінчення тренувань. В іншій події користувач-клієнт вибирає створені особисті програми або групові програми в Activity класі Program_menu.

У інтерфейсі відображається список програм. Натиснувши на певну програму запускається нове вікно в Activity клас Training1 з вибором тренування і кнопкою створення нового тренування. Користувач вибирає вже створене тренування або створює нове, задаючи дату і тип Обравши тренування, користувач вибирає вправи, які будуть входити в це тренування в Activity класі Training2 (рис. 4.10).

Якщо тренування входить до складу групової програми, то користувач просто вибирає готове тренування.

Рисунок 4.10 – Вибір чи створення тренування, вибір вправ

Натиснувши на кнопку «Начать» запускається нове вікно с початком тренування На сторінці розташований список допустимих вправ, натиснувши на вибрану вправу користувач додає її в список тренування (рис. 4.11).

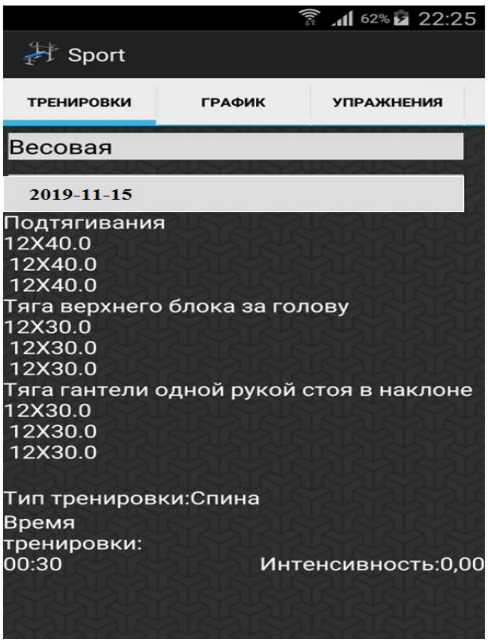


Рисунок 4.11 – Статистика тренування

Після того як тренування буде виконане, можливо проглянути виконані вправи с усіма підходами та вагою, а також інтенсивність, яку підраховує сервіс.

4.4 Реалізація класів взаємодії з сервером сервісу «Фітнес-студія»

Розглянемо реалізацію класів. У класі Login виполняється авторизація користувача. У класі создаються змінні екземпляри класу JSON для створення запитів до сервера:

```
JSONParser jParser = new JSONParser();
     private ProgressDialog pDialog;
     JSONArray products = null;
```

Далі створюються екземпляри view елементів екрану: кнопка, текстові поля. Об'являються змінні константи для формування запиту до сервера:

```
EditText pass, login;
     Button enter;
private static String url_enterUser = "enterUser.php";
     private static final String TAG_SUCCESS = "success";
     private static final String TAG_PRODUCTS = "User";
     private static final String TAG_ID = "_id";
     private static final String TAG_LOGIN = "name";
     private static final String TAG_PASS = "pass";
```

Далі оголошуються змінні для занесення значень в SharedPreferences. SharedPreferences – це клас, який дозволяє зберігати та видавати дані у вигляді пар «ключ-значення».

```
final String ID = "id";
     final String NAME = "name";
     final String PASS = "pass";
     final String FLAG = "flag";
     SharedPreferences sPref;
```

При натисканні на кнопку на екрані пристрою спрацьовує функція обробки натискання. При натисканні на кнопку відбувається зчитування значень з Shared Preferences.

Для запиту на сервер і зчитування інформації з бази даних создается окремий потік AsyncTask, в якому буде виконуватися формування запиту і зчитування відповіді. Інтерфейс для здійснення авторизації у системі наведено на рис. 4.12.

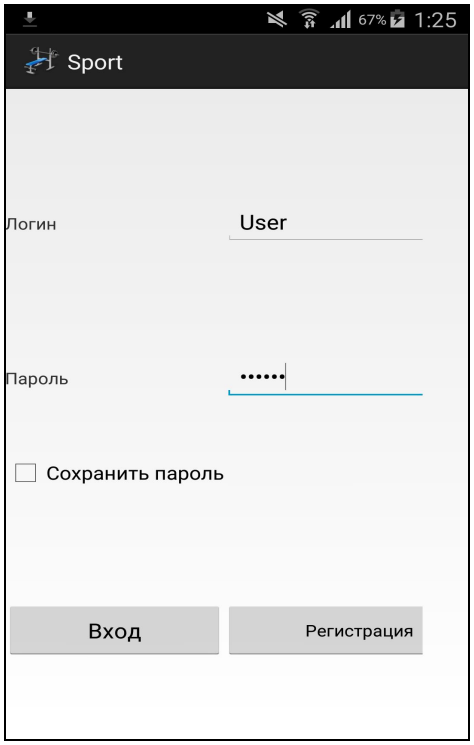


Рисунок 4.12 – Авторизація користувача в додатку

Клас AsyncTask пропонує простий і зручний механізм для переміщення трудомістких операцій у фоновий потік. AsyncTask створює, синхронізує потоки, а також управляє ними, що дозволяє створювати асинхронні завдання, що складаються з операцій, що виконуються у фоновому режимі, і оновлювати інтерфейс користувача по їх завершенні.

```
class enterUser extends AsyncTask<String, String, String> {
```

Під час виконання потоку на формі для користувача виводитися view елемент ProgressDialog для отображенія обробки запиту:

```
@Override
          protected void onPreExecute() {
               super.onPreExecute();
               pDialog = new ProgressDialog(Login.this);
               pDialog.setMessage("Enter, wait...");
               pDialog.setIndeterminate(false);
               pDialog.setCancelable(false);
               pDialog.show();
          }
```

Запит на сервер формується у функції doInBackground яка складає тіло потоку. На початку створюється масив значень в який записуються значення, які будуть передаватися через запит на сервер.

```
     protected String doInBackground(String... args) {
                    // Будет хранить параметры
List<NameValuePair> params = new ArrayList<NameValuePair>();
Log.d(LOG_TAG, "логин  " + login.getText().toString());
Log.d(LOG_TAG, "пароль  " + pass.getText().toString());
params.add(new BasicNameValuePair("login",
login.getText().toString()));
params.add(new BasicNameValuePair("pass",
pass.getText().toString()));
```

Далі створюється екземпляр класу JSONObject, за допомогою якого формуєте запит на сервер:

```
JSONObject json = jParser.makeHttpRequest(url.getUrl()
          + url_enterUser, "GET", params);
```

Далі відбувається відправлення запиту на сервер та отримання відповіді та запис отриманих значень з сервера в SharedPreferences:

```
try {
success = Integer.parseInt(json.getString(TAG_SUCCESS));
                    Log.d(LOG_TAG, "success  " + success);
                    if (success == 1) {
                    products = json.getJSONArray(TAG_PRODUCTS);
                    JSONObject c = products.getJSONObject(0);
                         saveId(c.getString(TAG_ID));
                         url.setId(c.getString(TAG_ID));
```

```
                    saveName(c.getString(TAG_LOGIN));
                    url.setName(c.getString(TAG_LOGIN));
                    savePass(c.getString(TAG_PASS));
                    url.setPass(c.getString(TAG_PASS));
                    } else {
                }
            } catch (JSONException e) {
                e.printStackTrace();
            }
```

Після отримання відповіді від сервера відбувається закриття потоку і перевірка відповіді від сервера, при проході перевірки відбувається перехід в головне меню програми.

```
    protected void onPostExecute(String file_url) {
              pDialog.dismiss();
              if(success==1){
    Intent intent = new Intent(Login.this, MainActivity.class);
                    startActivity(intent);
                    finish();
              }
```

Головне меню програми представляє собою TabActivity.

```
public class MainActivity extends TabActivity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.statist);
```

Вкладки, як у браузерів, можна реалізувати за допомогою елементів TabHost і TabWidget. Віджет TabHost, що дозволяє групувати пов'язані елементи управління в серію сторінок-вкладок.

```
TabHost tabHost = getTabHost();
        TabHost.TabSpec tabSpec;
        tabSpec = tabHost.newTabSpec("tag1");
        tabSpec.setIndicator("Главное меню");
        tabSpec.setContent(new Intent(this, MainMenu.class));
        tabHost.addTab(tabSpec);
```

TabHost є контейнером, який може містити елементи TabWidget. Тому невипадково, зображення TabHost виглядає як набір вкладок з TabWidget – Tab1, Tab2, Tab3 (рис. 4.13).

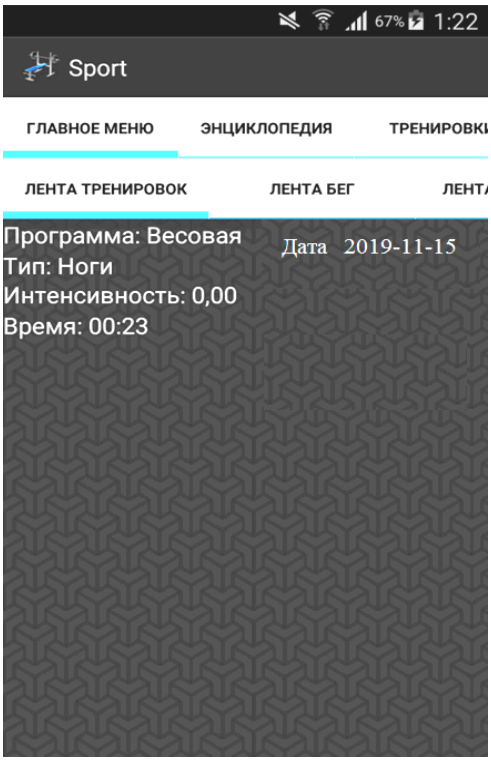


Рисунок 4.13 – Головне меню сервісу «Фітнес-студія»

TabHost показує ярлички кожної сторінки у своїй колекції. Коли користувач обирає вкладку, цей об'єкт посилає повідомлення в батьківський контейнер TabHost для перемикання на обрану вкладку.

```
tabSpec = tabHost.newTabSpec("tag2");
        tabSpec.setIndicator("Энциклопедия");
 tabSpec.setContent(new Intent(this, Encyclopedia_menu.class));
        tabHost.addTab(tabSpec);

        tabSpec = tabHost.newTabSpec("tag3");
        tabSpec.setIndicator("Тренировки");
   tabSpec.setContent(new Intent(this, All_Trainings.class));
        tabHost.addTab(tabSpec);
```

При переході в розділ Енциклопедія визивається клас Encyclopedia _menu, в якому здійснюється запит на серверну базу даних для виведення

груп м'язів і подальшим вибором вправ для певної групи. У класі створюються змінні екземпляри класу JSON для створення запитів до сервера.

```
JSONParser jParser = new JSONParser();
     private ProgressDialog pDialog;
     JSONArray products = null;
```

Оголошуються змінні константи для формування запиту до сервера.

```
Url url ;
     private static String url_all_programm = "readAllMus-
cleGroup.php";
     private static final String TAG_SUCCESS = "success";
     private static final String TAG_PRODUCTS = "Muscle-
Group";
     private static final String TAG_PID = "id";
     private static final String TAG_NAME = "name";
```

При запуску активності відбувається запуск окремого потоку, який формує запит до сервера.

```
public void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.encycl);
url = ((Url)getApplicationContext());
// присваиваем адаптер списку
new LoadAllProgramms().execute();
}
class LoadAllProgramms extends AsyncTask<String, String,
String> {
```

Під час виконання потоку на формі для користувача виводитися view елемент ProgressDialog для відображення обробки запиту:

```
@Override
protected void onPreExecute() {
super.onPreExecute();
pDialog = new ProgressDialog(Login.this);
pDialog.setMessage("Enter, wait...");
pDialog.setIndeterminate(false);
pDialog.setCancelable(false);
pDialog.show();
}
```

Запит на сервер формується у функції doInBackground, яка складає тіло потоку. На початку створюється масив значень, в який записуються значення які, будуть передаватися через запит на сервер.

```
     protected String doInBackground(String... args) {
     // Судет хранить параметры
     List<NameValuePair> params = new Array-
List<NameValuePair>();
     // получаем JSON строк с URL
     JSONObject json =
jParser.makeHttpRequest(url.getUrl()+url_all_programm, "GET",
          params);
```

Далі створюється екземпляр класу JSONObject, за допомогою якого формується запит на сервер. У змінну success записується результат запиту на сервер, якщо запит пройшов вдало, то виполняється запис повертаються сервером значень в об'єкт класу MuscleGroup.

```
     try {
     int success = Inte-
ger.parseInt(json.getString(TAG_SUCCESS));
     if (success == 1) {
     products = json.getJSONArray(TAG_PRODUCTS);
     for (int i = 0; i < products.length(); i++) {
     JSONObject c = products.getJSONObject(i);
     MuscleGroup muscleGroup = new MuscleGroup();
     muscleGroup.set_id(Integer.parseInt(c.getString(TAG_PID)));
     muscleGroup.setName(c.getString(TAG_NAME));
     list.add(muscleGroup);
```

Після завершення запиту до сервера відбувається закриття потоку. У view елемент ListView записуються значення отримані після виконання запиту і також виконується обробка натискання на елемент списку, при якому відкривається наступна активність.

```
     ListView lvMain = (ListView) findViewById(R.id.lvMain);
     MuscleGroupAdapter catAdapter = new MuscleGroupAdapter(
     Encyclopedia_menu.this, list);
     lvMain.setAdapter(catAdapter);
     lvMain.setOnItemClickListener(new OnItemClickListener() {
     public void onItemClick(AdapterView<?> parent,
     View view, int position, long id) {
     String nameGroup = ((TextView)
view.findViewById(R.id.textView1)).getText().toString();
```

```
Intent intent = new Intent(Encyclopedia_menu.this,
Type_muscles.class);
intent.putExtra("fname", nameGroup);
startActivity(intent);
```

Виконання тренування контролює клас Training_in_progres. Як тільки почали тренування починається відлік часу. Якщо користувачеві необхідно зупинити час чи скинути його, то для цього є аналогічні кнопки.

Для запису даних виконання вправи є два параметри: кількість повторів і вага. Ввівши ці дані користувач натискає кнопку записати, після чого ці дані відображаються нижче в списку. Цей список змінюється залежно від того, яка саме вправа обрана. Для вибору вправи відображається список з усіма вправами (рис. 4.14).

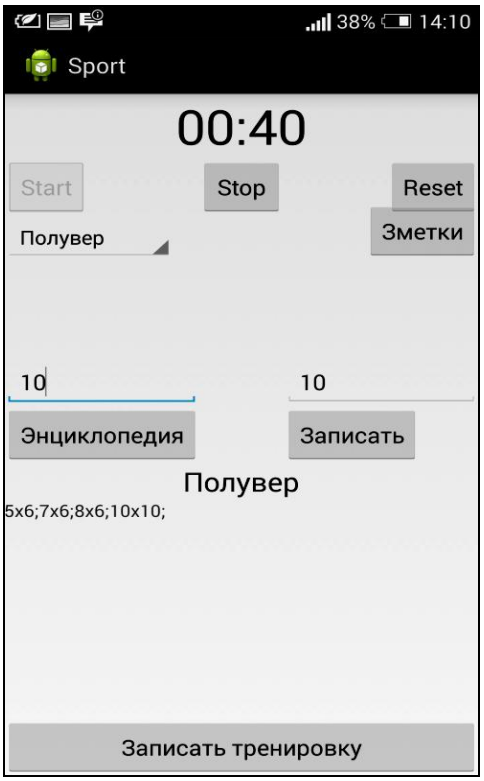


Рисунок 4.14 – Виконання тренування

Створюємо список з усіма вправами, зчитування з бази даних вправи до нашого тренування. Створюємо адаптер який формуватиме наш список і

передаємо дані які повернула база даних. Далі створюємо об'єкт spinner, які буде відображати наш список і цей список буде випадаючим:

```
     list1 = db.readAllExercises2(id_tr);
     ArrayAdapter<String> adapter = new Ar-
rayAdapter<String>(this, an-
droid.R.layout.simple_spinner_item,list1);
adapter.setDropDownViewResource(android.R.layout.simple_spinner_
dropdown_item);
            final Spinner spinner = (Spinner) findViewBy-
Id(R.id.spinner1);
             spinner.setAdapter(adapter);
             spinner.setPrompt("Упражнения");
             spinner.setSelection(0);
             String name = spinner.getSelectedItem().toString();
          text.setText(name);
```

Вибір вправи відбувається коли обрано в списку потрібна вправа. За ці виконання відповідає функція onItemSelected. Вона зчитує позицію вправи, її назву, а далі виводить підходи з бази даних, якщо вони є.

```
     final String names = spinner.getSelectedItem().toString();
                    text.setText(names);
                    int id_exes= db.getId_exes(names,id_tr);
                    List<Exercises> exer=null;
                    exer=db.readExercise(id_exes);
                    temp="";
                    String temp1="";

                    if(exer==null)temp="";
                    else
                    for(int i=0;i<exer.size();i++){
temp1=String.valueOf(exer.get(i).getNumber_of_repetitions())+"x"
+String.valueOf(exer.get(i).getWieght())+";";
                         temp=temp+temp1;
                    }
```

Підхід будь-якої вправи складається з кількості повторень і ваги. Ці дані записуються в базу даних і виводяться користувачеві у список. Запис в базу даних йде кількість повторень, вага, ідентифікатор вправи.

```
     String povtorText = povtor.getText().toString();
       String vesText = ves.getText().toString();
       String podhodText = Podhod.getText().toString();
       Podhod.setText(podhodText+""+povtorText+"x"+vesText+";");
     int id_exes=db.getId_exes(text.getText().toString(),id_tr);
```

```
       db.addExercises(id_exes, Integer.parseInt(povtorText),
Double.parseDouble(vesText));
```

Для зручності виконання вправи є кнопка, яка виводить нове вікно, де відображається сторінка з енциклопедії. У ній відображається фотографія з правильним виконанням вправи, короткий опис, посилання на опис в Інтернет, а також посилання на відео в YouTube.

```
     String nameExer = text.getText().toString();
     Intent intent = new Intent(Training_in_progres.this,
Gym_specification.class);
     intent.putExtra("fname",nameExer );
     startActivity(intent);
```

Так само користувач може залишити текстовий коментар до певного тренування. Для цього відповідна кнопка запускає діалогове вікно з текстовим полем, користувач пише свій коментар і він зберігається в базі даних. Даний коментар можна буде подивитися в статистиці тренувань. Приклад запуску і записи коментаря приведений нижче:

```
      public void onNothingSelected(AdapterView<?> arg0) {
                }
              });
             alert = new AlertDialog.Builder(this);
             alert.setTitle("Заголовок");
             alert.setMessage("Сообщение");
             final EditText input = new EditText(this);
             alert.setView(input);
             alert.setPositiveButton("OK", new DialogInter-
face.OnClickListener() {
             public void onClick(DialogInterface dialog, int
whichButton) {
               String value = input.getText().toString();
               //Podhod.setText(value);
               db.saveZametki(value, id_tr);
               }
             });
```

Завершення тренування проводиться натисненням відповідної кнопки. Після її натискання в базі даних зберігається час проведення тренування, всі вправи і ставиться прапор що це тренування виконано.

```
currentTime=chrono.getText().toString();
db.cheked_Traning(id_tr,currentTime);
Intent intent2 = new Intent(this, MainActivity.class);
startActivity(intent2);
```

4.5 Реалізація функцій тренера сервісу «Фітнес-студія»

Для створення групи тренеру необхідно записати назву і опис групи. Після чого натиснувши на кнопку створити дані відправляються зберігаються на сервері з позначкою недоступна. Для того щоб група була доступна користувачам, вона повинна пройти перевірку адміністрацією. Так само тренер повинен створити тренування для групи, наприклад:

```
     List<NameValuePair> params = new Array-
List<NameValuePair>();
     params.add(new BasicNameValuePair("id_user", url.getId()));
     params.add(new BasicNameValuePair("name",
text1.getText().toString()));
     params.add(new BasicNameValuePair("descr",
text2.getText().toString()));
     // получаем JSON строк с URL
     JSONObject json = jParser.makeHttpRequest(url.getUrl()
     + url_creategroup, "POST", params);
     try {
     // Получаем SUCCESS тег для проверки статуса ответа сервера
     int success = Inte-
ger.parseInt(json.getString(TAG_SUCCESS));
     if (success == 1) {
```

Графічне подання створення групи тренером представлено на рис. 4.15.

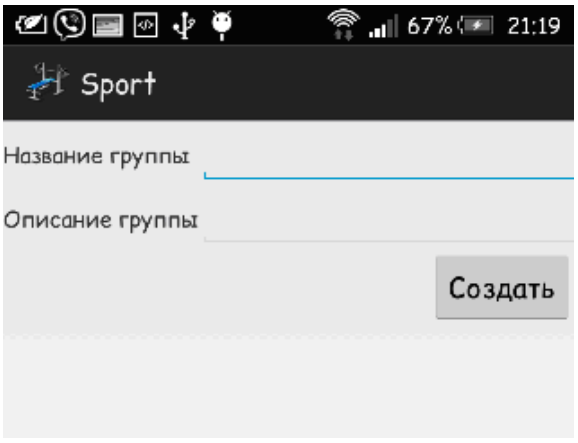


Рисунок 4.15 – Створення групи тренером

Якщо тренер бажає змінити назву чи опис своєї групи тренувань, він може обрати розділ редагування групи. Для того щоб запустити тренування, необхідно створити програму або вибрати вже існуючу. Нижче наведено графічне представлення інтерфейсу (рис. 4.16).

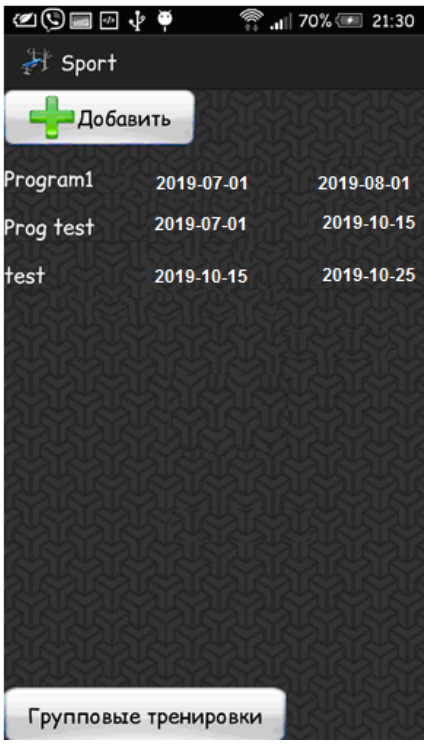


Рисунок 4.16 – Створення програми тренувань

За вибір програми відповідає клас Program_menu. У цьому класі відображаються всі програми. За додавання програми відповідає клас Program_add. У ньому додаємо ім'я програми, дату початку і закінчення її. Далі всі дані записуються в базу даних. Нижче наведено приклад коду.

```
String prog_name = edit1.getText().toString();
String start_date = edit2.getText().toString();
String end_date = edit3.getText().toString();
try {
 db.createDataBase();
    } catch (IOException e) {
// TODO Auto-generated catch block
 e.printStackTrace();
          }
db.openDataBase();
db.saveProgram(prog_name,start_date,end_date);
```

Обравши певну програму, можливо перейти до формування набору тренувань, які входять в цю програму (рис. 4.17).

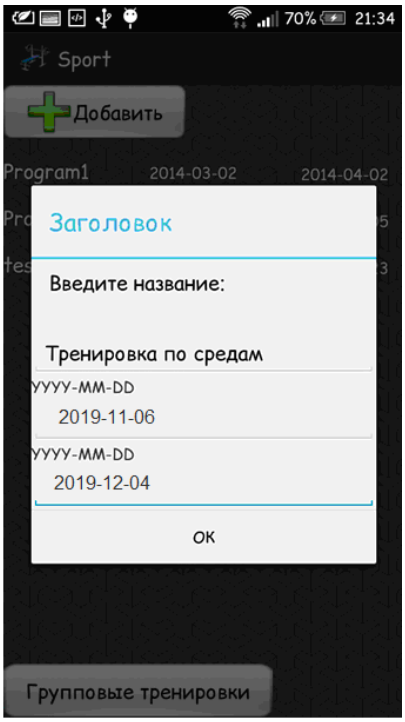


Рисунок 4.17 – Створення набору вправ для обраного тренування

Нижче наведено створення коду для класу Program_menu:

```
     db.openDataBase();
     list = db.readAllProgram();
     // находим список
     ListView lvMain = (ListView) findViewById(R.id.lvMain);
     // создаем адаптер
     ProgramAdapter catAdapter= new ProgramAdapter(this, list);
     lvMain.setAdapter(catAdapter);
     lvMain.setOnItemClickListener(new OnItemClickListener() {
     public void onItemClick(AdapterView<?> parent, View view,
     int position, long id) {
     String nameProgram = ((TextView)
view.findViewById(R.id.textView4)).getText().toString();
     Intent intent = new Intent(Program_menu.this, Train-
ings1.class);
     intent.putExtra("fname",nameProgram );
     Log.d(LOG_TAG,"fgh:"+String.valueOf( nameProgram));
     startActivity(intent);
             }});
```

За виведення тренувань, які входять у певну програму, відповідає клас Trainings1. За ідентифікатором , функція тренування readAllTrainings зчитує всі тренування, які входять в програму. Список тренувань вноситься в адаптер, який відповідає за їх висновок. Нижче наведений код:

```
db.openDataBase();
list = db.readAllTrainings(id_prog);
// находим список
ListView lvMain = (ListView) findViewById(R.id.lvMain);
// создаем адаптер
TrainingAdapte catAdapter= new TrainingAdapte(this, list);
lvMain.setAdapter(catAdapter);
```

Кожне тренування має своє ім'я і свій тип, а також завершено воно чи ні. Якщо користувач хоче нове тренування, він може його додати, ввівши назву і тип. Якщо тренування виконано, то його вибрати не можна, а якщо ні, то переходимо до його налаштування. За це відповідає функція onItemClick.

```
public void onItemClick(AdapterView<?> parent, View view,
int position, long id) {
String nameTR = ((TextView)
view.findViewById(R.id.textView2)).getText().toString();
String check = ((TextView)
view.findViewById(R.id.textView4)).getText().toString();
if(check.equals("Выполнено"))return;
Intent intent = new Intent(Trainings1.this, Train-
ings2.class);
intent.putExtra("fname",nameTR );
startActivity(intent);
  }});
```

До налаштування тренування відноситься вибір вправ, які будуть входить в це тренування. Буде відображається список тільки допустимих вправ.

На рисунку 4.18 наведено графічне представлення відображення списку тренувань і створення тренування.

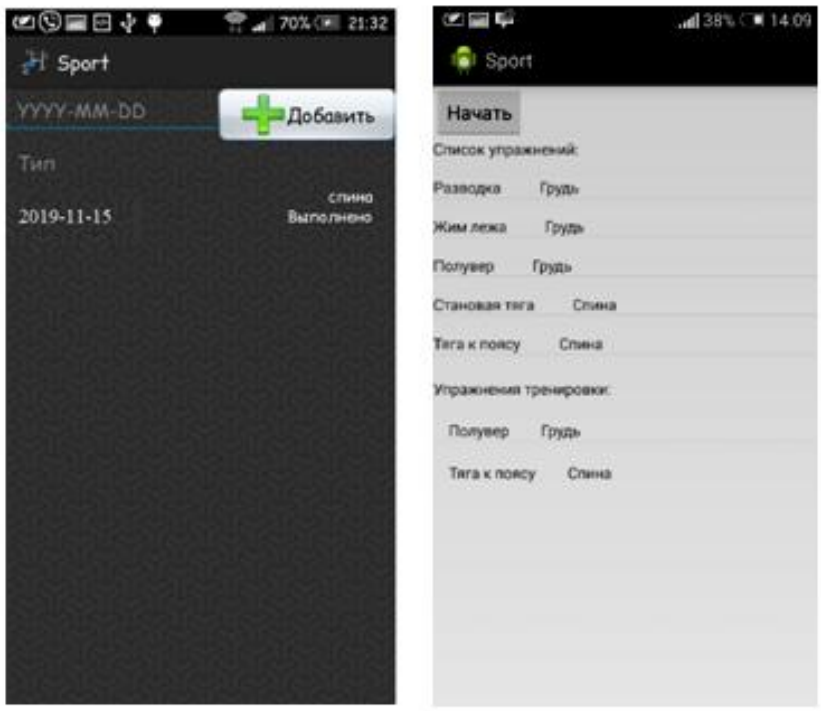


Рисунок 4.18 – Список тренувань і налаштування тренування

Обравши будь-яку вправу, користувач додає її в список тренування. За це все виконання відповідає клас Trainings2. Приклад наведено нижче. Вивід допустимих вправ:

```
     db.openDataBase();
     list = db.readAllEncyclopedia1();
     // находим список
      ListView listView1 = (ListView) findView-
ById(R.id.listView1);
     // создаем адаптер
     EncyclopediaAdapter catAdapter= new Encyclope-
diaAdapter(this, list);
     listView1.setAdapter(catAdapter);
```

Додавання вправи до тренування і виведення всіх вправ, що входять в це тренування забезпечено наступним кодом:

```
     db.openDataBase();
     Intent intent = getIntent();
     int id_tr = Inte-
ger.parseInt(intent.getStringExtra("fname"));
     String Name = ((TextView)
view.findViewById(R.id.textView1)).getText().toString();
```

```
     String NameMG =  ((TextView)
view.findViewById(R.id.textView2)).getText().toString();
     db.saveExercises1(Name, id_tr, NameMG);
     list1 = db.readAllExercises1(id_tr);
     // ListView listView2 = (ListView) findView-
ById(R.id.listView2);
     ExercisesAdapter catAdapter1= new ExercisesA-
dapter(Trainings2.this, list1);
     listView2.setAdapter(catAdapter1);
```

Для використання розробленого в рамках магістерської роботи хмарного сервісу «Фітнес-студія» необхідною умовою є наявність мобільного пристрою з версією операційної системи Android 5.0 або більш пізнішою. А також наявністю вільного місця в пам’яті в мобільного телефону не менш двох мегабайт. При запуску встановленого додатка, відкривається форма введення логіну і паролю. При першому запуску програми користувачу необхідно пройти процедуру реєстрації, яка відбувається у формі реєстрації, перехід у вікно реєстрації відбувається з форми введення логіну і пароля.

При першому запуску сервісу необхідно зайди в пункт додатку «Особисті дані», в якому користувач введе свої дані: ім'я, вік та інші біометричні дані. Ці дані необхідно для подальшої статистики користувача і запису даних в серверну частину базу даних. Здійснивши реєстрацію у додатку сервісу, користувач може вибрати одну з двох ролей: клієнт або тренер.

У розділі меню додатка Групи, користувач може знайти групи, які його цікавлять, стати учасником групи, почати виконувати групові тренування і залишити коментар в меню групи.

Для запуску тренування необхідно вибрати пункт на головній сторінці «Тренування». Далі обрати програму, призначену для користувача або групову, або створити нову програму користувача. Створити тренування для цієї програми, вибрати її, на наступній сторінці додати вправи які будуть входити до цього тренування і розпочати тренування.

Під час тренування користувачу необхідно вносити підходи вправи, записуючи кількість вправ і вагу. При необхідності користувач, якщо не знає, як правильно виконувати вправу, може натиснути кнопку енциклопедії з

описом вправи. Виконавши тренування, користувач повинен натиснути кнопку «Записати тренування».

Створенням і управлінням групою може займатися тільки користувач-тренер. Тренер при створенні тренування повинен задати назву і опис тренування. Після чого створити програму тренувань для групи. Створена група повинна пройти перевірку у адміністратора системи і, якщо її схвалять, вона потрапляє в публічний доступ користувачам.

Використання розробленого сервісу допоможе всім зацікавленим особам займатися тренуванням без відвідування спортивних фітнес-залів, отримувати консультації фітнес-тренерів, слідкувати за статистикою тренувань, вступати у цікаві користувачу групи та спілкуватися.

ВИСНОВКИ

В результаті виконання магістерської роботи було виконано дослідження і аналіз застосування моделей хмарних технологій, та згідно з обраною архітектурою та обраним видом хмарного сервісу, розроблена структура хмарного сервісу «Фітнес-студія», виконана програмна реалізація.

Створений хмарний сервіс «Фітнес-студія», реалізує як локальні функції з організації тренувань на базі OS Android, так і забезпечує можливість реалізації серверних функцій зберігання інформації, вибору та участі в фітнес групах, отримання консультацій тренера.

Для здійснення проектування та розробки хмарного сервісу «Фітнес-студія» проведено аналіз та вибір середовища та інструментів розробки, досліджені механізми реалізації клієнт-серверного спілкування. Була спроектована реляційна база даних, для зберігання інформації мобільного додатку. Були спроектовані і реалізовані функції роботи з серверної базою даних, розроблений графічний інтерфейс користувачів та реалізовані функції розділів: енциклопедія, тренування, їжа, статистика, групи, особисті дані. Сервіс забезпечує також реалізацію функцій підсистеми тренера, які розроблені засобами середовища Eclipse з доповненням Android SDK, на мові Java.

Результатом магістерської роботи є розроблений хмарний сервіс «Фітнес-студія», що забезпечує користувачів, бажаючих займатися спортом, зручним мобільним додатком. Використання розробленого сервісу значно допоможе людині підтримувати спортивну форму без очного відвідування фітнес-центрів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Облачные вычисления: что же это такое? URL: https://www.itweek.ru/its/article/detail.php?ID=135408 (дата звернення: 25.08.2019).
2. Облачные вычисления: тенденции развития и основные «игроки». Часть 2. URL: https://npsod.ru/analytics/1956.html (дата звернення: 05.09.2019).
3. Про фізичну культуру і спорт від 24.12.1993 № 3808-XII URL: https://zakon.rada.gov.ua/laws/main/3808-12 (дата звернення: 07.09.2019).
4. Разработка ПО (мировой рынок). URL: http://www.tadviser.ru/index.php/Статья:Разработка_ПО_%28мировой_рынок%29 (дата звернення: 10.10.2019).
5. Ринок хмарних послуг в Україні зріс на 70% за минулий рік – дослідження. Mind.ua. URL: https://mind.ua/news/20202735-rinok-hmarnih-poslug-v-ukrayini (дата звернення: 16.10.2019).
6. Приложение Nike+ Training Club для iOS и Android. Nike.com (RU). URL: https://www.nike.com/ru/ru_ru/c/nike-plus/training-app (дата звернення: 16.10.2019).
7. PumpUp – Fitness Community 6.0.1 Загрузить APK для Android – Aptoide. URL: https://pumpup.ru.aptoide.com/ (дата звернення: 16.10.2019).
8. App Store: Jillian Michaels Fitness App. URL: https://apps.apple.com/ru/app/jillian-michaels-fitness-app/id1190148494 (дата звернення: 18.10.2019).
9. Автостопом по галактике «Cloud Computing». Статьи. Компьютерное Обозрение. URL: https://ko.com.ua/avtostopom_po_galaktike_cloud_computing_102566 (дата звернення: 04.10.2019).

10. З. Медникс. Программирование для Android: Программирование на Java для нового поколения мобильных устройств. Пер. с англ. СПб.: Питер, 2013. 560 с.
11. Обзор платформы Eclipse – как её использовать. Статьи о программном обеспечении. URL: http://hightech.in.ua/content/art-eclipse-platform (дата звернення 28.10.2019).
12. Люк Веллинг, Лора Томсон. Разработка веб-приложений с помощью PHP и MySQL. М.: Вильямс, 2010. 848 с.