МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК
СЕКЦІЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ПРОЕКТУВАННЯ

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: «Мобільний додаток моніторингу характеристик літій-іонних батарей»

за спеціальністю 122 «Комп'ютерні науки»,
освітньо-професійна програма «Інформаційні технології проектування»

Виконавець роботи: студент групи ІТ-61-8 Кобцов Володимир Юрійович

Кваліфікаційна робота бакалавра
захищена на засіданні ЕК
з оцінкою _____ «___» _____2020 р.

Науковий керівник _____ к.т.н., доц., Алексенко О.В.
(підпис) (науковий ступінь, вчене звання, прізвище та ініціали)

Голова комісії _____ Шифрін Д.М.
(підпис) (науковий ступінь, вчене звання, прізвище та ініціали)

Засвідчую, що у цій дипломній роботі немає
запозичень з праць інших авторів
без відповідних посилань.

Студент _____
(підпис)

Суми-2020

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук
Секція інформаційних технологій проектування
Спеціальність 122 «Комп'ютерні науки»
Освітньо-професійна програма «Інформаційні технології проектування»

ЗАТВЕРДЖУЮ
Зав. секцією ІТП

_____ В. В. Шендрик
«__» _____ 2020 р.

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА СТУДЕНТУ

Кобцов Володимир Юрійович

1 Тема роботи *Мобільний додаток моніторингу характеристик літій-іонних батарей*

керівник роботи *Алексенко Ольга Василівна, к.т.н., доцент*_____,

затверджені наказом по університету від « 14 » травня 2020 р. № 0576-III

2 Строк подання студентом роботи «1» червня 2020 р.

3 Вхідні дані до роботи перелік вимог на розробку мобільного додатку моніторингу характеристик літій-іонних батарей.

4 Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) аналіз процесів дослідження літій-іонних батарей, проектування мобільного додатку, розробка мобільного додатку.

5 Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) актуальність, мета та задачі, об'єкт та предмет дослідження, аналіз існуючих систем візуалізації, моделювання процесу моніторингу характеристик літій-іонних батарей, декомпозиція процесу моніторингу характеристик літій-іонних батарей, засоби реалізації, модель даних у формі ER-діаграми, діаграма варіантів використання мобільного додатку, архітектура системи моніторингу, UML діаграма послідовності побудови графіку за вхідними даними, приклад роботи мобільного додатку, висновки.

6 Консультанти розділів роботи:

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7 Дата видачі завдання_____01.10.2019_____

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Узгодження технічного завдання	14.02.20 – 20.02.20	
2	Огляд досліджень роботи батарей	21.02.20 – 26.02.20	
3	Аналіз існуючих систем візуалізації	27.02.20 – 02.02.20	
4	Постановка задачі щодо створення мобільного додатку	03.02.20 – 06.03.20	
5	Встановлення програмного забезпечення	07.03.20 – 10.03.20	
6	Розгортання серверу бази даних	11.03.20 – 13.03.20	
7	Планування робіт	14.03.20 – 17.03.20	
8	Розробка модуля авторизації	18.03.20 – 23.03.20	
9	Розробка модуля головного меню	24.03.20 – 29.03.20	
10	Розробка модуля історичних замірів	30.03.20 – 03.04.20	
11	Розробка модуля поточного заміру	04.04.20 – 09.04.20	
12	Розробка модуля налаштувань графіку	10.04.20 – 15.04.20	
13	Розробка модуля збереження графіку	16.04.20 – 21.04.20	
14	Підключення розроблених модулів	22.04.20 – 30.04.20	
15	Тестування мобільного додатку	01.05.20 – 15.05.20	
16	Реліз мобільного додатку	16.05.20 – 31.05.20	

Студент _____ Кобцов В.Ю.
(підпис)

Керівник роботи _____ к.т.н., доц. Алексенко О.В.
(підпис)

РЕФЕРАТ

Тема кваліфікаційної роботи бакалавра «Мобільний додаток моніторингу характеристик літій-іонних батарей».

Пояснювальна записка складається зі вступу, 3 розділів, висновків, списку використаних джерел із 21 найменування та 4 додатків. Загальний обсяг роботи − 81 сторінка, у тому числі 47 сторінок основного тексту, 3 сторінки списку використаних джерел та 31 сторінка додатків.

Кваліфікаційну роботу бакалавра присвячено розробці мобільного додатку моніторингу характеристик літій-іонних батарей.

У роботі були вивчені характеристики роботи літій-іонних батарей, розглянуті існуючі системи аналізу та візуалізації експериментальних даних, сформульоване технічне завдання на розробку, побудовані моделі процесу моніторингу та розроблюваного додатку, спроектовано базу даних. Останнім кроком було виконано опис архітектури мобільного додатку, етапів розробки та наведено приклад використання.

Результатом проведеної роботи є реалізований мобільний додаток моніторингу характеристик літій-іонних батарей.

Практичне значення роботи полягає у полегшенні аналізу експериментальних даних щодо характеристик літій-іонних батарей завдяки застосуванню розроблюваного мобільного додатку.

Ключові слова: мобільний додаток, Android Studio, літій-іонна батарея, база даних, layout, activity, fragment, java, php, MySQL.

ЗМІСТ

ВСТУП

Літій-іонні батареї доволі широко використовуються в якості джерела живлення для планшетів, смартфонів, відео та фото камер, ноутбуків, портативних зарядних пристроїв, а також електромобілів.

Для покращення роботи літій-іонних батарей потрібно виконувати обробку та аналіз великих обсягів експериментальних даних, щодо потреб візуального представлення цих даних [1].

Необхідність візуалізації цифрових даних для істотного зменшення часу виконання аналізу експериментальних даних моніторингу електрохімічних характеристик електродів літій-іонних батарей, зумовлює актуальність розроблення мобільного додатку.

Об’єктом дослідження є засоби аналізу результатів моніторингу електрохімічних характеристик електродів літій-іонних батарей.

Предметом дослідження є засоби візуалізації експериментальних даних за допомогою мобільних додатків.

Мета дослідження полягає у створенні мобільного додатку моніторингу електрохімічних характеристик електродів для літій-іонних батарей, використання якого розширить можливості дослідників.

Основні задачі роботи:

– аналіз систем-аналогів та розроблення технічного завдання на основі дослідження предметної області;

– вибір та налаштування інструментів реалізації;

– макетування користувацького інтерфейсу та визначення структури інформації, з якою буде працювати додаток;

– проектування алгоритмів візуалізації даних;

– реалізація мобільного додатку.

1 АНАЛІЗ ПРОЦЕСІВ ДОСЛІДЖЕННЯ ЛІТІЙ-ІОННИХ БАТАРЕЙ

1.1 Огляд досліджень роботи літій-іонних батарей

Розвиток електроенергетичних систем останнім часом характеризується інтенсивним зростанням використання батарей на основі літій-іонних акумуляторів. Цьому в значній мірі сприяє прогрес в таких областях, як поновлювані джерела енергії, мережеві технології обробки даних і управління на базі технологій інтелектуальних мереж електропостачання. Акумуляторна батарея на основі літій-іонних акумуляторів володіє оптимальним співвідношенням масо-габаритних характеристик і накопиченої енергії і, як наслідок, може використовуватися не тільки в стаціонарних, але і мобільних системах різного призначення [2].

В основі безпечної та ефективної експлуатації літій-іонних батарей лежить безперервний контроль за основними параметрами акумуляторів, а також моніторинг історії (зарядки/розрядки) і фактичної ємності.

При формуванні багатоелементної послідовно з'єднаної літій-іонної акумуляторної батареї (ЛІАБ) виникає проблема розкиду напруги і рівнів заряду окремих акумуляторних осередків. Після досягнення хоча б однієї з осередків критичної напруги при розряді необхідно відключити накопичувач від навантаження, так як подальший розряд спричинить порушення вимог експлуатації. У цьому випадку ємність батареї буде визначатися ємністю найслабшої осередки.

Таким чином, щоб підвищити експлуатаційні характеристики акумуляторних батарей, необхідно управляти зарядним процесом накопичувача і балансуванням акумуляторних осередків.

Способи балансування поділяються на два типи: активне і пасивне балансування. Вид балансування акумуляторної батареї визначається технічними, експлуатаційними, економічними вимогами, а також особливостями побудови накопичувача.

Перший спосіб балансування літій-іонної батареї полягає у відведенні надлишків енергії з акумуляторів, напруга на яких перевищує найменше з напруги на всіх осередках батареї на певну величину. Запропонований спосіб збільшує час розряду багатоелементної батареї, проте він витратний з точки зору тривалості часу підготовки накопичувача і технічної складності реалізації.

Другий спосіб балансування літій-іонної батареї ефективний для автономних систем, підзарядка яких проводиться в довільний момент часу, і неприйнятний для накопичувачів, час заряду яких визначається оператором, тобто накопичувачів енергії мобільних енергосистем і електротранспортних засобів.

Отже, необхідність візуалізації цифрових даних для істотного зменшення часу виконання аналізу експериментальних даних моніторингу електрохімічних характеристик електродів літій-іонних батарей, зумовлює актуальність розроблення мобільного додатку за допомогою якого дослідники матимуть доступ до даних моніторингу у будь-який момент часу, без потреби знаходитись на робочому місці.

1.2 Аналіз існуючих систем аналізу та візуалізації експериментальних даних та засобів їх створення

Для порівняльного аналізу були обрані мобільні додатки які аналізують кількість споживаної енергії акумулятором, а саме: Battery Charging Monitor - Ampere Meter, GSam Battery Monitor, Battery Monitor.

Battery Charging Monitor - Ampere Meter – це мобільний додаток за допомогою якого вимірюється швидкість зарядження акумулятора в режимі реального часу та швидкість розряду в міліамперах (мА) [3]. На рисунку 1.1 наведено приклад моніторингу зарядки акумулятора, а на рисунку 1.2 наведено приклад моніторингу розрядки.

Позитивне значення та графік синього кольору відображає моніторинг зарядки телефону, а від’ємні значення та графік червоного кольору відображає моніторинг

розрядки. Під графіком зарядки та розрядки відображається поточний статус (зарядка/розрядка), відсоток заряду батареї, значення напруги та температура батареї, технологія батареї (Li-ion), максимальний вміст батареї (mAh), значення оперативної пам’яті та температура процесору.

Рисунок 1.1 – Моніторинг зарядки

Рисунок 1.2 – Моніторинг розрядки

Також вимірюється температура, доступна оперативна пам’ять та температура процесора яка використовується під час використання програм або ігор. На рисунку 1.2 наведено приклад аналізу навантаження на телефон.

Рисунок 1.3 – Моніторинг навантаження

Мобільний додаток дозволяє переглянути та видалити історію моніторингу. На рисунку 1.3 наведено перегляд історії моніторингу.

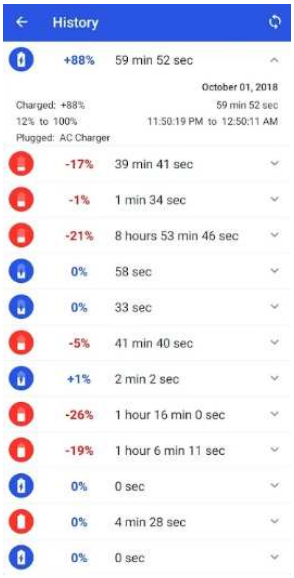

Рисунок 1.4 – Історія моніторингу

GSam Battery Monitor – це мобільний додаток за допомогою якого відображається споживання батареї іншими додатками або програмами [4]. На рисунку 1.5 наведено споживання батареї.

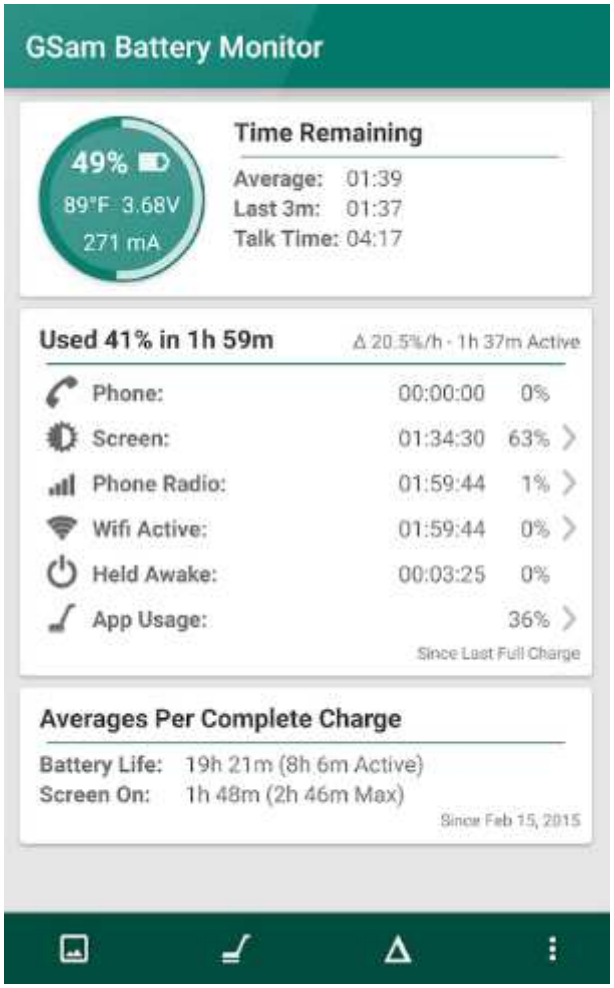

Рисунок 1.5 – Споживання батареї

Також є можливість створення списку додатків за допомогою їх сортування по використанню процесора, сенсору, кількості активності додатків, батареї та оперативної пам’яті. На рисунку 1.6 наведено відсортований список додатків.

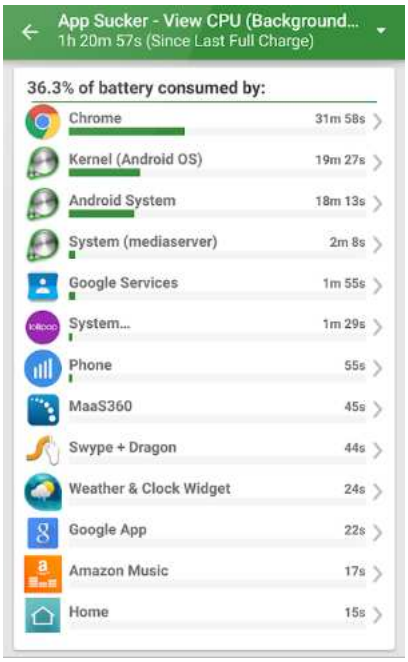

Рисунок 1.6 – Список додатків

Додаток дозволяє переглядати історію на графіку, який наведений на рисунку 1.7, а також додавати на головний екран віджет, за допомогою якого відображається статус батареї. На рисунку 1.8 наведено віджет, що відображає статус батареї.

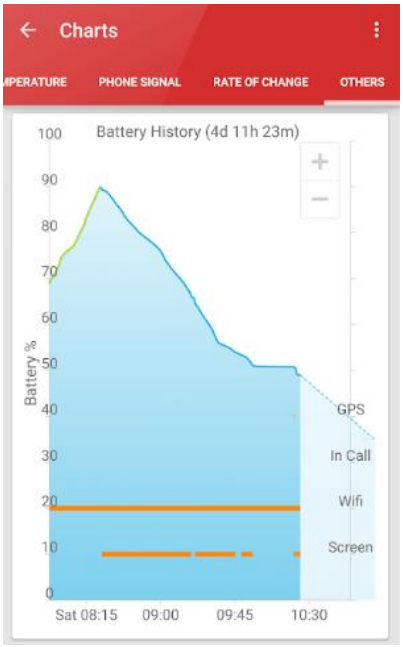

Рисунок 1.7 – Перегляд історії

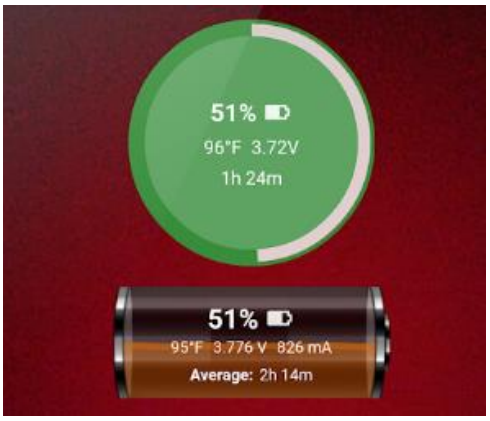

Рисунок 1.8 – Віджет статусу

Battery Monitor – це мобільний додаток що дозволяє контролювати температуру акумулятора та інформацію в режимі реального часу, включаючи температуру акумулятора, стан живлення (зарядка/розрядка), напругу, температуру батареї та рівень заряду [5]. На рисунку 1.9 наведено приклад роботи моніторингу рівня заряду акумулятора та її температури.

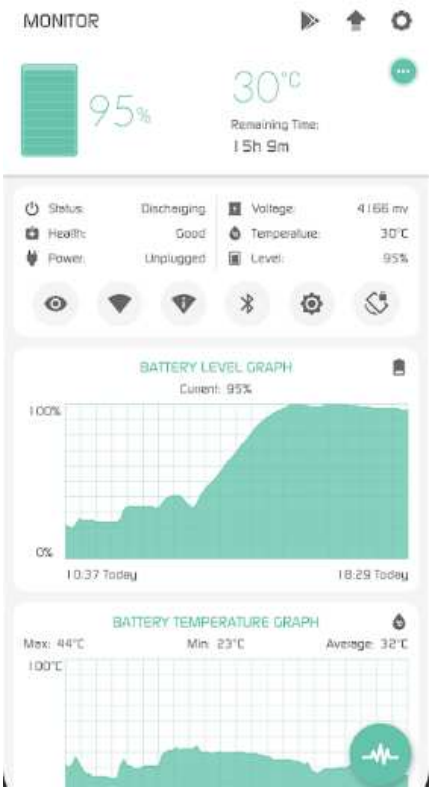

Рисунок 1.9 – Моніторинг акумулятора

Додаток дозволяє налаштувати кольорову гамму, тему та відображення віджетів на головному екрані. На рисунку 1.10 наведено вікно налаштувань мобільного додатку.

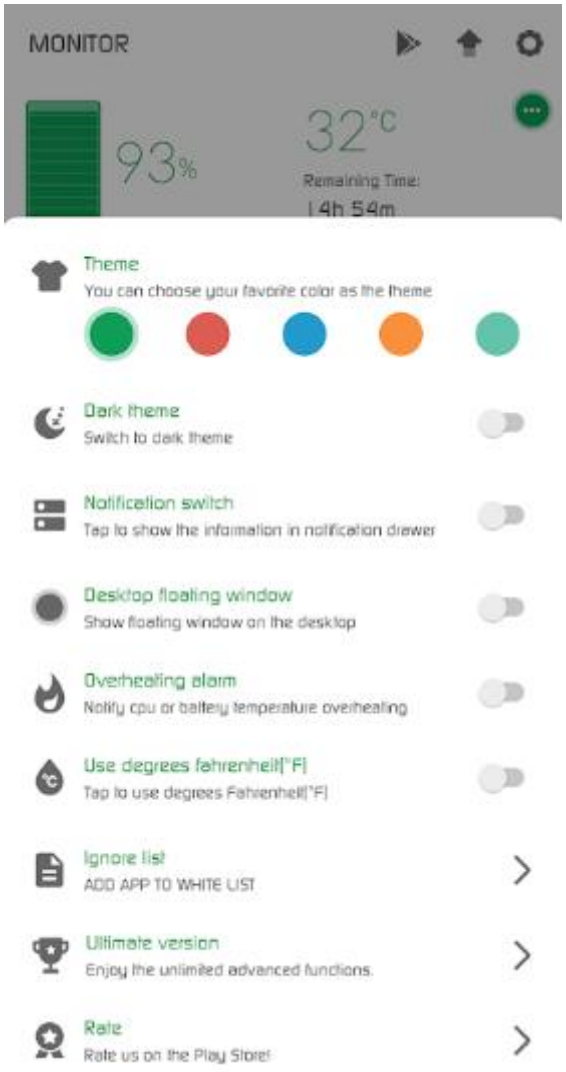

Рисунок 1.10 – Вікно налаштувань

За результатами аналізу мобільних додатків моніторингу акумуляторів було оформлено та заповнено порівняльну характеристику за деякими критеріями, яка представлена у таблиці 1.1.

Таблиця 1.1 – Порівняльна характеристика мобільних додатків

Критерії порівняння	**Наявність у відповідному мобільному додатку**		
	Battery Charging Monitor – Ampere Meter	**GSam Battery Monitor**	**Battery Monitor**
Наявність навігації	+	+	-
Зміна орієнтації екрану	+	-	-
Продуктивність	+/-	+	+/-
Стабільність	+/-	+	+
Якість візуалізації графіків	+	+/-	+
Апаратне прискорення	+	-	-
Адаптивність	+	+/-	+/-
Збереження результатів моніторингу	-	-	-
Перегляд історії моніторингу	+	-	-

1.3 Постановка задачі щодо створення мобільного додатку

Як було визначено вище, для удосконалення літій-іонних батарей необхідні засоби аналізу їх режимів роботи, зокрема засоби візуалізації результатів вивчення електрохімічних характеристик електродів.

Виходячи із виділеної проблеми була сформульована мета роботи, яка полягає у створенні мобільного додатку моніторингу електрохімічних характеристик електродів для літій-іонних батарей.

Для досягнення мети визначено, що необхідно вирішити такі основні задачі: формування технічного завдання на основі аналізу предметної області, у тому числі систем-аналогів; вибір інструментів реалізації та їх налаштування; макетування користувацького інтерфейсу та визначення структури інформації, з якою буде працювати додаток; проектування алгоритмів візуалізації даних; реалізація мобільного додатку та розроблення інструкції користувача.

Провівши аналіз існуючих технологій які використовуються для розробки мобільних додатків було обрано середовище розробки Android Studio та мова програмування Java. Це середовище підтримує підключення великої кількості бібліотек, які використовуються для побудови графіків, збереження інформації та підключення до локальних баз даних.

Під час реалізації буде використовуватися сервер бази даних, на якому будуть зберігатися вхідні дані для проведення експериментів, а також вихідні дані, які потрібні для аналізу роботи літій-іонних батарей.

Для тестування роботи під час реалізації було встановлено спеціальний емулятор роботи мобільного додатку (virtual device), з його допомогою з’являється можливість переглянути та перевірити поточний стан роботи мобільного додатку.

За результатами огляду процесів дослідження літій-іонних батарей було визначено основні функції розроблюваного мобільного додатку:

– побудова графіку за вхідними даними поточного циклу;
– збереження графіку у вигляді png формату;
– перегляд історії замірів;
– можливість видалення певної історії заміру;
– налаштування зовнішнього вигляду графіку: кольору та розміру точок та тексту;
– можливість запуску та зупинки експерименту із візуалізацією результатів замірів.

Детальний опис з призначенням, вимогами та макетами сторінок мобільного додатку представлений у технічному завданні (ТЗ), представленому у додатку А.

2 ПРОЕКТУВАННЯ МОБІЛЬНОГО ДОДАТКУ

2.1 Моделювання процесу моніторингу характеристик

Контекстна діаграма процесу моніторингу даних складається з головного блоку (activity), в якому розміщується назва процесу та стрілок входу, управління, механізму та виходу [6]. Контекстна діаграма наведена на рисунку 2.1.

Процес моніторингу на початку роботи приймає на вхід набір даних експерименту за допомогою яких буде відбуватися побудова поточного графіку та початкові налаштування для відображення графіків, після чого підключаються механізми взаємодії база даних і співробітник, які впливають на роботу процесу та методи управління вимогами до графіків які необхідні для контролю відображення графіку. На виході відображається графічне представлення результатів експерименту, файли збережених графіків та історія замірів.

Рисунок 2.1 – Контекстна діаграма

У процесі декомпозиції процесу моніторингу було виділено 6 підпроцесів:

Побудова графіку за вхідними даними – додаток будує та відображає графік у поточному циклі експерименту, збереження графіку – користувач може зберегти поточний графік у форматі *.png , перегляд історії замірів – користувач може переглянути архівні дані замірів за минулий час, можливість видалення певної історії заміру – користувач може видалити обраний замір з архіву, налаштування зовнішнього вигляду графіку – користувач може налаштувати зовнішній вигляд графіку, запуск та зупинка експерименту – користувач може запускати експеримент та переглядати його поточний стан або зупиняти, якщо він вже запущений.

Виконана декомпозиція наведена на рисунку 2.2.

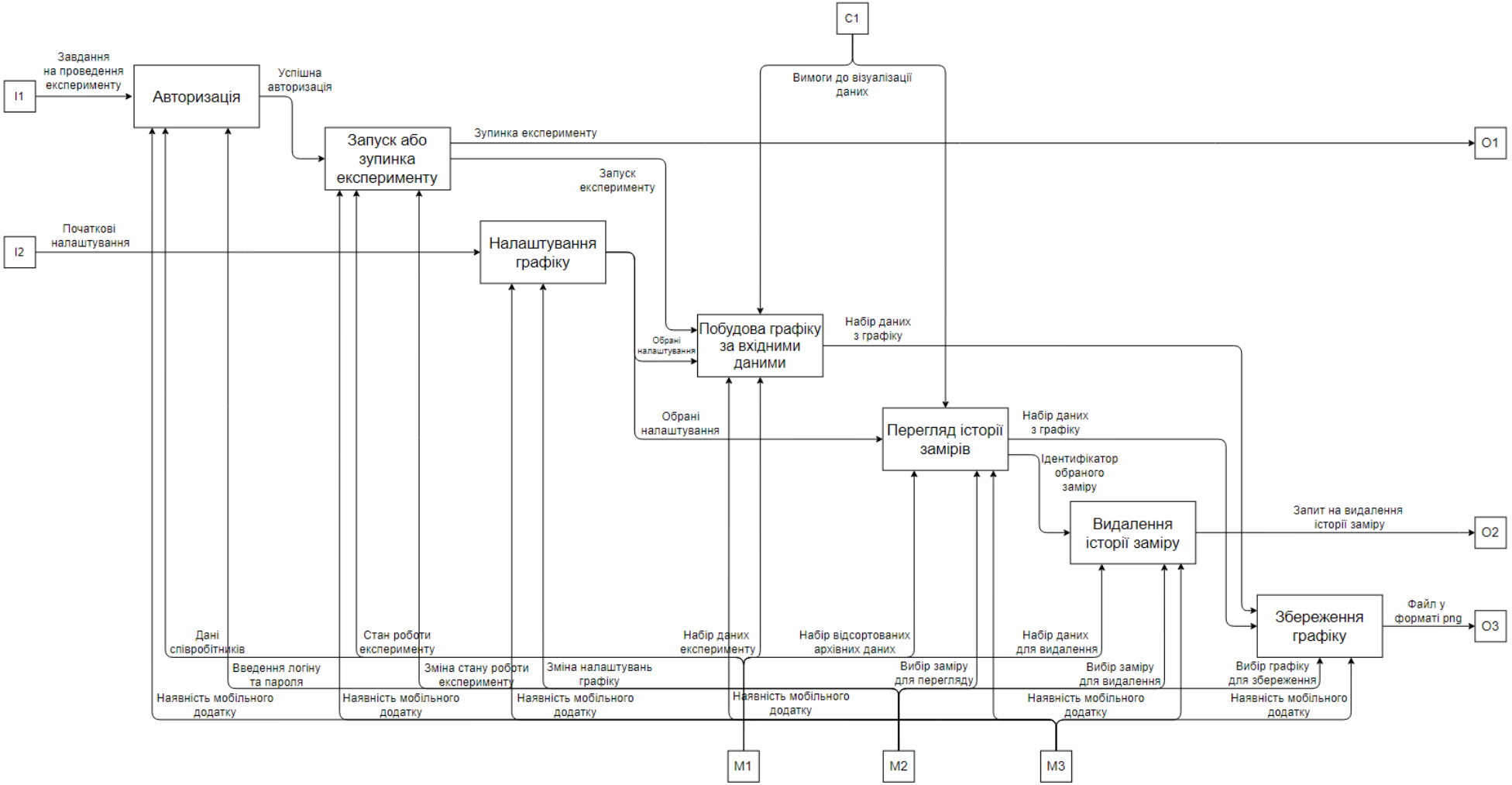

Рисунок 2.2 – Декомпозиція процесу моніторингу

2.2 Моделювання додатку

Діаграма варіантів використання представлена як взаємозв’язок акторів, артефактів та варіантів використання [7]. Актор – це сутність, яка перебуває поза модельованою системою і безпосередньо взаємодіє з нею, артефакти – це елементи інформації, які використовуються в процесі розробки, а варіанти використання – це послідовність дій, які система виконує для отримання важливого для актора результату [8].

Актори:

– Співробітник – користувач, який має доступ до мобільного додатку;

– База даних – база даних, у якій зберігаються дані експерименту.

Артефакти:

– MySQL – дамп бази даних, на якій зберігаються дані для експерименту.

Варіанти використання:

– UC01 Побудова графіку за вхідними даними – додаток будує та показує користувачеві графік характеристик, заміряних у поточному циклі експерименту;

– UC02 Збереження графіку – користувач може зберегти поточний графік у форматі *.png;

– UC03 Перегляд історії замірів – користувач може переглянути архівні дані замірів за минулий час;

– UC04 Можливість видалення певної історії заміру – користувач може видалити обраний замір з архіву;

– UC05 Налаштування зовнішнього вигляду графіку – користувач може налаштувати зовнішній вигляд графіку;

– UC06 Можливість запуску та зупинки експерименту із візуалізацією результатів – користувач може запускати експеримент та переглядати його поточний стан або зупиняти, якщо він вже запущений;

– UC07 Необхідність авторизації у системі – для доступу до повного функціоналу мобільного додатку користувачеві необхідно авторизуватись у системі.

Діаграма варіантів використання наведена на рисунку 2.3.

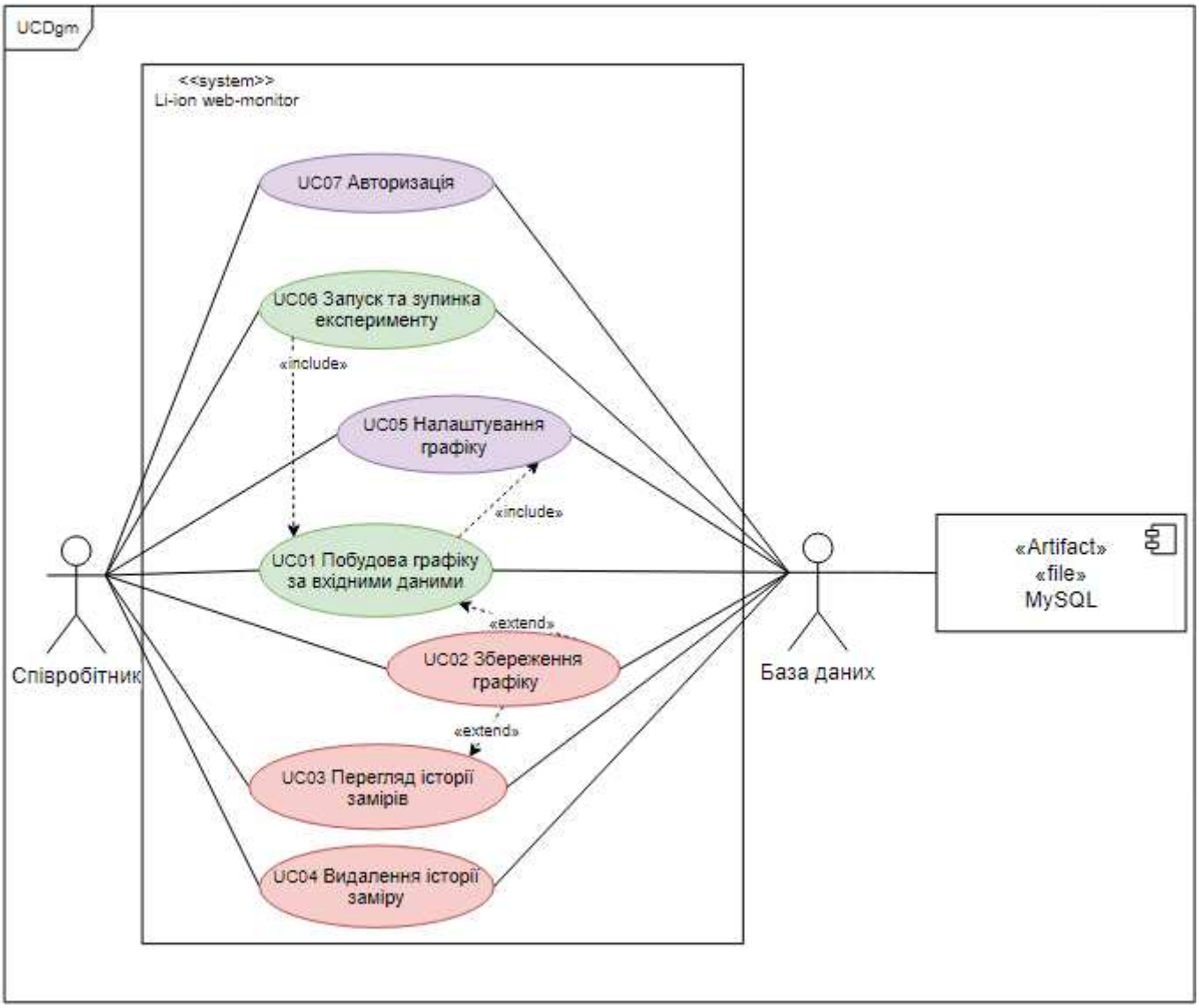

Рисунок 2.3 – Діаграма варіантів використання

Для опису життєвого циклу кожного варіанту використання з рахуванням їх взаємодії з акторами у рамках відповідного запиту використані діаграми послідовності які наведені на рисунку 2.4-2.10.

На рисунку 2.4 наведено діаграму послідовності для UC01 «Побудова графіку за вхідними даними».

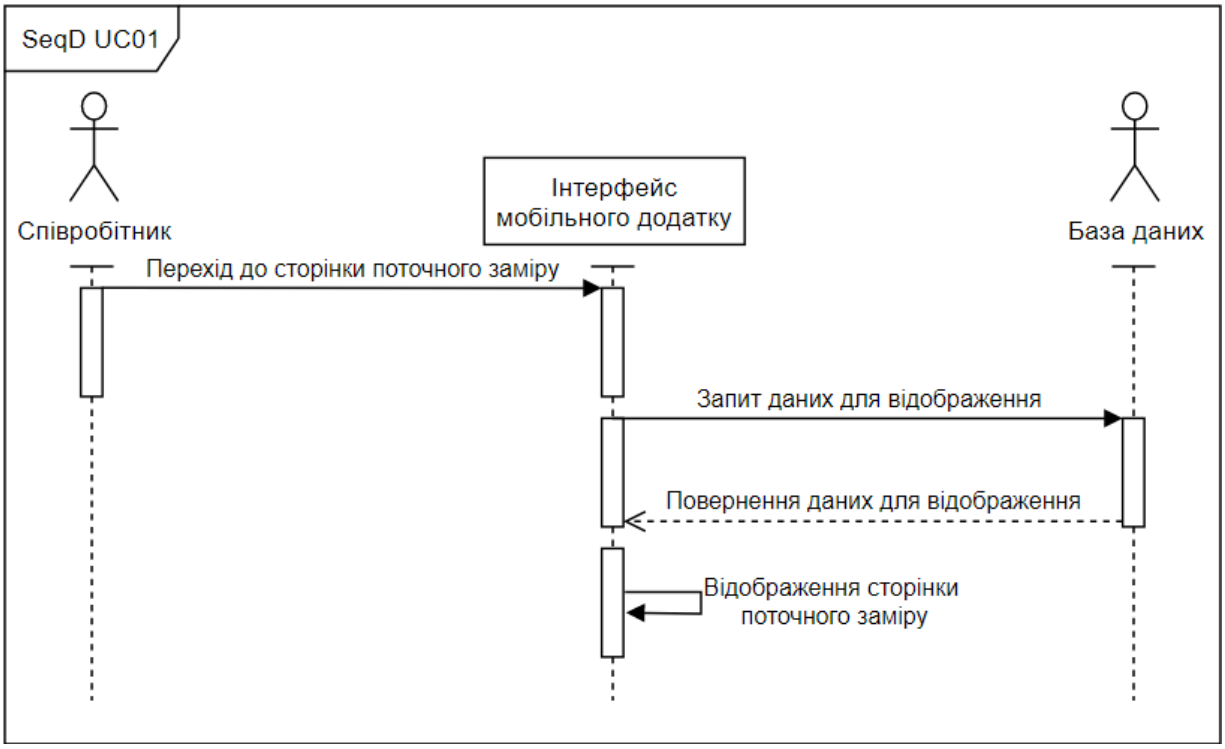

Рисунок 2.4 – Діаграма послідовності «Побудова графіку за вхідними даними»

На рисунку 2.5 наведено діаграму послідовності для UC02 «Збереження графіку».

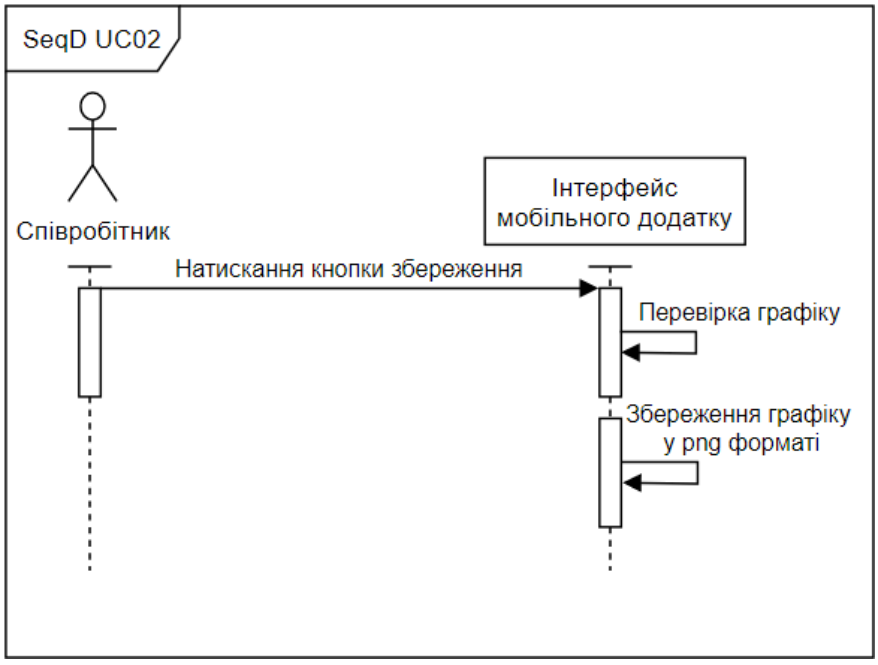

Рисунок 2.5 – Діаграма послідовності «Збереження графіку»

На рисунку 2.6 наведено діаграму послідовності для UC03 «Перегляд історії замірів».

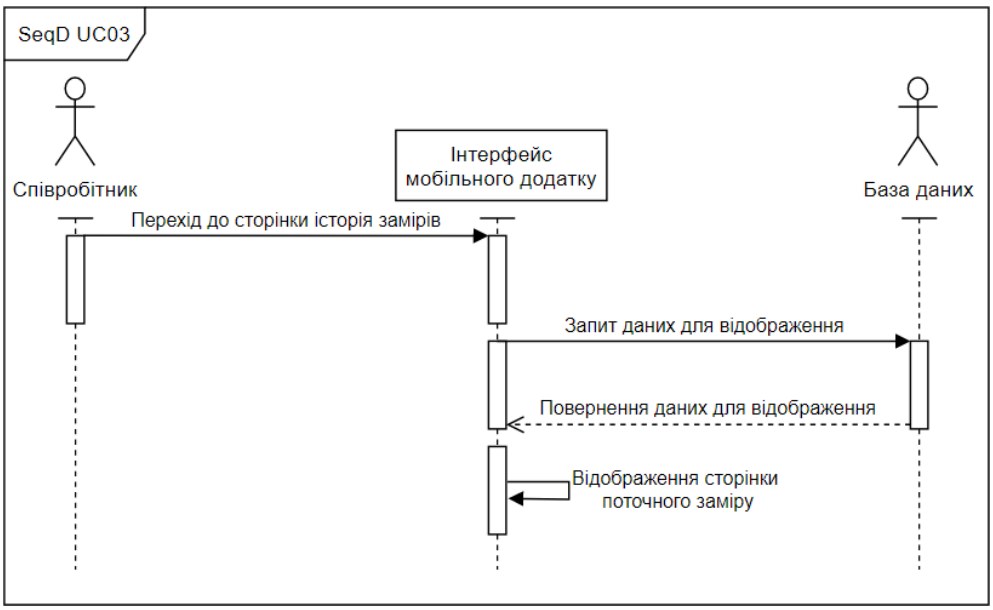

Рисунок 2.6 – Діаграма послідовності «Перегляд історії замірів»

На рисунку 2.7 наведено діаграму послідовності для UC04 «Видалення історії заміру».

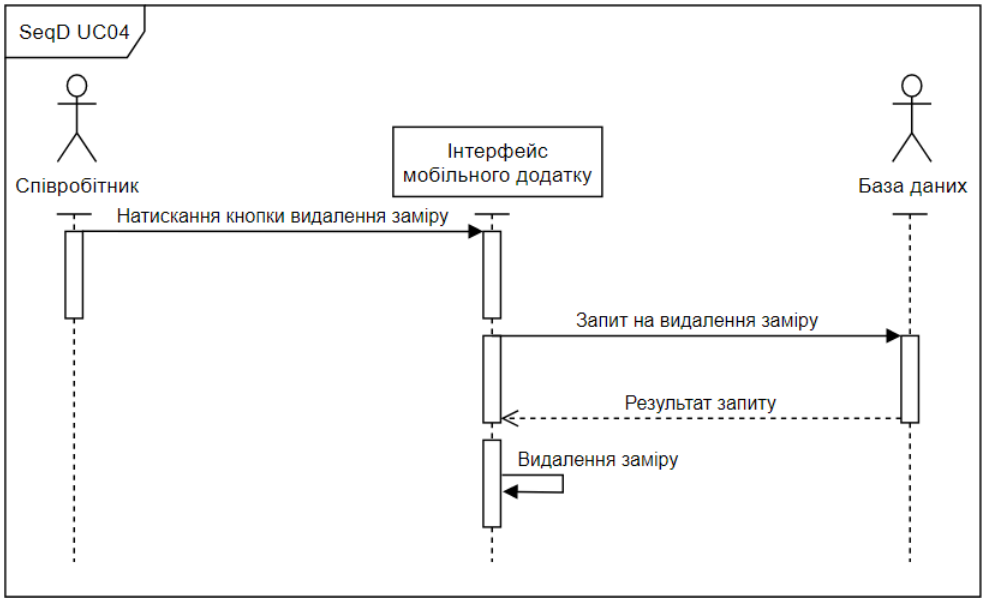

Рисунок 2.7 – Діаграма послідовності «Видалення історії заміру».

На рисунку 2.8 наведено діаграму послідовності для UC05 «Налаштування графіку».

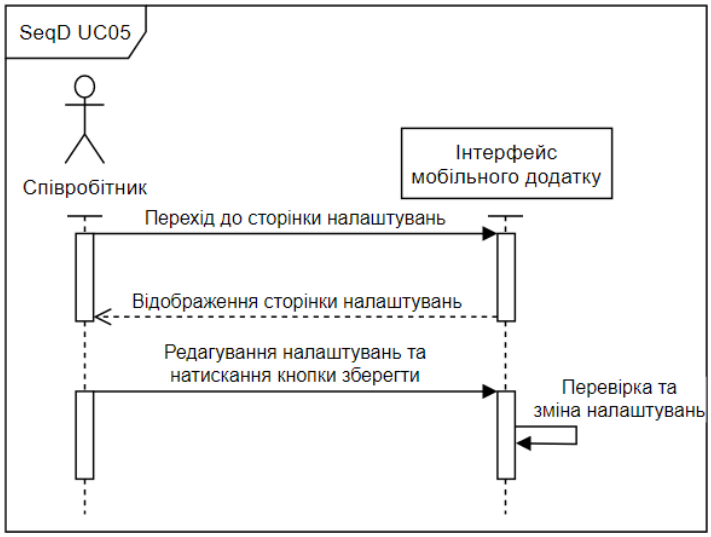

Рисунок 2.8 – Діаграма послідовності «Налаштування графіку»

На рисунку 2.9-2.10 наведено діаграму послідовності для UC06 «Запуск експерименту» та «Зупинка експерименту».

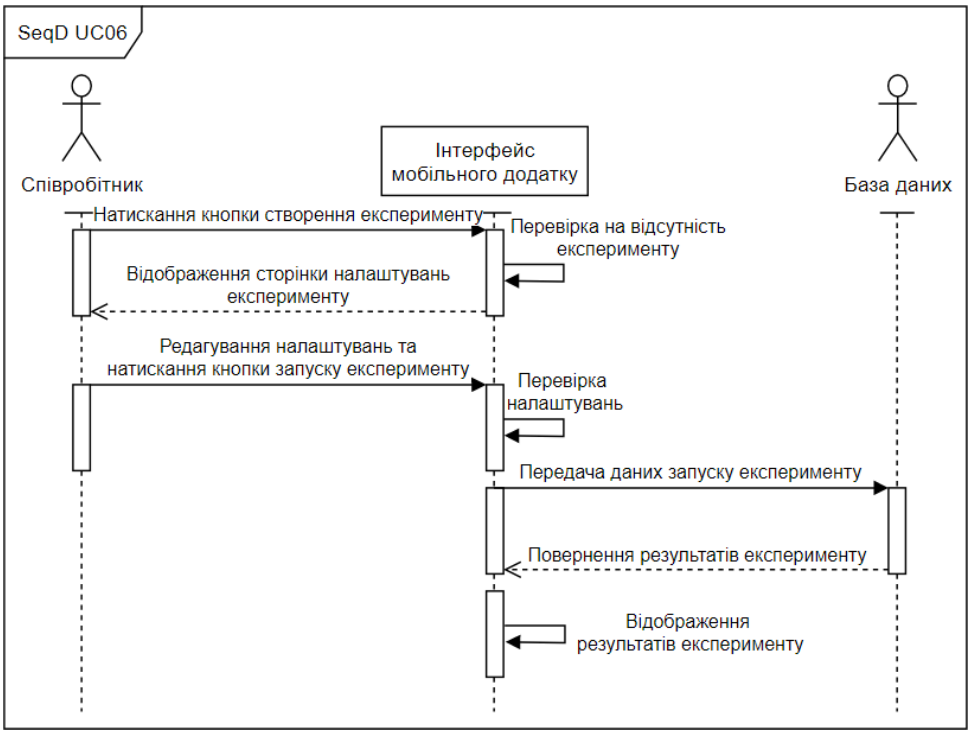

Рисунок 2.9 – Діаграма послідовності «Запуск експерименту»

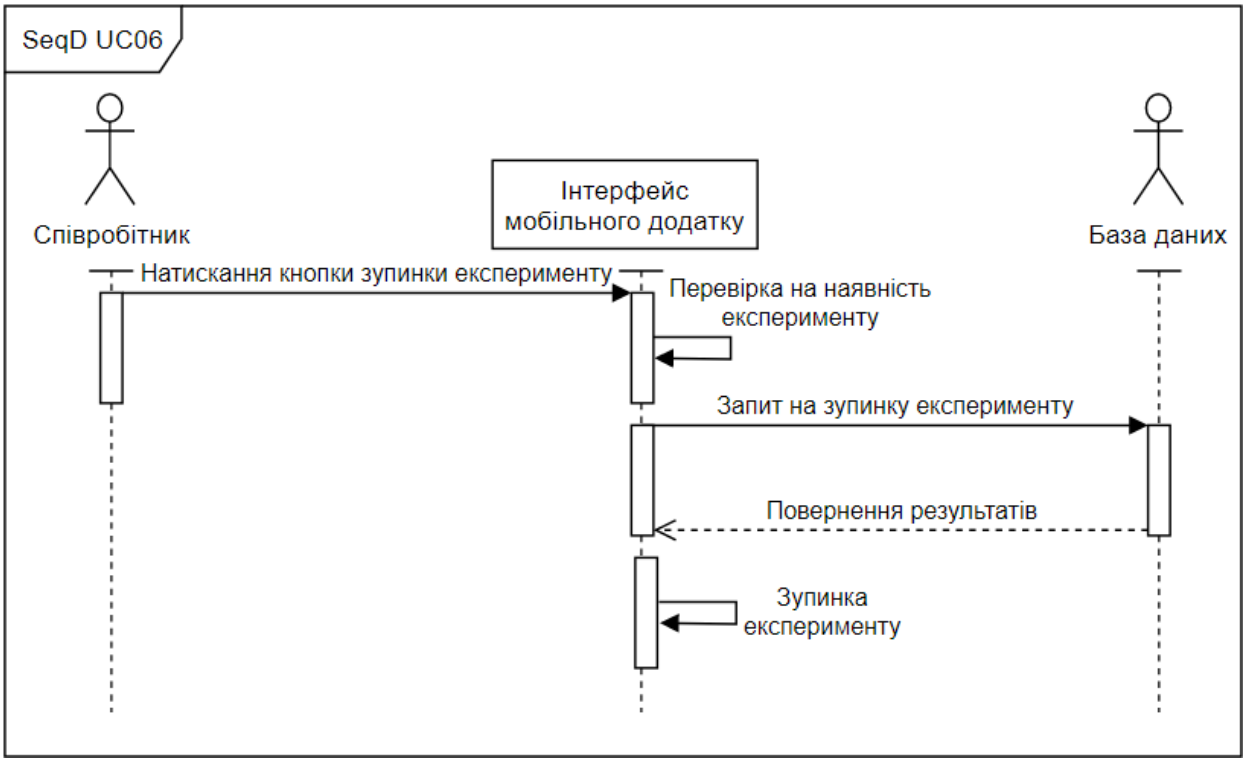

Рисунок 2.10 – Діаграма послідовності «Зупинка експерименту»

На рисунку 2.11 наведено діаграму послідовності для UC07 «Авторизація».

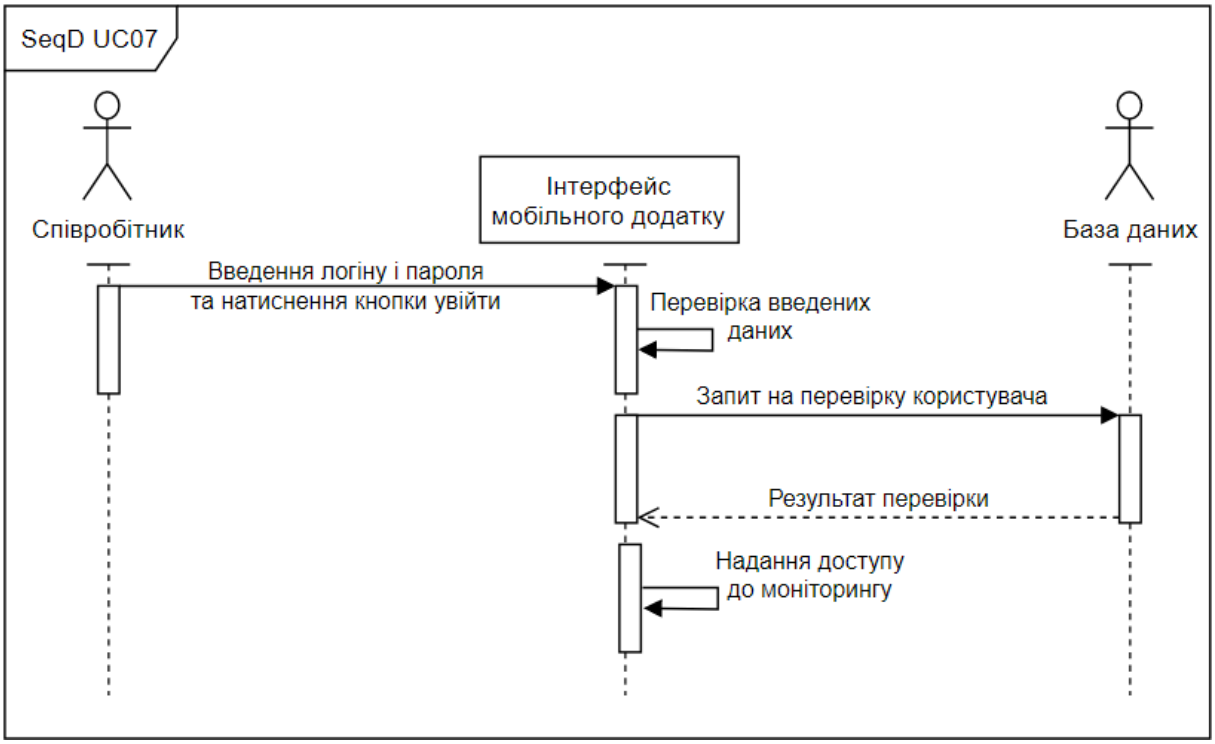

Рисунок 2.11 – Діаграма послідовності «Авторизація»

Макети сторінок користувацького інтерфейсу подані в технічному завданні, наведеному у Додатку А.

2.3 Моделювання використовуваних даних

У ході аналізу предметної області було визначено перелік функцій, які повинна виконувати модельована система, а саме:

– зберігати дані дослідників;
– зберігати поточні значення експерименту;
– зберігати історичні значення експерименту;
– зберігати дані результату проведення експерименту;
– зберігати дату проведення експерименту.

Виділені функції дозволяють визначити структуру інформації, яку повинен використовувати додаток у формі ER-діаграми «сутність-зв’язок» (рис. 2.12), на якій представлені виділені сутності, їх атрибути та зв’язки між ними [9]. У подальшому виділені сукупності інформації перетворені у відповідні таблиці бази даних [10].

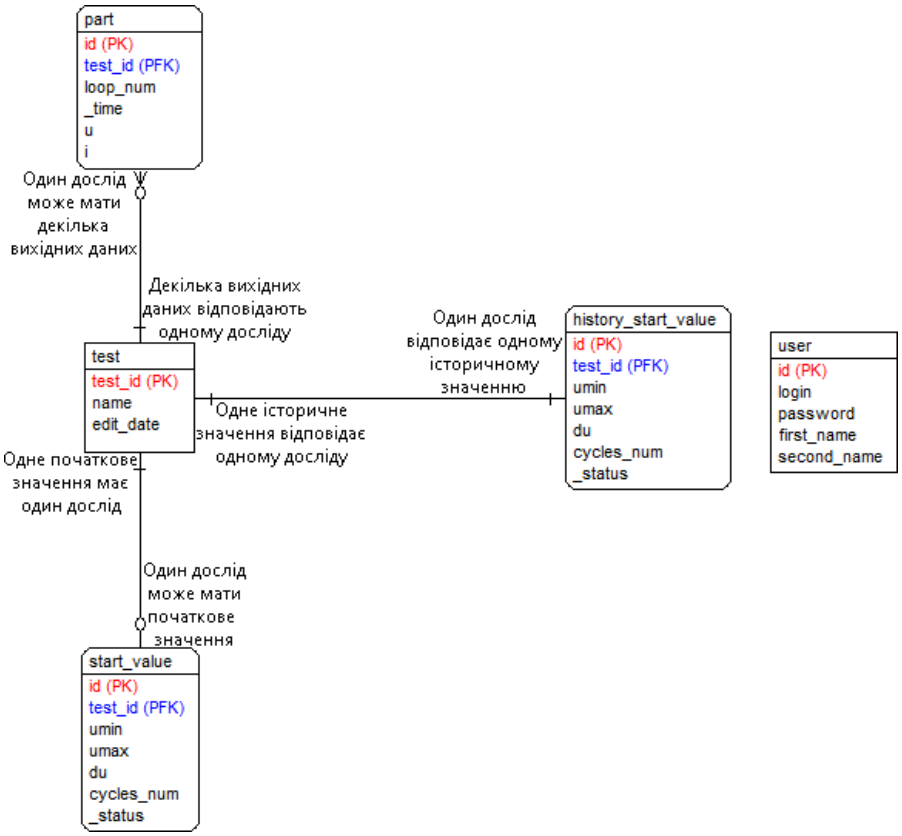

Рисунок 2.12 – ER-діаграма

ER-діаграма складається з 4 зв’язаних таблиць, що містять результати експериментів та окремої таблиці, яка виділена для збереження даних авторизації.

Сутність test призначена для зберігання імені експерименту та дати запуску, має такі атрибути: номер досліду, назва досліду (name) та дата початку досліду (edit_date).

Сутність part призначена для зберігання результатів експерименту, складається з таких атрибутів: номеру значень, номеру досліду (test_id), номеру циклу (loop_num), часу виконання експерименту (_time), значення напруги(u) та значення струму (i).

Сутність history_start_value складається з таких атрибутів: номеру історичних значень, номеру досліду, мінімальної напруги, максимальної напруги, кроку зміни напруги, кількості циклів та стану досліду.

Сутність start_value призначена для зберігання початкових значень для запуску експерименту, складається з таких атрибутів: номеру поточного досліду (test_id), мінімальної напруги (umin), максимальної напруги (umax), кроку зміни напруги (du), кількості циклів (cycles_num) та стану досліду (_status).

Сутність user складається з таких атрибутів: номеру дослідника, логіну, пароля, імені та прізвища дослідника.

Проаналізувавши сутності було перейдено до реалізації структури бази даних. Для цього було представлено імена необхідних таблиць, атрибутів, типів, первинного [11], зовнішнього ключа [12] та обмеження які відображені у таблиці 2.1.

Таблиця 2.1 – Структура бази даних

Таблиця	**Поле**	**Зміст**	**Тип**	**Ключ**	**Обмеження**
Test	test_id	Номер досліду	bigint(20)	PK	not null
	name	Назва досліду	varchar(40)		not null
	edit_date	Дата початку	datetime		not null
Part	id	Номер значень	bigint(20)	PK	not null
	test_id	Номер досліду	bigint(20)	PFK	not null
	loop_num	Номер циклу	bigint(20)		not null
	_time	Часу	bigint(20)		not null
	u	Значення напруги	double		not null
	i	Значення струму	double		not null
History_ start_value	id	Номер історії	bigint(20)	PK	not null
	test_id	Номер досліду	bigint(20)	PFK	not null
	umin	Мінімальна напруга	double		not null
	umax	Максимальна напруга	double		not null
	du	Крок зміни напруги	double		not null
	cycles_num	Кількість циклів	double		not null
	_status	Стан досліду	tinyint(1)		not null

Продовження таблиці 2.1 – Структура бази даних

Таблиця	Поле	Зміст	Тип	Ключ	Обмеження
Start_value	id	Номер поточного заміру	bigint(20)	PK	not null
	test_id	Номер досліду	bigint(20)	PFK	not null
	umin	Мінімальна напруга	double		not null
	umax	Максимальна напруга	double		not null
	du	Крок зміни напруги	double		not null
	cycles_num	Кількість циклів	double		not null
	_status	Стан досліду	tinyint(1)		not null
User	id	Номер дослідника	int(11)	PK	not null
	login	Логін	varchar(15)		not null
	password	Пароль	varchar(60)		not null
	first_name	Ім’я	varchar(20)		not null
	second_name	Прізвище	varchar(20)		not null

3 РОЗРОБКА МОБІЛЬНОГО ДОДАТКУ

3.1 Архітектура системи моніторингу

Для визначення роботи мобільного додатку було побудовано архітектуру яка відображена на рисунку 3.1, на основі архітектурного шаблону MVVM (Model, View Model, View) [13], який є модифікацією більш загального паттерну MVC [14] і який відображає основні три компоненти програмних модулів мобільного додатку, а саме:

– компонент інтерфейсу користувача – це компонент, що містить головну активність (activity), та фрагменти екрану (fragment), які відображаються при взаємодії з мобільним додатком;

– компонент бізнес-логіки – це компонент, що містить усі модулі та дані, які необхідні для роботи мобільного додатку;

– компонент доступу до даних – це компонент, що містить php скрипти, за допомогою яких відбувається з'єднання з базою даних, додавання, редагування, та видалення інформації.

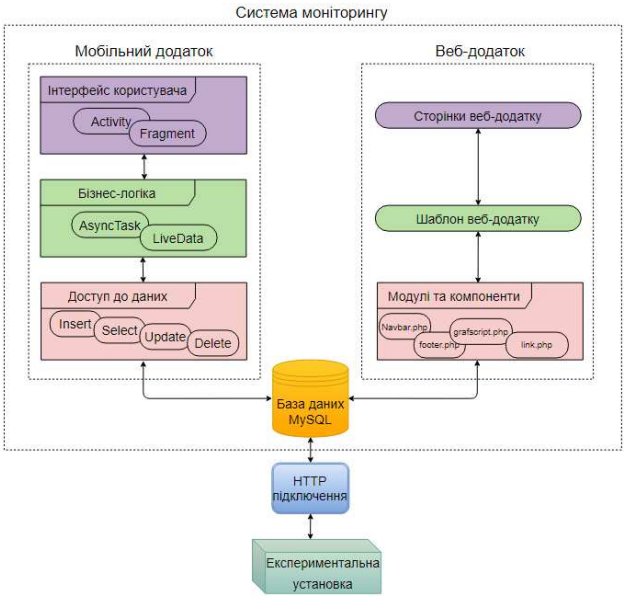

Рисунок 3.1 – Архітектура системи моніторингу

Також, на рисунку 3.1 представлено веб-додаток і базу даних, яка зберігає інформацію про батареї, що надходить від експериментальної установки.

3.2 Програмна реалізація мобільного додатку

Розробка модуля авторизації

Спочатку був налаштований користувацький інтерфейс у layout файлі [15], який дозволяє описати макети сторінок. Файл містить елементи управління, зображення, поля вводу/виводу, activity [16], fragments [17] та інші елементи взаємодії, у ньому налаштовується їх колір та розмір. Редагування layout файлу відбувається за допомогою текстового редактору (рис. 3.2) або дизайнеру (рис. 3.3).

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:background="@color/colorFrame"
    android:padding="20dp"
    android:orientation="vertical"
    android:gravity="center"
    tools:context=".AuthorizationActivity">

    <TextView
        android:id="@+id/textViewAuthorization"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:layout_margin="20dp"
        android:textSize="25sp"
        android:text="Авторизація" />

    <EditText
        android:id="@+id/et_login"
        android:background="@drawable/capsule_authorization"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="5dp"
        android:paddingLeft="20dp"
        android:hint="Логін"
        android:inputType="text"/>

    <EditText
        android:id="@+id/et_password"
        android:background="@drawable/capsule_authorization"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="5dp"
        android:paddingLeft="20dp"
        android:hint="Пароль"
        android:inputType="textPassword"/>

    <Button
        android:id="@+id/btn_authorization"
        android:background="@drawable/capsule_authorization"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="20dp"
        android:layout_gravity="center"
        android:text="Увійти"/>

</LinearLayout>
```

Рисунок 3.2 – Сторінка авторизації у текстовому редакторі

На рисунку 3.2 представлено вигляд сторінки авторизації у текстовому редакторі, а на рисунку 3.3 представлено у дизайнері. Вигляд сторінки у мобільному додатку наведено на рисунку 3.16.

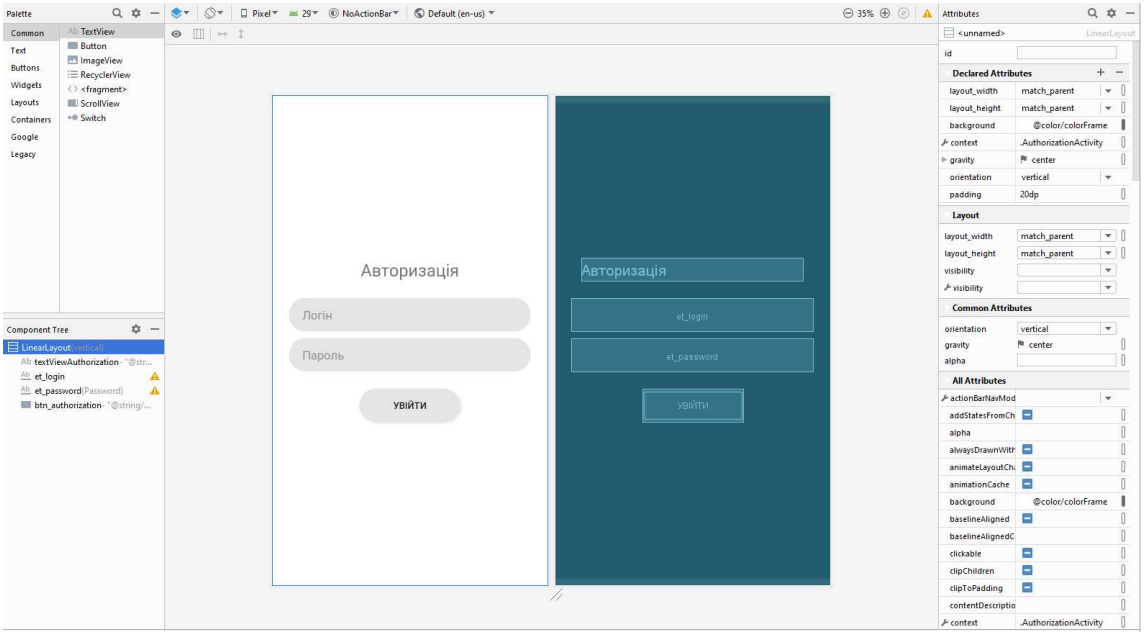

Рисунок 3.3 – Сторінка авторизації у дизайнері

Після створення layout файлу було створено відповідний java клас що описує основний функціонал полів вводу та натиснення на кнопку. Код java класу авторизації наведено в додатку В.

Розробка модуля навігаційного меню

Після створення модуля авторизації було розпочато розробку модуля навігації. За допомогою layout файлу було розроблено макет сторінки, та додано такі пункти меню: історія замірів, поточний замір, налаштування графіків та вихід.

На рисунку 3.5 представлено вигляд сторінки навігації у текстовому редакторі, а на рисунку 3.6 представлено у дизайнері. Вигляд сторінки у мобільному додатку наведено на рисунку 3.26.

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    tools:showIn="navigation_view">

    <group android:checkableBehavior="single">
        <item
            android:id="@+id/nav_history"
            android:checked="true"
            android:icon="@drawable/ic_history_experiment"
            android:title="@string/menu_history" />
        <item
            android:id="@+id/nav_now"
            android:icon="@drawable/ic_now_experiment"
            android:title="@string/menu_now" />
        <item
            android:id="@+id/nav_settings"
            android:icon="@drawable/ic_settings_chart"
            android:title="@string/menu_settings" />
    </group>

    <item android:title="Система">
        <menu>
            <item
                android:id="@+id/nav_out"
                android:icon="@android:drawable/ic_lock_power_off"
                android:title="@string/menu_out" />
        </menu>
    </item>

</menu>
```

Рисунок 3.4 – Сторінка навігації у текстовому редакторі

Рисунок 3.5 – Сторінка навігації у дизайнері

Після створення layout файлу було створено відповідний java клас що описує основний функціонал натиснення на пункти меню та подальшого відображення відповідних фрагментів. Код java класу меню навігації наведено в додатку В.

Розробка модуля історії замірів

Наступним етапом розробки було створення модуля історії замірів, який складається з двох layout файлів. За допомогою першого layout файлу було розроблено макет сторінки, в якому знаходиться список замірів, після вибору необхідного заміру відображається другий layout файл, який містить детальний опис обраного заміру, та графік [18] залежності струму від напруги.

На рисунку 3.6 представлено вигляд першого layout файлу у текстовому редакторі, а на рисунку 3.7 представлено у дизайнері. Також, на рисунку 3.8 представлено вигляд другого layout файлу у текстовому редакторі, а на рисунку 3.9 представлено у дизайнері. Вигляд відповідних сторінок у мобільному додатку наведено на рисунку 3.17 та 3.18.

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout 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:background="@color/colorFrame"
    tools:context=".HistoryFragment">

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/rv_history"
        android:layout_width="match_parent"
        android:layout_height="match_parent">

    </androidx.recyclerview.widget.RecyclerView>

</FrameLayout>
```

Рисунок 3.6 – Список історії замірів у текстовому редакторі

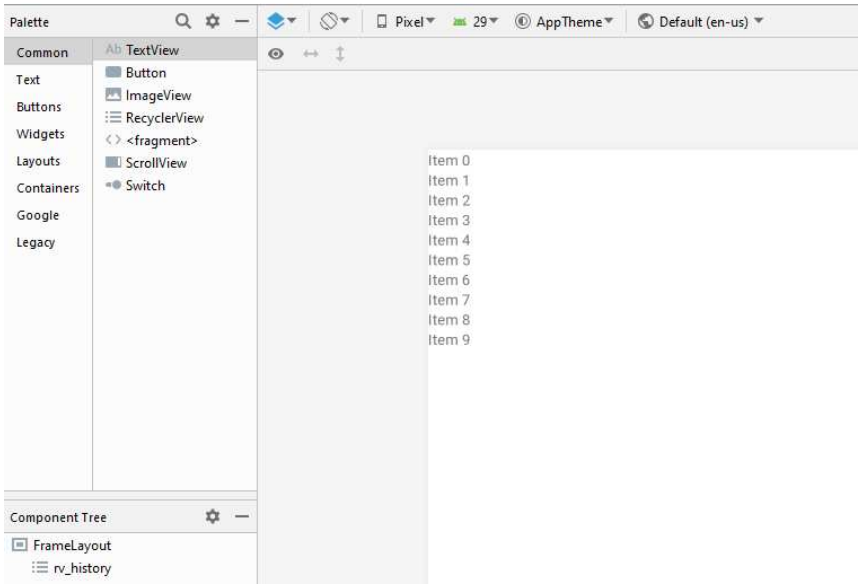

Рисунок 3.7 – Список історії замірів у дизайнері

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:background="@color/colorFrame"
    android:orientation="vertical"
    tools:context=".ClickHistoryRecyclerFragment">

    <TextView
        android:id="@+id/textDB"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="top|center"
        android:layout_marginTop="10dp"
        android:layout_marginLeft="10dp"
        android:layout_marginRight="10dp"
        android:text="База даних"/>

    <com.github.mikephil.charting.charts.LineChart
        android:id="@+id/linechart_history"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_margin="10dp"/>

</LinearLayout>
```

Рисунок 3.8 – Сторінка детального перегляду у текстовому редакторі

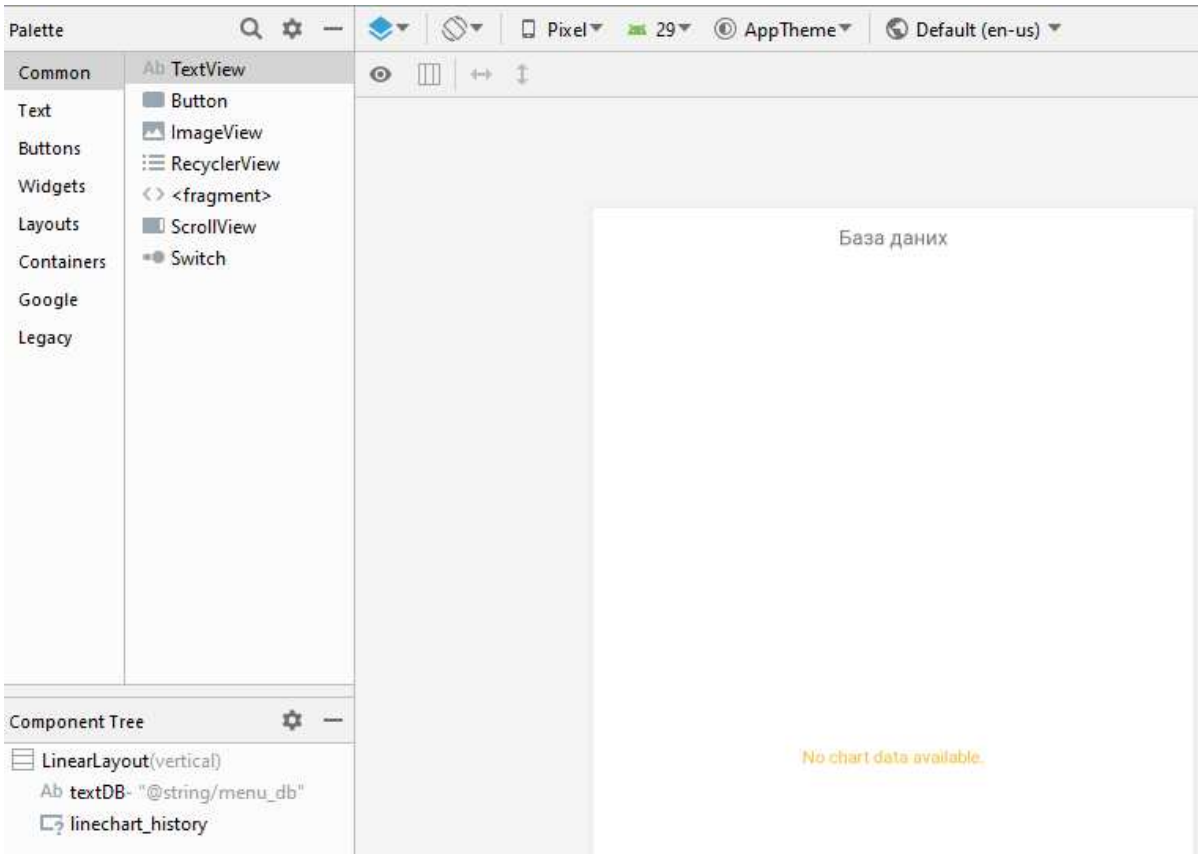

Рисунок 3.9 – Сторінка детального перегляду у дизайнері

Після створення layout файлів було створено відповідні java класи що описують основний функціонал перегляду списку замірів та їх детального опису.

Код java класів списку історії замірів та детального опису заміру наведено в додатку В.

Розробка модуля поточного заміру

Розробку модуля поточного заміру було розпочато зі створення двох layout файлів. Перший layout файл відповідає за відображення сторінки поточного заміру за його відсутності та надає змогу його розпочати, а другий, якщо замір вже було розпочато, надає детальну інформацію та побудову графіку [19] у реальному часі.

На рисунку 3.10 представлено вигляд сторінки поточного заміру за його відсутності у текстовому редакторі, а на рисунку 3.11 представлено у дизайнері. Також, на рисунку 3.12 представлено вигляд поточного заміру, який вже було розпочато у текстовому редакторі, а на рисунку 3.13 представлено у дизайнері. Вигляд відповідних сторінок у мобільному додатку наведено на рисунку 3.21 та 3.22.

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/colorFrame"
    tools:context=".NonActiveExperimentFragment">

    <TextView
        android:id="@+id/textNonActiveExperiment"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:gravity="center"
        android:text="@string/textNonActiveExperiment" />

    <com.google.android.material.floatingactionbutton.FloatingActionButton
        android:id="@+id/btnAddExperiment"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="bottom|end"
        android:src="@drawable/ic_add_black_24dp"
        android:layout_margin="20dp"
        app:backgroundTint="@color/colorPrimaryDark"
        app:fabSize="normal" />

</FrameLayout>
```

Рисунок 3.10 – Сторінка поточного заміру за відсутності у текстовому редакторі

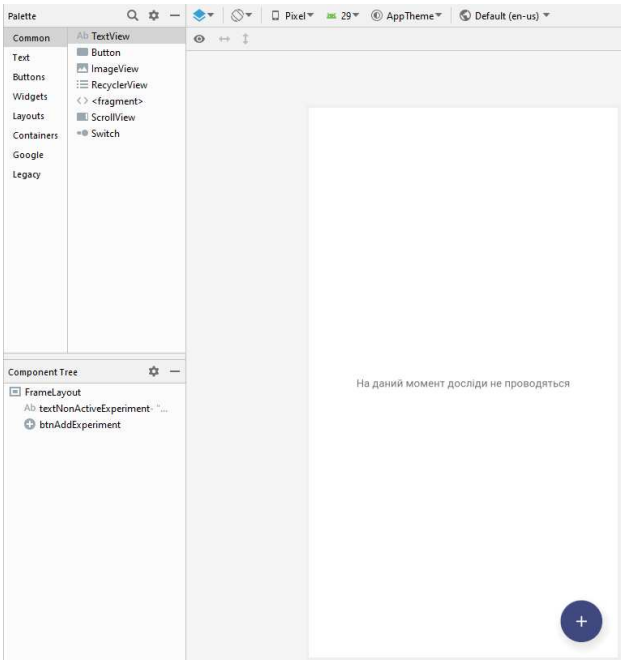

Рисунок 3.11 – Сторінка поточного заміру за його відсутності у дизайнері

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:background="@color/colorFrame"
    android:orientation="vertical"
    tools:context=".ActiveExperimentFragment">

    <TextView
        android:id="@+id/textActiveExperiment"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="top|center"
        android:layout_marginTop="10dp"
        android:layout_marginLeft="10dp"
        android:layout_marginRight="10dp"
        android:text="Значення поточного експерименту"/>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="100"
        android:orientation="horizontal">

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

            <com.github.mikephil.charting.charts.LineChart
                android:id="@+id/linechart_now"
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:layout_marginLeft="18dp"
                android:layout_marginRight="5dp"/>

            <TextView
                android:id="@+id/tv_y_axis"
                android:layout_width="wrap_content"
                android:layout_height="match_parent"
                android:layout_marginLeft="-28dp"
                android:gravity="center"
                android:rotation="-90"
                android:text="Напруга (U) " />

        </FrameLayout>

    </LinearLayout>
```

Рисунок 3.12 – Сторінка розпочатого поточного заміру у текстовому редакторі

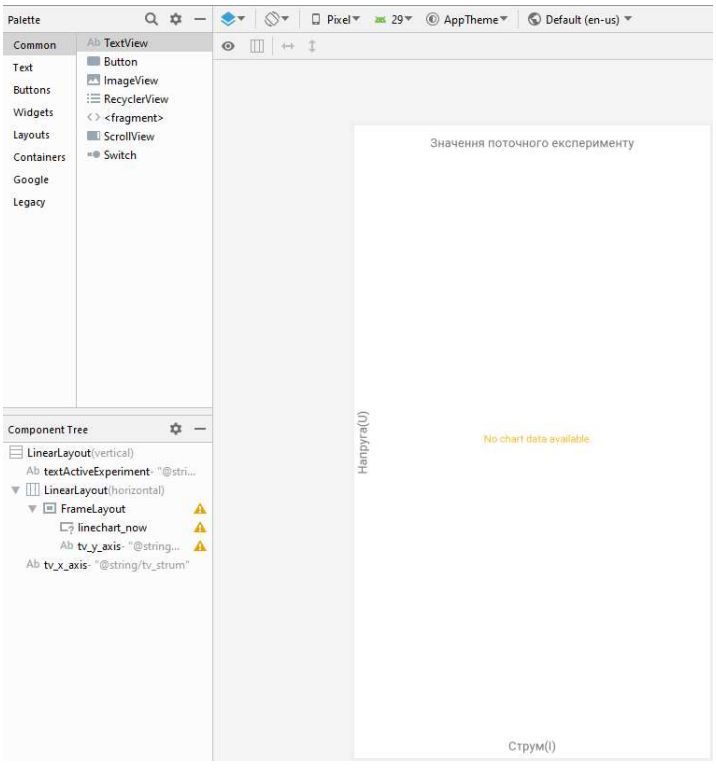

Рисунок 3.13 – Сторінка розпочатого поточного заміру у дизайнері

Після створення layout файлів було створено відповідні java класи що описують основний функціонал перегляду поточного заміру за його відсутності або за умови що замір вже було розпочато.

Код java класів поточного заміру за його відсутності та розпочатого поточного заміру наведено в додатку В.

Розробка модуля налаштувань

Останнім етапом розробки було створено модуль налаштувань. За допомогою layout файлу було розроблено макет сторінки, на якому знаходяться такі пункти налаштувань: розмір тексту, колір тексту, радіус точки, колір точки.

Збереження даних налаштувань відбувається за допомогою функції Shared Preferences [20]. Ця функція зберігає значення за допомогою класу Editor [21] та завантажує їх при наступному вході.

На рисунку 3.14 представлено вигляд сторінки налаштувань у текстовому редакторі, а на рисунку 3.15 представлено у дизайнері. Вигляд сторінки у мобільному додатку наведено на рисунку 3.25.

```
<TextView
    android:id="@+id/textSettings"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:gravity="center|top"
    android:layout_margin="20dp"
    android:textSize="20sp"
    android:text="Налаштування графіків" />

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:layout_margin="5dp">

    <TextView
        android:id="@+id/tv_sp_text_size"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="2"
        android:gravity="center"
        android:text="Розмір тексту" />

    <Spinner
        android:id="@+id/sp_set_text_size"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:spinnerMode="dialog"
        android:layout_weight="1" />

</LinearLayout>
```

Рисунок 3.14 – Сторінка налаштувань у текстовому редакторі

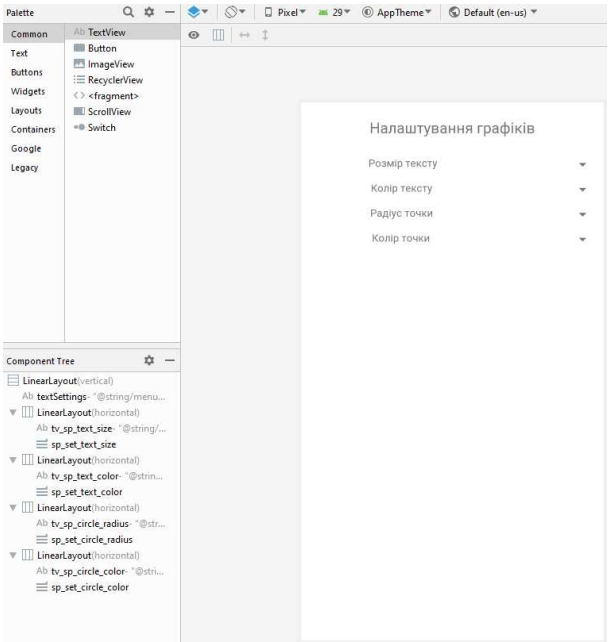

Рисунок 3.15 – Сторінка налаштувань у дизайнері

Після створення layout файлу було створено відповідний java клас що описує основний функціонал зміни налаштувань. Код java класу налаштувань наведено в додатку В.

3.3 Використання мобільного додатку

Після запуску мобільного додатку необхідно авторизуватись для доступу до основного функціоналу мобільного додатку. Вікно авторизації наведено на рисунку 3.16.

Рисунок 3.16 – Вікно авторизації

Після успішної авторизації дослідник потрапляє до сторінки перегляду списку історії замірів. Вікно історії замірів наведено на рисунку 3.17.

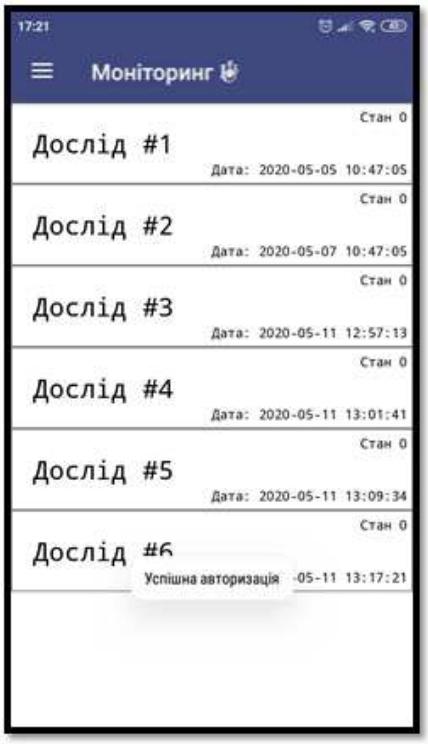

Рисунок 3.17 – Вікно історії замірів

Після вибору необхідного заміру надається опис даних, які використовувалися під час проведення експерименту та їх графічне представлення, в залежності струму від напруги на основі даних з бази. Вікно детального перегляду обраної історії заміру наведено на рисунку 3.18.

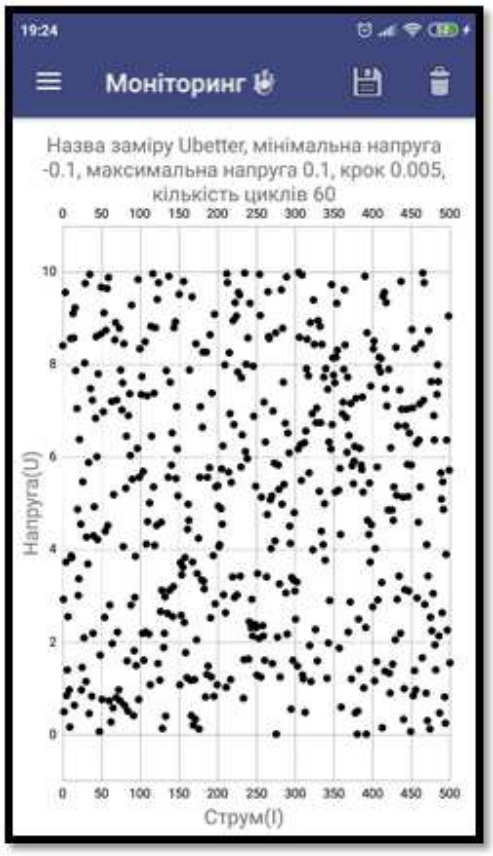

Рисунок 3.18 – Вікно детального перегляду

Під час перегляду опису історії заміру у дослідника є можливість зберегти графік у форматі png, наведено на рисунку 3.19 та видалити замір з історії, наведено на рисунку 3.20.

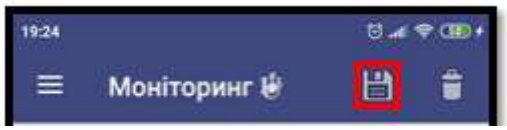

Рисунок 3.19 – Кнопка збереження графіку

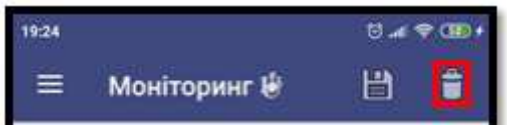

Рисунок 3.20 – Кнопка видалення заміру

Також, дослідник має змогу перейти до перегляду поточного заміру. Якщо на даний момент експеримент не проводиться, то є можливість його розпочати на основі введених початкових значеннях, наведено на рисунку 3.21 та 3.22.

Рисунок 3.21– Вікно поточного експерименту

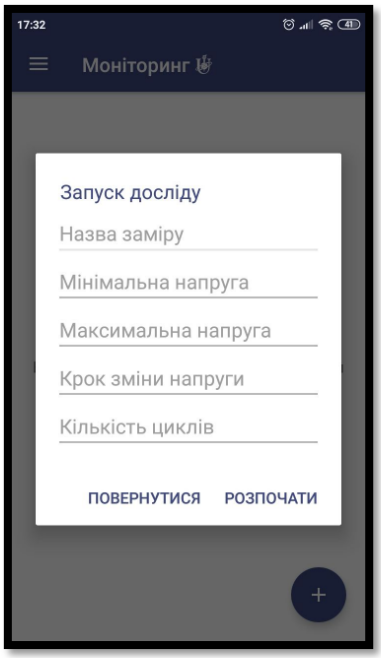

Рисунок 3.22 – Вікно запуску експерименту

У іншому випадку, якщо експеримент вже розпочатий, надається опис даних експерименту, побудова залежності струму від напруги у реальному часі на основі даних, які надходять до бази даних від експериментальної установки, наведено на рисунку 3.23 та можливість зупинити експеримент, наведено на рисунку 3.24.

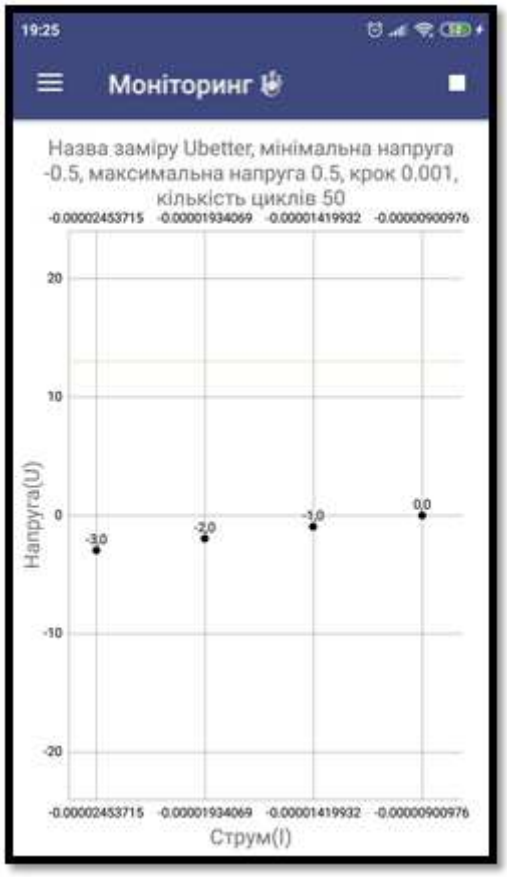

Рисунок 3.23 – Вікно поточного заміру

Рисунок 3.24 – Кнопка зупинки експерименту

Також, є можливість налаштувати основний вигляд графіку під свої потреби, змінивши розмір крапки, підпису крапки та їх колір. Вікно зміни налаштувань графіку наведено на рисунку 3.25.

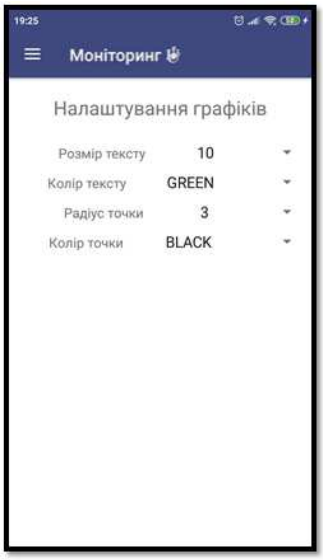

Рисунок 3.25 – Вікно налаштувань графіку

За необхідності є можливість виконати деавторизацію у системі, наведено на рисунку 3.26.

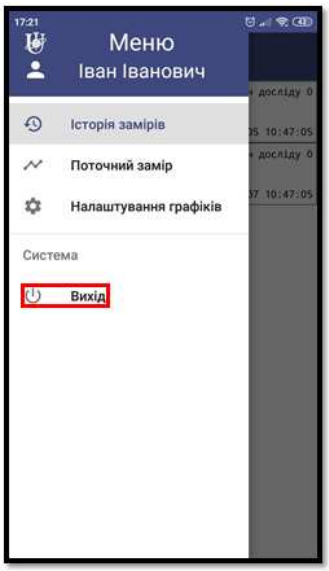

Рисунок 3.26 – Пункт деавторизації

ВИСНОВКИ

У ході виконання дипломної роботи було проведено аналіз предметної області, а саме:

– розглянуто сучасні дослідження та публікації щодо дослідження характеристик літій-іонних батарей;

– проведено пошук та аналіз програмних продуктів-аналогів та використаних інформаційних технологій для візуалізації експериментальних даних.

За результатами проведеного аналізу було визначено актуальність розробки та визначено мету роботи, яка полягає у створенні мобільного додатку моніторингу електрохімічних характеристик електродів для літій-іонних батарей, використання якого розширить можливості дослідників. Також було виділено задачі, які потрібно вирішити для реалізації поставленої мети. В ході проведеної постановки завдання було сформоване технічне завдання на розробку.

По закінченню аналізу предметної області було проведено проектування інформаційної системи, а саме:

– виконано моделювання процесу моніторингу дослідження характеристик літій-іонних батарей, представлене у формі контекстної діаграми процесу та його декомпозиції у нотації IDEF0;

– розроблено модель додатку як сукупність моделі варіантів використання (у формі діаграми варіантів використання із описом проекту інтерфейсу) та моделі аналізу (у формі сукупності діаграм послідовності варіантів використання);

– визначена схема даних, з якими повинен працювати додаток.

Також, було виконано проектування мобільного додатку, наведено архітектуру та описано етапи програмної реалізації.

Для організації проекту було проведено планування робіт за допомогою WBS та OBS, розроблено календарний план та виконано аналіз ризиків проекту.

Результати дослідження апробовано на конференції IMA 2020 (Додаток Д).

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Методология тестирования литий-ионных аккумуляторов [Електронний ресурс] // Finance. – 2018. – Режим доступу до ресурсу: https://news.finance.ua/ru/news/-/421013/skorost-zaryadki-litij-ionnyh-batarej-mozhno-uvelichit-v-5-raz (дата звернення: 15.04.20);

2. Serdechnyy D. V. THE CONTROL OF A MULTI-ELEMENT CHARGE LITHIUM-ION BATTERIES [Електронний ресурс] / D. V. Serdechnyy, Y. B. Tomashevskiy // Saratov State Technical University. – 2017. – Режим доступу до ресурсу: https://imuk.pnzgu.ru/files/imuk.pnzgu.ru/16317.pdf (дата звернення: 17.04.20);

3. Battery Charging Monitor - Ampere Meter [Електронний ресурс] // GooglePlay. – 2019. – Режим доступу до ресурсу: https://play.google.com/store/-apps/details?id=com.tofabd.batteryanalyzer&hl=ru (дата звернення: 20.04.20);

4. GSam Battery Monitor [Електронний ресурс] // GooglePlay. – 2013. – Режим доступу до ресурсу: https://play.google.com/store/apps/details?id=com.gsamlabs.bbm (дата звернення: 20.04.20);

5. Battery Monitor [Електронний ресурс] // GooglePlay. – 2015. – Режим доступу до ресурсу: https://play.google.com/store/apps/details?id=com.glgjing.hulk (дата звернення: 20.04.20);

6. Allison L. Step-to-Step Guide - How to Create IDEF0 Diagram [Електронний ресурс] / Lynch Allison // edraw. – 2019. – Режим доступу до ресурсу: https://www.edrawsoft.com/how-to-create-idef0-diagram.html (дата звернення: 22.04.20);

7. UML Tutorial [Електронний ресурс] // tutorialspoint. – 2016. – Режим доступу до ресурсу: https://www.tutorialspoint.com/uml/index.htm (дата звернення: 23.04.20);

8. Use Case Diagram [Електронний ресурс] // visual paradigm. – 2018. – Режим доступу до ресурсу: https://online.visual-paradigm.com/diagrams/tutorials/use-case-diagram-tutorial/ (дата звернення: 23.04.20);

9. MySQL Database Diagram [Електронний ресурс] // devart. – 2018. – Режим доступу до ресурсу: https://www.devart.com/dbforge/mysql/studio/database-designer.html (дата звернення: 25.04.20);

10. ER Diagram [Електронний ресурс] // Creately. – 2019. – Режим доступу до ресурсу: https://creately.com/blog/diagrams/er-diagrams-tutorial/ (дата звернення: 25.04.20);

11. MySQL Primary Key [Електронний ресурс] // MySQ. – 2018. – Режим доступу до ресурсу: https://www.mysqltutorial.org/mysql-primary-key/ (дата звернення: 25.04.20);

12. Using Foreign Keys [Електронний ресурс] // MySQL. – 2018. – Режим доступу до ресурсу: https://dev.mysql.com/doc/refman/8.0/en/example-foreign-keys.html (дата звернення: 25.04.20);

13. Anupam C. Android MVVM Design Pattern [Електронний ресурс] / Chugh Anupam // JournalDev. – 2020. – Режим доступу до ресурсу: https://www.journaldev.com/20292/android-mvvm-design-pattern (дата звернення: 01.05.20);

14. Muntenescu F. Android Architecture MVC [Електронний ресурс] / Florina Muntenescu // upday devs. – 2016. – Режим доступу до ресурсу: https://medium.com/upday-devs/android-architecture-patterns-part-1-model-view-control-3baecef5f2b6 (дата звернення: 01.05.20);

15. Layout [Електронний ресурс] // tutorialspoint. – 2018. – Режим доступу до ресурсу: https://www.tutorialspoint.com/android/android_user_interface_layouts.htm (дата звернення: 03.05.20);

16. Introduction to Activities [Електронний ресурс] // developers. – 2019. – Режим доступу до ресурсу: https://developer.android.com/guide/components-/activities/intro-activities (дата звернення: 03.05.20);

17. Android - Fragments [Електронний ресурс] // tutorialspoint. – 2019. – Режим доступу до ресурсу: https://www.tutorialspoint.com/android/android_fragments.htm (дата звернення: 03.05.20);

18. Jay P. MPAndroidChart [Електронний ресурс] / Phil Jay // GitHub. – 2016. – Режим доступу до ресурсу: https://github.com/PhilJay/MPAndroidChart (дата звернення: 05.05.20);

19. Janson J. Android Chart Example [Електронний ресурс] / Joe Janson // javapapers. – 2018. – Режим доступу до ресурсу: https://javapapers.com/android/android-chart-example-app-using-mpandroidchart/ (дата звернення: 05.05.20);

20. Климов А. SharedPreferences [Електронний ресурс] / Александр Климов // alexanderklimov. – 2018. – Режим доступу до ресурсу: http://developer.alexanderklimov.ru/android/theory/sharedpreferences.php (дата звернення: 10.05.20);

21. Editor [Електронний ресурс] // developers. – 2017. – Режим доступу до ресурсу: https://developer.android.com/reference/android/content/SharedPreferences.Editor (дата звернення: 10.05.20).

ДОДАТОК А

Технічне завдання

1 Призначення й мета створення мобільного додатку

1.1 Призначення мобільного додатку

Для покращення роботи літій-іонних батарей потрібно виконувати обробку та аналіз великих обсягів експериментальних даних, щодо потреб візуального представлення цих даних. Графічне представлення цих даних дозволить істотно зменшити час проведення аналізу. Мобільні пристрої є найбільш розповсюдженими обчислювальними пристроями, тому використання мобільного додатку розширить можливості дослідників.

1.2 Мета створення мобільного додатку

Мета – розроблення мобільного додатку моніторингу характеристик літій-іонних батарей.

1.3 Цільова аудиторія

Доступ до мобільного додатку матимуть авторизовані користувачі, так як це система із закритим доступом.

2 Вимоги до мобільного додатку

2.1 Загальні вимоги

– робота мобільного додатку на платформі Android 4.0 та вище;

– підтримка роботи в довільній орієнтації екрану;

– адаптація до різних розширень екрану для коректного відображення даних.

2.2 Потреби користувача

Потреби користувача представлені в таблиці А.1.

Таблиця А.1 – Потреби користувача

ID	Потреби користувача	Джерело
UN-01	Побудова графіку у реальному часі	Співробітник
UN-02	Збереження графіку	Співробітник
UN-03	Перегляд історії замірів	Співробітник
UN-04	Видалення історії певного заміру	Співробітник
UN-05	Налаштування графіку	Співробітник
UN-06	Запуск та зупинка експерименту	Співробітник

2.3 Системні вимоги

Системні вимоги представлені в таблиці А.2.

Таблиця А.2 – Системні вимоги

ID	Системні вимоги	Пріоритет	Опис
SR-01	Сторінка поточного заміру	M	Перегляд результатів заміру у реальному часі
SR-02	Наявність модуля збереження обраного графіку	M	Надає можливість зберегти обраний з історії графік у форматі png

Продовження таблиці А.2 – Системні вимоги

ID	Системні вимоги	Пріоритет	Опис
SR-03	Сторінка історії замірів	M	Перегляд архівних замірів які були зроблені раніше
SR-04	Наявність модуля видалення обраного заміру	M	Надає можливість видалення обраного заміру з історії
SR-05	Сторінка налаштувань	M	Налаштування кольору і розміру точок та надписів графіку
SR-06	Наявність модуля запуску та зупинки експерименту	M	Надає можливість зупинити експеримент або розпочати за його відсутності

2.4 Мова реалізації

– українська.

2.5 Мова програмування

– java.

2.6 Основні функції мобільного додатку

– побудова графіку за вхідними даними поточного циклу;

– збереження графіку у вигляді png формату;

– перегляд історії замірів;

– можливість видалення певної історії заміру;

– налаштування зовнішнього вигляду графіку: кольору та розміру точок та тексту;

– можливість запуску та зупинки експерименту із візуалізацією результатів замірів.

2.7 Вхідні дані для роботи мобільного додатку

Показники для роботи мобільного додатку надходять з бази даних, а саме:

– для побудови графіків використовуються значення напруги та струму;

– для початку експерименту вводиться номер досліду, мінімальна напруга, максимальна напруга, крок зміни напруги та кількість циклів.

3 Макети сторінок мобільного додатку

Рисунок А.1 – Макет головної сторінки

Рисунок А.2 – Макет меню головної сторінки

Рисунок А.3 – Макет сторінки авторизації

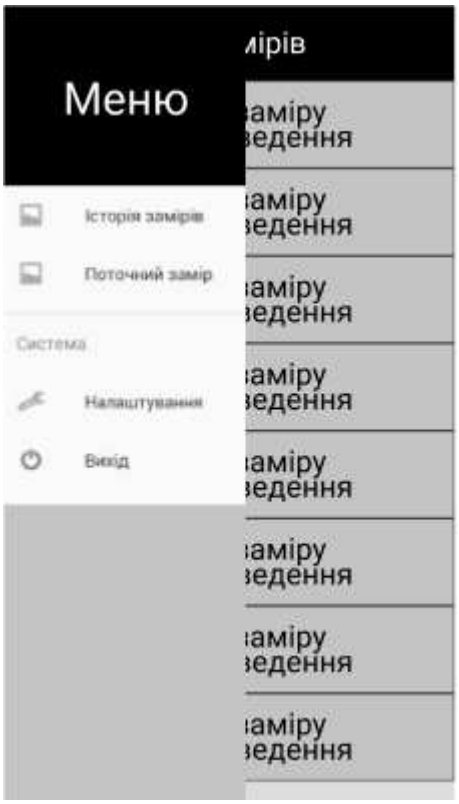

Рисунок А.4 – Макет меню сторінки після успішної авторизації

Рисунок А.5 – Макет сторінки історія замірів

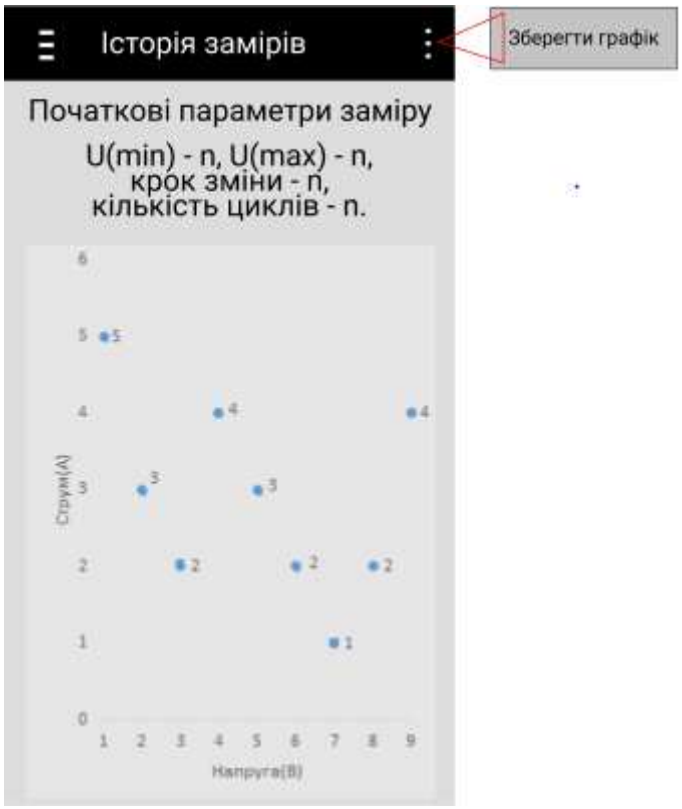

Рисунок А.6 – Макет перегляду обраного заміру із історія замірів

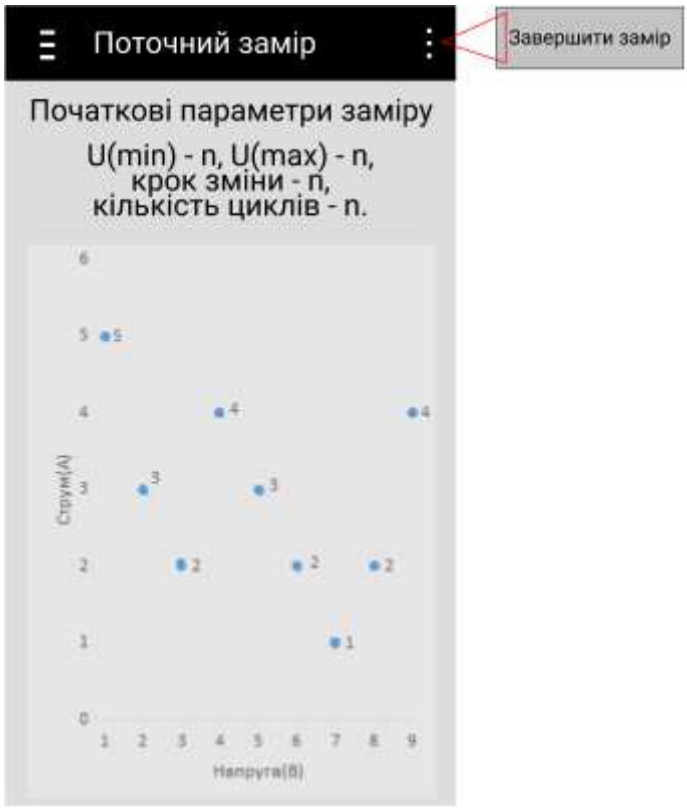

Рисунок А.7 – Макет сторінки з поточними замірами під час проведення досліду

Рисунок А.8 – Макет сторінки з поточними замірами під час відсутності досліду

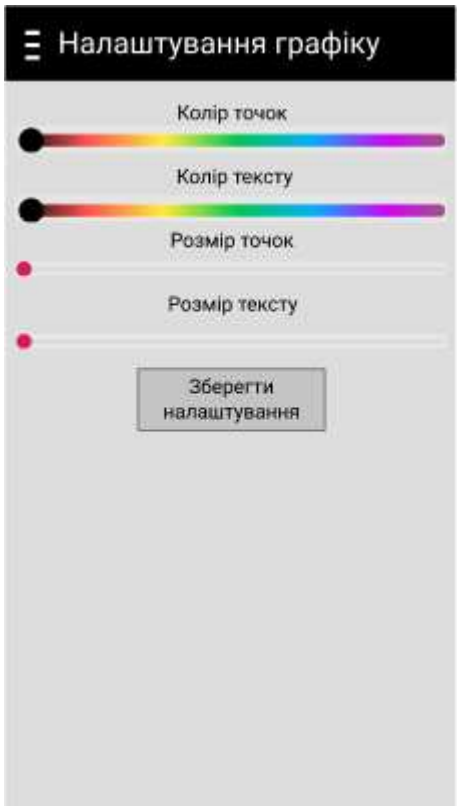

Рисунок А.9 – Макет налаштування графіку

4 Перелік програмної документації

– технічне завдання (даний документ);
– інструкція користувача.

5 Порядок виконання робіт проекту та етапи розробки

Таблиця А.3 – Етапи розробки проекту

№	Склад робіт	Строки виконання
1.	Розроблення та затвердження технічного завдання	5 днів
2.	Аналіз систем-аналогів на основі дослідження предметної області	10 днів
3.	Вибір та налаштування інструментів реалізації	5 днів
4.	Макетування користувацького інтерфейсу	5 днів
5.	Проектування алгоритмів візуалізації даних та їх програмування	15 днів
6.	Реалізація додатку та його тестування	5 днів
7.	Розроблення інструкції користувача	3 дні
8.	Виправлення можливих помилок виявлених під час тестування	2 дні
Всього		50 днів

6 Загальні вимоги до приймання мобільного додатку

Приймання роботи здійснюється у строки дипломного проектування із обов’язковим представленням виконавцем мобільного додатку та супровідної документації. Після чого замовником проводиться тестування та затвердження роботи.

7 Введення мобільного додатку в експлуатацію

Експлуатація мобільного додатку виконується за допомогою *.apk файлу мобільного додатку, який повинен надати виконавець. Після чого замовник здійснює публікацію додатку на GooglePlay або AppStore.

ДОДАТОК Б

Планування робіт

Деталізація мети методом SMART.

Мета проекту полягає у розробці мобільного додатку моніторингу електрохімічних характеристик електродів для літій-іонних батарей.

Результати деталізації мети методом SMART розміщені у таблиці Б.1

Таблиця Б.1 – Деталізація мети проекту методом SMART

Specific	Створити мобільний додаток моніторингу електрохімічних характеристик електродів для літій-іонних батарей
Measurable	Здати мобільний додаток замовнику за результатами приймальних іспитів продукту та отримати оцінку роботи над проектом
Achievable	Створити мобільний додаток, який виконує визначені у ТЗ функції, встановити середовище розробки Android Studio, пройти курс програмування Android
Relevant	Мати достатні навички для розробки та необхідне програмне забезпечення
Time-framed	Дотримуватися розробленого календарного плану

Планування змісту структури робіт (WBS).

Ієрархічна структура проекту представлена як декомпозиція робіт WBS (work breakdown structure) на рисунку Б.1. На верхньому рівні ієрархічної структури проекту зафіксовано продукт проекту, після чого показані задачі проекту та підзадачі. Декомпозиція робіт проекту була закінчена після того, як для кожної під задачі був визначений фахівець, який розуміє технологічні особливості їх вирішення. Закріплення фахівців за підзадачами наведено в організаційній структурі проекту на рисунку Б.2.

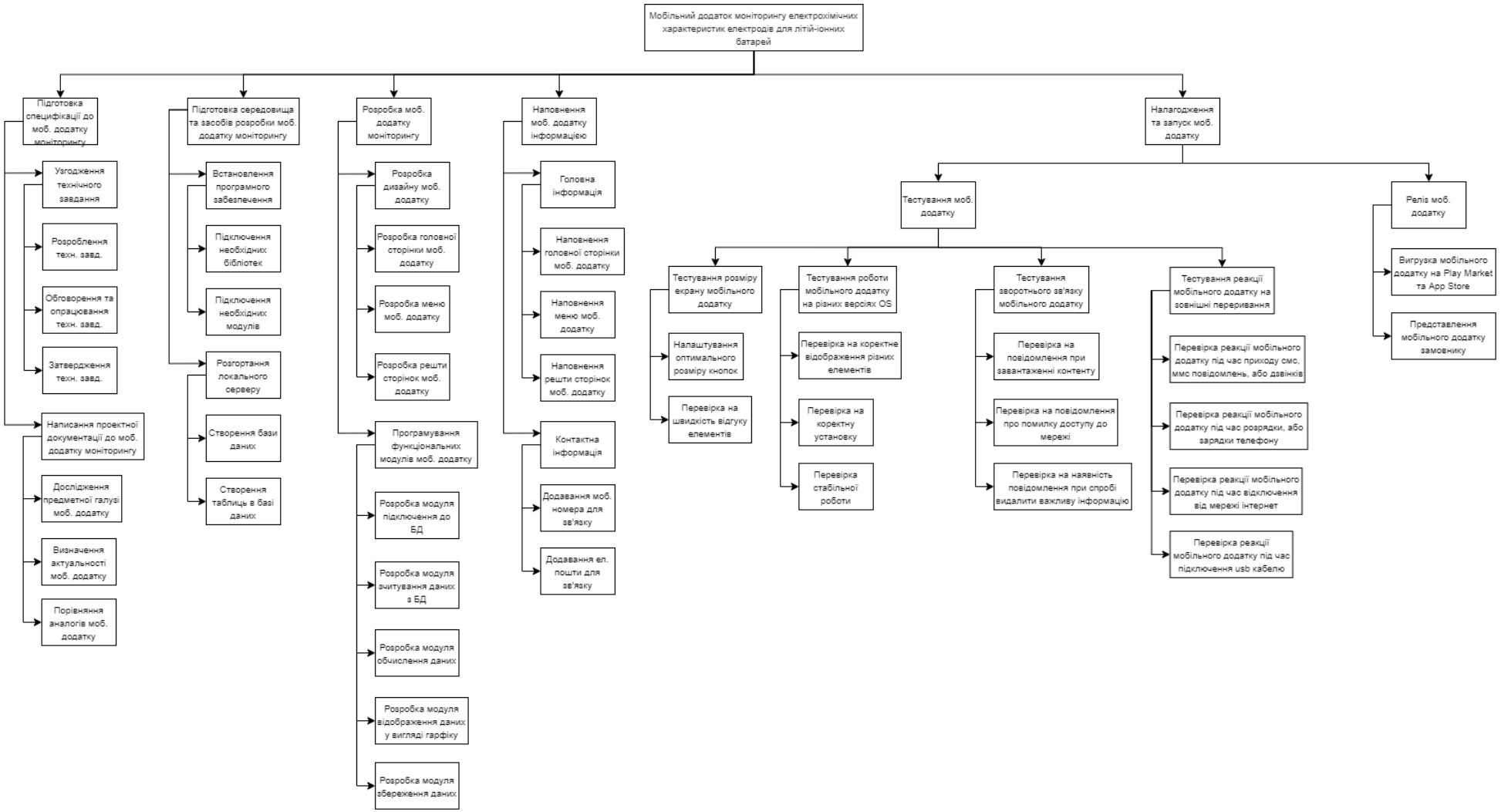

Рисунок Б.1 – Ієрархічна структура проекту

Планування структури організації (OBS).

Відображення учасників проекту та відповідальних осіб, які задіяні в реалізації проекту показане у організаційній структурі проекту OBS (organization breakdown structure). На верхньому рівні організаційної структури проекту розташована команда проекту, нижче визначені виконавці до кожної задачі та підзадачі елементарних робіт ієрархічної структури проекту.

На рисунку Б.2 наведено організаційну структуру проекту.

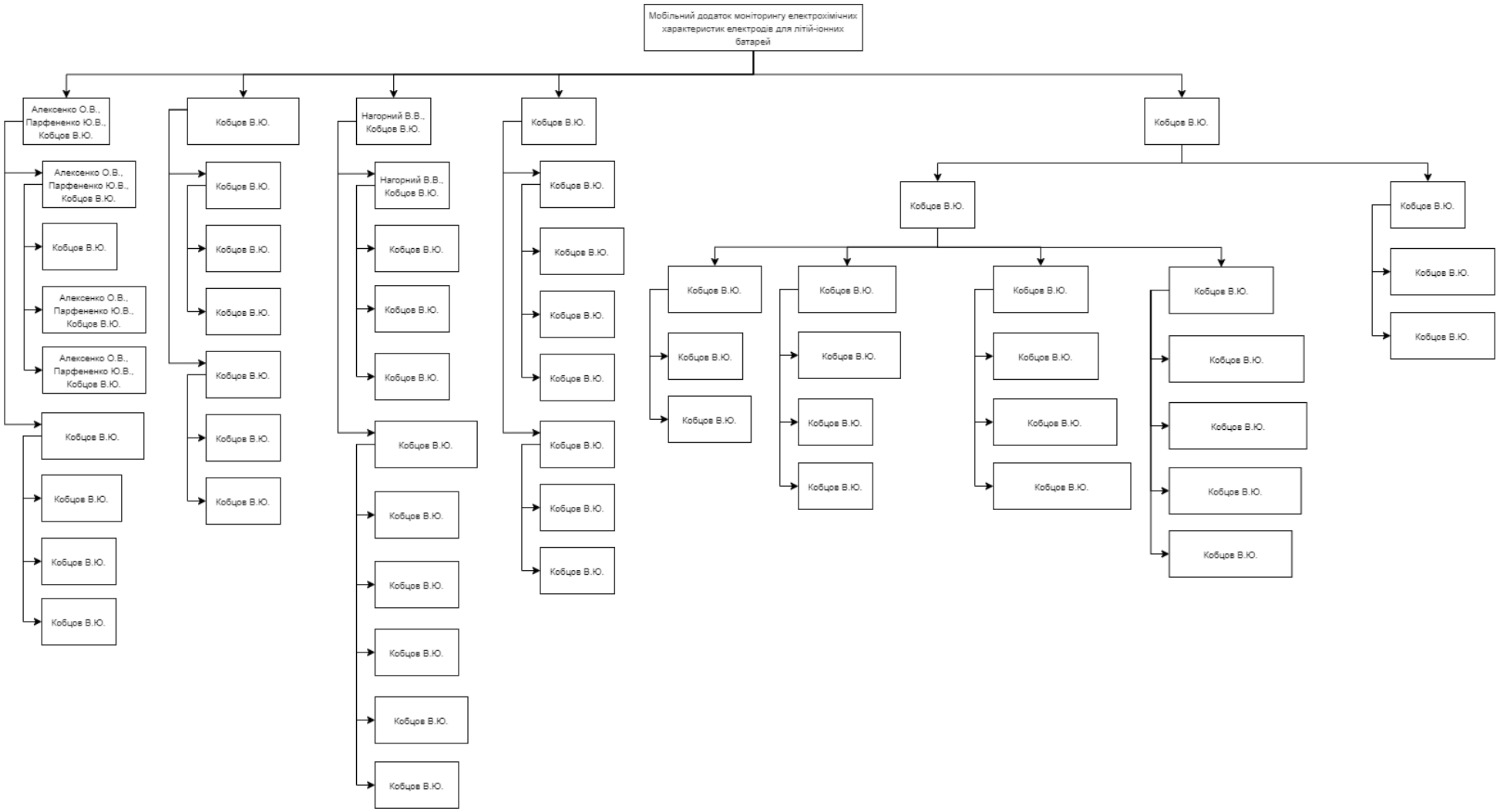

Рисунок Б.2 – Організаційна структура проекту

Побудова календарного плану.

Відображення календарного плану що складається зі списку цілей та задач, які виконуються під час роботи над проектом та діаграми Ґанта, яка відображає задачі та відсоток їх виконання наведено на рисунку Б.3.

Название задачи	Длительн	Начало	Окончание
Розробка мобільного додатку моніторнигу характеристик літій-іонних батарей при проведенні експериментів з їх зарядки-розрядки	99 дней	Вт 14.01.20	Пт 29.05.20
Підготовка специфікації до ІС моніторингу	15 дней	Вт 14.01.20	Пн 03.02.20
Узгодження технічного завдання	5 дней	Вт 14.01.20	Пн 20.01.20
Написання проектної документації до ІС моніторингу	10 дней	Вт 21.01.20	Пн 03.02.20
Підготовка середовища та засобів розробки ІС моніторингу	5 дней	Вт 04.02.20	Пн 10.02.20
Встановлення програмного забезпечення	2 дней	Вт 04.02.20	Ср 05.02.20
Розгортання локального серверу	3 дней	Чт 06.02.20	Пн 10.02.20
Розробка моб. додатку ІС моніторингу	50 дней	Вт 11.02.20	Пн 20.04.20
Розробка дизайну моб. додатку	10 дней	Вт 11.02.20	Пн 24.02.20
Програмування функціональних модулів моб. додатку	40 дней	Вт 25.02.20	Пн 20.04.20
Наповнення моб. додатку інформацією	15 дней	Вт 21.04.20	Пн 11.05.20
Головна інформація	13 дней	Вт 21.04.20	Чт 07.05.20
Контактна інформація	2 дней	Пт 08.05.20	Пн 11.05.20
Налагодження та запуск моб. додатку	14 дней	Вт 12.05.20	Пт 29.05.20
Тестування моб. додатку	12 дней	Вт 12.05.20	Ср 27.05.20
Реліз моб. додатку	2 дней	Чт 28.05.20	Пт 29.05.20

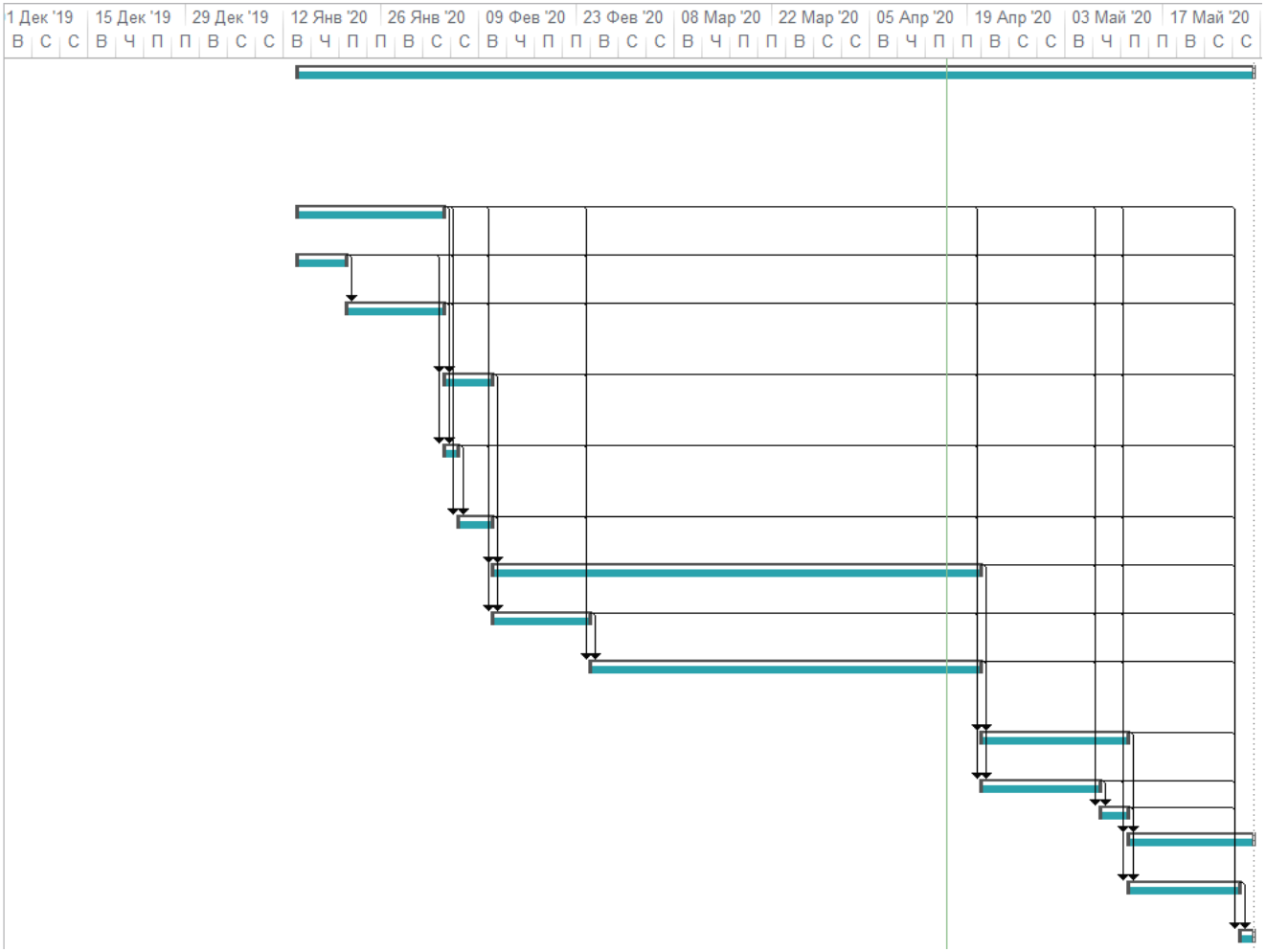

Рисунок Б.3 – Діаграма Ґанта

Управління ризиками проекту.

Управління ризиками – це моніторинг ризиків та реагування на їх зміни під час виконання проекту.

Ризики та відповідні до них описи представлені в таблиці Б.2-Б.5.

Таблиця Б.2 – Ризики

№	Назва ризику	Опис ризику
1	Оцінка термінів	Характерні помилки в визначення термінів необхідних для реалізації проекту. Часто це пов'язано з недостатністю опрацювання календарного плану проекту, що призводить до появи нових(необхідних) робіт та зміщення термінів.
2	Хостинг	Збій у роботі хостингу.
3	Кваліфікація	Недостатня кваліфікація розробника.
4	Інтеграція	Виникнення різних проблем у процесі інтеграції нового модулю до мобільного додатку.
5	Технічні	Відмова устаткування, його поломка можуть вплинути на терміни здійснення проекту, частково призупинити роботу над проектом, до відновлення несправності.
6	Невизначеність	Замовник, як правило, усвідомлює тільки мету, яку хоче досягти, але не має уявлення про процес і способи реалізації проекту. Замовник і розробник розмовляють різними мовами, і одна з основних задач правильно зрозуміти вимоги замовника. На етапі ініціації проекту і підготовки технічного завдання, необхідно чітко визначити всі специфікації продукту і яким чином вони повинні бути реалізовані. Крім того, під час реалізації мобільного доданку замовник може внести зміни в специфікації. Часта зміна вимог призводить до порушення графіка проекту і збільшення його вартості.

Продовження таблиці Б.2 – Ризики

№	Назва ризику	Опис ризику
7	Взаємозв’язок	Відсутність взаємодії з замовником може привести до різноманітних проблем. На завершальних стадіях проекту призводить до можливого виявлення нових вимог. Ці вимоги можуть виникнути при підготовці і проведенні приймальних випробувань додатку. Дана ситуація здатна чинити серйозний вплив на терміни реалізації проекту.
8	Технології	Використання для реалізації проекту нових, ще не випробуваних технологій може привести до ускладнень в реалізації проекту.

Таблиця Б.3 – Аналіз ризиків

№ ризику	Ймовірність виникнення	Ступінь впливу	Важливість ризику(оцінка)	Значення ризику(грн.)	Пом’якшення ризику
1	1	4	4	0	Час
2	2	3	6	300	Час та гроші
3	4	5	20	500	Час та гроші
4	3	3	9	0	Час
5	3	4	12	0	Час
6	2	4	8	700	Час та гроші
7	4	4	16	0	Час
8	2	2	4	800	Час та гроші

Таблиця Б.4 – Матриця впливу

5					
4				7	3
3			4	5	

Продовження таблиці Б.4 – Матриця впливу

2		8	2	6	
1				1	
0	1	2	3	4	5

Оцінювання:

1 – Дуже низький

2 – Низький

3 – Середній

4 – Високий

5 – Дуже високий

Таблиця Б.5 – План вирішення ризиків

№ ризику	Вирішення
1	Розробити WBS в якій зробити детальну декомпозицію робіт.
2	Дочекатися поки хостинг знову запрацює, або перейти на новий.
3	Повисити кваліфікацію за допомогою інтернет ресурсів, або відповідним курсам.
4	Проводити розробку мобільного додатку на крос-платформній мові програмування.
5	Дочекатися відновлення несправності, або на час відновлення несправності перейти до іншої роботи.
6	Обговорити всі нюанси які можуть виникнути під час розробки мобільного додатку, узгодити технічне завдання та при виникненні питань посилатися на вже узгоджене технічне завдання.
7	За можливості підтримувати постійний зв’язок з замовником.
8	Для запобігання можливих проблем з використанням нових технологій в графік проекту необхідно закладати час на можливе вивчення нової технології.

ДОДАТОК В

Лістинг програмного коду

Java клас авторизації

```
public class AuthorizationActivity extends AppCompatActivity {

    private Button btnAuthorization;
    private EditText etLogin;
    private EditText etPassword;
    private SharedPreferences sPref;

    private String login;

    final String AUTHORIZATION = "0";
    final String USERNAME = "";

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

        btnAuthorization = findViewById(R.id.btn_authorization);
        etLogin = findViewById(R.id.et_login);
        etPassword = findViewById(R.id.et_password);

        if(SaveStaticValue.saveOutput == 1){
            saveSettings();
        }

        loadSettings();

        //Toast.makeText(AuthorizationActivity.this,        "Static="        +
SaveStaticValue.checkAuthorization, Toast.LENGTH_SHORT).show();

        ConnectivityManager conManager = (ConnectivityManager)

getApplicationContext().getSystemService(Context.CONNECTIVITY_SERVICE);

        NetworkInfo activeNetwork = conManager.getActiveNetworkInfo();

        if(null != activeNetwork){
            if(activeNetwork.getType() == ConnectivityManager.TYPE_WIFI){
                //Toast.makeText(AuthorizationActivity.this,   "Підключено   до
Wifi", Toast.LENGTH_SHORT).show();
            }
            if(activeNetwork.getType() == ConnectivityManager.TYPE_MOBILE){
                //Toast.makeText(AuthorizationActivity.this,   "Підключено   до
Data Network", Toast.LENGTH_SHORT).show();
```

```
                }

                if(SaveStaticValue.checkAuthorization == 1){
                    Toast.makeText(AuthorizationActivity.this,   "Авторизація   вже
була виконана", Toast.LENGTH_SHORT).show();
                    startActivity(new               Intent(getApplicationContext(),
MainActivity.class));
                    finish();
                }
                else if(SaveStaticValue.checkAuthorization == 0){
                    Toast.makeText(AuthorizationActivity.this,   "Для   доступу   до
моніторингу необхідно виконати авторизацію", Toast.LENGTH_SHORT).show();
                }
            }
            else {
                Toast.makeText(AuthorizationActivity.this,    "Відсутнє    інтернет
підключення, авторизація неможлива", Toast.LENGTH_SHORT).show();
            }

            btnAuthorization.setOnClickListener(new View.OnClickListener() {
                @Override
                public void onClick(View v) {
                    ConnectivityManager conManager = (ConnectivityManager)

getApplicationContext().getSystemService(Context.CONNECTIVITY_SERVICE);

                    NetworkInfo activeNetwork = conManager.getActiveNetworkInfo();

                    if(null != activeNetwork){
                        if(activeNetwork.getType()                                  ==
ConnectivityManager.TYPE_WIFI){
                            //Toast.makeText(AuthorizationActivity.this,
"Підключено до Wifi", Toast.LENGTH_SHORT).show();
                        }
                        if(activeNetwork.getType()                                  ==
ConnectivityManager.TYPE_MOBILE){
                            //Toast.makeText(AuthorizationActivity.this,
"Підключено до Data Network", Toast.LENGTH_SHORT).show();
                        }

                        login();
                    }
                    else {
                        Toast.makeText(AuthorizationActivity.this,        "Відсутнє
інтернет підключення", Toast.LENGTH_SHORT).show();
                    }
                }
            });
        }

        public void login(){
            StringRequest   request   =   new   StringRequest(Request.Method.POST,
"http://batteriestests.tk/QueryApp/userInfo.php", new Response.Listener<String>() {
                @Override
                public void onResponse(String response) {
                    if(response.contains("1")){

                        login = etLogin.getText().toString();
                        SaveStaticValue.userName = login;

                        SaveStaticValue.checkAuthorization = 1;
                        Toast.makeText(AuthorizationActivity.this,         "Успішна
авторизація", Toast.LENGTH_SHORT).show();
```

```
                        Intent intent = new Intent(getApplicationContext(),
MainActivity.class);
                        intent.putExtra("indexAuthorization", "1");
                        startActivity(intent);
                        finish();
                    }
                    else{
                        Toast.makeText(AuthorizationActivity.this,         "Невірно
введений логін або пароль", Toast.LENGTH_SHORT).show();
                    }
                }
            }, new Response.ErrorListener() {
                @Override
                public void onErrorResponse(VolleyError error) {

                }
            }){
                @Override
                protected Map<String, String> getParams() throws AuthFailureError {

                    Map<String, String> params = new HashMap<>();
                    params.put("username", etLogin.getText().toString());
                    params.put("password", etPassword.getText().toString());
                    return params;
                }
            };

            Volley.newRequestQueue(this).add(request);
        }

        private void saveSettings(){
            sPref = getPreferences(MODE_PRIVATE);
            SharedPreferences.Editor editor = sPref.edit();
            editor.putString(AUTHORIZATION,
String.valueOf(SaveStaticValue.checkAuthorization));
            editor.putString(USERNAME, String.valueOf(SaveStaticValue.userName));
            editor.commit();
        }

        private void loadSettings(){
            sPref = getPreferences(MODE_PRIVATE);
            String savedAuthorization = sPref.getString(AUTHORIZATION, "0");
            String savedUserName = sPref.getString(USERNAME, "0");
            SaveStaticValue.checkAuthorization                                =
Integer.parseInt(savedAuthorization);
            SaveStaticValue.userName = savedUserName;
            //Toast.makeText(AuthorizationActivity.this,         "INPUT="         +
savedAuthorization    +    "    Static="    +    SaveStaticValue.checkAuthorization,
Toast.LENGTH_SHORT).show();
        }

        @Override
        protected void onDestroy() {
            super.onDestroy();
            saveSettings();
        }

        @Override
        protected void onPause() {
            super.onPause();
            saveSettings();
        }
```

Java клас навігаційного меню

```
      @Override
      public boolean onNavigationItemSelected(@NonNull MenuItem menuItem) {
              ConnectivityManager conManager = (ConnectivityManager)

getApplicationContext().getSystemService(Context.CONNECTIVITY_SERVICE);

              NetworkInfo activeNetwork = conManager.getActiveNetworkInfo();

              if(null != activeNetwork){
                  if(activeNetwork.getType() == ConnectivityManager.TYPE_WIFI){
                      //Toast.makeText(this,       "Підключено       до       Wifi",
Toast.LENGTH_SHORT).show();
                  }
                  if(activeNetwork.getType() == ConnectivityManager.TYPE_MOBILE){
                      //Toast.makeText(this,    "Підключено    до    Data    Network",
Toast.LENGTH_SHORT).show();
                  }

                  new GetExperimentValueStatus(this).execute();
                  new GetActiveExperimentTestID(this).execute();

                  ft = getSupportFragmentManager().beginTransaction();

                  int id = menuItem.getItemId();
                  switch (id) {
                      case R.id.nav_history:
                          //Toast.makeText(this,          "Історія          замірів",
Toast.LENGTH_SHORT).show();
                          ft.replace(R.id.fr_main,       new       HistoryFragment(),
"HISTORY_FRAGMENT");
                          break;
                      case R.id.nav_now:
                          //Toast.makeText(this,          "Поточний          замір",
Toast.LENGTH_SHORT).show();
                          new GetExperimentValueStatus(this).execute();
                          new GetActiveExperimentTestID(this).execute();
                          ft.replace(R.id.fr_main,   new   SelectExperimentFragment(),
"SELECT_FRAGMENT");
                          break;
                      case R.id.nav_settings:
                          //Toast.makeText(this,                      "Налаштування",
Toast.LENGTH_SHORT).show();
                          ft.replace(R.id.fr_main,      new      SettingsFragment(),
"SETTINGS_FRAGMENT");
                          break;
                      case R.id.nav_out:
                          SaveStaticValue.checkAuthorization = 0;
                          SaveStaticValue.saveOutput = 1;
                          Intent   intent   =   new   Intent(getApplicationContext(),
AuthorizationActivity.class);
                          startActivity(intent);
                          finish();
                          break;
                      default:
                          Toast.makeText(this,                              "Виключення",
Toast.LENGTH_SHORT).show();
                          break;
```

```
                }
                drawer.closeDrawer(GravityCompat.START);
                //ft.addToBackStack(null);
                ft.commit();

            }
            else {
                Toast.makeText(this,      "Відсутнє      інтернет      підключення",
Toast.LENGTH_SHORT).show();
                ft = getSupportFragmentManager().beginTransaction();
                ft.replace(R.id.fr_main,                                                new
InternetConnectionDisabledFragment(),"INT_CON_DIS_FRAGMENT");
                //ft.addToBackStack(null);
                ft.commit();
            }
            return true;
        }
```

Java клас історії замірів

```
    public class HistoryFragment extends Fragment {

        private RecyclerView recyclerView;
        private ArrayList<HistoryRecyclerDataList> arrayList;

        @Override
        public View onCreateView(LayoutInflater inflater, ViewGroup container,
                                 Bundle savedInstanceState) {
            // Inflate the layout for this fragment
            View  view  =  inflater.inflate(R.layout.fragment_history,  container,
false);

            recyclerView = view.findViewById(R.id.rv_history);
            recyclerView.setHasFixedSize(true);
            recyclerView.setLayoutManager(new LinearLayoutManager(getActivity()));

            arrayList = new ArrayList<HistoryRecyclerDataList>();

            JsonFetch jsonFetch = new JsonFetch();
            jsonFetch.execute();

            return view;
        }

        public class JsonFetch extends AsyncTask<String, String, String>{

            HttpURLConnection httpURLConnection = null;
            String mainfile;
            @Override
            protected String doInBackground(String... strings) {

                try{
                    URL                  url                  =                  new
URL("http://batteriestests.tk/QueryApp/historyQuery.php");
                    httpURLConnection = (HttpURLConnection) url.openConnection();
                    httpURLConnection.connect();
```

```
                    InputStream inputStream = httpURLConnection.getInputStream();
                    BufferedReader   bufferedReader   =   new   BufferedReader(new
InputStreamReader(inputStream));
                    StringBuffer stringBuffer = new StringBuffer();
                    String line = "";

                    while ((line = bufferedReader.readLine())!=null){
                        stringBuffer.append(line);
                    }

                    mainfile = stringBuffer.toString();

                    JSONArray parent = new JSONArray(mainfile);

                    int i = 0;
                    while (i <= parent.length()){
                        JSONObject child = parent.getJSONObject(i);

                        String id = child.getString("test_id");
                        String status = child.getString("_status");
                        String date = child.getString("edit_date");

                        arrayList.add(new    HistoryRecyclerDataList(id,    status,
date));

                        i++;
                    }

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

                return null;
            }

            @Override
            protected void onPostExecute(String s) {
                super.onPostExecute(s);

                HistoryRecyclerAdapter      historyRecyclerAdapter      =      new
HistoryRecyclerAdapter(arrayList, getActivity());
                recyclerView.setAdapter(historyRecyclerAdapter);

            }
        }
    }
```

Java клас поточного заміру

```
public class ActiveExperimentFragment extends Fragment {
```

```
        private LineChart lineChart;
        private LineDataSet lineDataSet;
        private TextView textActiveExperiment;

        InputStream is = null;
        String line = null;
        String result = null;
        String[] napruga;
        String[] strum;

        private Handler mHandler = new Handler();

        private Runnable badTimeUpdater = new Runnable() {
            @Override
            public void run() {

                try {

                    //Text
                    textActiveExperiment                                                    =
getView().findViewById(R.id.textActiveExperiment);
                    new                     GetActiveExperimentInfo(getActivity(),
textActiveExperiment).execute();

                    //Chart
                    lineChart = getView().findViewById(R.id.linechart_now);

                    StrictMode.setThreadPolicy((new
StrictMode.ThreadPolicy.Builder().permitNetwork().build()));

                    getData();

                    ArrayList<Entry> list = new ArrayList<>();
                    for(int i = 0; i < napruga.length; i++){
                        float x = (float) Float.parseFloat(strum[i]);
                        float y = (float) Float.parseFloat(napruga[i]);
                        list.add(new Entry(i, y));
                    }

                    lineDataSet = new LineDataSet(list, "history3");
                    LineData data = new LineData(lineDataSet);
                    lineChart.setData(data);

                    //lineChart.animateX(1000);
                    lineChart.setDragEnabled(true);
                    lineChart.setScaleEnabled(true);
                    lineChart.getAxisRight().setEnabled(false);
                    lineChart.getDescription().setEnabled(false);
                    lineChart.getAxisLeft().setDrawGridLines(true);
                    lineChart.getAxisRight().setDrawGridLines(true);
                    lineChart.getXAxis().setEnabled(true);

lineChart.getXAxis().setPosition(XAxis.XAxisPosition.BOTH_SIDED);

                    lineChart.getXAxis().setLabelCount(10, false);
                    lineChart.setVisibleXRangeMaximum(strum.length);      //default
strum.length

                    IndexAxisValueFormatter         formatter         =         new
IndexAxisValueFormatter(strum); //default strum
                    lineChart.getXAxis().setGranularity(1f);
                    lineChart.getXAxis().setValueFormatter(formatter);
```

```
                    lineDataSet.setFillAlpha(110);
                    lineDataSet.setLineWidth(0);
                    lineDataSet.setValueTextSize(SaveStaticValue.valueTextSize);

lineDataSet.setValueTextColor(Color.parseColor(SaveStaticValue.valueTextColor));
                    lineDataSet.setDrawFilled(false);
                    lineDataSet.setFillColor(Color.parseColor("#000000"));
                    lineDataSet.setDrawCircles(true);
                    lineDataSet.setCircleRadius(SaveStaticValue.valueCircleRadius);

lineDataSet.setCircleColor(Color.parseColor(SaveStaticValue.valueCircleColor));

lineDataSet.setCircleHoleColor(Color.parseColor(SaveStaticValue.valueCircleColor));
                    lineDataSet.setColor(Color.parseColor("#FFFFFF"));

                    Legend l = lineChart.getLegend();
                    l.setEnabled(false);

                    lineChart.invalidate();

                }catch (Exception e){
                    Toast.makeText(getActivity(), "Зчитування вхідних даних " +
SaveStaticValue.activeID + " досліду для побудови", Toast.LENGTH_SHORT).show();
                }

                //Toast.makeText(getActivity(),          "Оновлення          даних",
Toast.LENGTH_SHORT).show();
                mHandler.postDelayed(this, 5000);
            }
        };

        @Override
        public View onCreateView(LayoutInflater inflater, ViewGroup container,
                                 Bundle savedInstanceState) {
            // Inflate the layout for this fragment

            View   view   =   inflater.inflate(R.layout.fragment_active_experiment,
container, false);

            mHandler.removeCallbacks(badTimeUpdater);
            //mHandler.post(badTimeUpdater);
            mHandler.postDelayed(badTimeUpdater, 300);

            return view;
        }

        private void getData(){
            try{

                URL                    url                    =                    new
URL("http://batteriestests.tk/QueryApp/historyDataArray.php?id="                      +
SaveStaticValue.activeID);
                HttpURLConnection con = (HttpURLConnection) url.openConnection();

                con.setRequestMethod("GET");

                is = new BufferedInputStream(con.getInputStream());

                BufferedReader br = new BufferedReader(new InputStreamReader(is));
                StringBuilder sb = new StringBuilder();
```

```
                while ((line = br.readLine()) != null){
                    sb.append(line + "\n");
                }

                is.close();
                result = sb.toString();

                JSONArray ja = new JSONArray(result);
                JSONObject jo = null;

                napruga = new String[ja.length()];
                strum = new String[ja.length()];

                for(int i = 0; i < ja.length(); i++){

                    jo = ja.getJSONObject(i);
                    napruga[i] = jo.getString("u");
                    strum[i] = " " + jo.getString("i");
                }

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

Java клас налаштувань

```
    public class SettingsFragment extends Fragment {

        private String[] valueTextSize = {"1", "2", "3", "4", "5", "6", "7", "8",
"9", "10"};
        private String[] valueTextColor = {"BLACK", "RED", "GREEN", "BLUE", "YELLOW",
"AZURE"};
        private String[] valueCircleRadius = {"1", "2", "3", "4", "5"};
        private String[] valueCircleColor = {"BLACK", "RED", "GREEN", "BLUE",
"YELLOW", "AZURE"};

        @Override
        public View onCreateView(LayoutInflater inflater, ViewGroup container,
                                 Bundle savedInstanceState) {
            // Inflate the layout for this fragment
            View view = inflater.inflate(R.layout.fragment_settings, container,
false);

            //Set text size adapter
            ArrayAdapter<String>         valueTextSizeAdapter         =         new
ArrayAdapter<String>(getActivity(),             android.R.layout.simple_spinner_item,
valueTextSize);

valueTextSizeAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown
_item);
            Spinner spValueTextSize = view.findViewById(R.id.sp_set_text_size);
            spValueTextSize.setAdapter(valueTextSizeAdapter);
            spValueTextSize.setPrompt(getString(R.string.tv_settings_text_size));
            spValueTextSize.setSelection(SaveStaticValue.selectionTextSize);
```

```
            spValueTextSize.setOnItemSelectedListener(new
AdapterView.OnItemSelectedListener() {
                @Override
                public void onItemSelected(AdapterView<?> parent, View view, int
position, long id) {

                    int choose = position;
                    switch (choose){
                        case 0: SaveStaticValue.valueTextSize = 1f;
SaveStaticValue.selectionTextSize = 0; break;
                        case 1: SaveStaticValue.valueTextSize = 2f;
SaveStaticValue.selectionTextSize = 1; break;
                        case 2: SaveStaticValue.valueTextSize = 3f;
SaveStaticValue.selectionTextSize = 2; break;
                        case 3: SaveStaticValue.valueTextSize = 4f;
SaveStaticValue.selectionTextSize = 3; break;
                        case 4: SaveStaticValue.valueTextSize = 5f;
SaveStaticValue.selectionTextSize = 4; break;
                        case 5: SaveStaticValue.valueTextSize = 6f;
SaveStaticValue.selectionTextSize = 5; break;
                        case 6: SaveStaticValue.valueTextSize = 7f;
SaveStaticValue.selectionTextSize = 6; break;
                        case 7: SaveStaticValue.valueTextSize = 8f;
SaveStaticValue.selectionTextSize = 7; break;
                        case 8: SaveStaticValue.valueTextSize = 9f;
SaveStaticValue.selectionTextSize = 8; break;
                        case 9: SaveStaticValue.valueTextSize = 10f;
SaveStaticValue.selectionTextSize = 9; break;
                        default: Toast.makeText(getActivity(), "Розмір тексту null",
Toast.LENGTH_SHORT).show(); break;
                    }
                }

                @Override
                public void onNothingSelected(AdapterView<?> parent) {
                    Toast.makeText(getActivity(), "Розмір тексту не було змінено",
Toast.LENGTH_SHORT).show();
                }
            });

            //Set text color adapter
            ArrayAdapter<String> valueTextColorAdapter = new
ArrayAdapter<String>(getActivity(), android.R.layout.simple_spinner_item,
valueTextColor);

valueTextColorAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdow
n_item);
            Spinner spValueTextColor = view.findViewById(R.id.sp_set_text_color);
            spValueTextColor.setAdapter(valueTextColorAdapter);
            spValueTextColor.setPrompt(getString(R.string.tv_settings_text_color));
            spValueTextColor.setSelection(SaveStaticValue.selectionTextColor);

            spValueTextColor.setOnItemSelectedListener(new
AdapterView.OnItemSelectedListener() {
                @Override
                public void onItemSelected(AdapterView<?> parent, View view, int
position, long id) {

                    int choose = position;
                    switch (choose){
                        case 0: SaveStaticValue.valueTextColor = "#000000";
SaveStaticValue.selectionTextColor = 0; break;
```

```
                        case   1:   SaveStaticValue.valueTextColor   =   "#FF0000";
SaveStaticValue.selectionTextColor = 1; break;
                        case   2:   SaveStaticValue.valueTextColor   =   "#00FF00";
SaveStaticValue.selectionTextColor = 2; break;
                        case   3:   SaveStaticValue.valueTextColor   =   "#0000FF";
SaveStaticValue.selectionTextColor = 3; break;
                        case   4:   SaveStaticValue.valueTextColor   =   "#FFFF00";
SaveStaticValue.selectionTextColor = 4; break;
                        case   5:   SaveStaticValue.valueTextColor   =   "#3F487F";
SaveStaticValue.selectionTextColor = 5; break;
                        default: Toast.makeText(getActivity(), "Колір тексту null",
Toast.LENGTH_SHORT).show(); break;
                    }
                }

                @Override
                public void onNothingSelected(AdapterView<?> parent) {
                    Toast.makeText(getActivity(), "Колір тексту не було змінено",
Toast.LENGTH_SHORT).show();
                }
            });



            //Set circle radius adapter
            ArrayAdapter<String>        valueRadiusCircleAdapter        =        new
ArrayAdapter<String>(getActivity(),             android.R.layout.simple_spinner_item,
valueCircleRadius);

valueRadiusCircleAdapter.setDropDownViewResource(android.R.layout.simple_spinner_drop
down_item);
            Spinner                         spValueRadiusCircle                         =
view.findViewById(R.id.sp_set_circle_radius);
            spValueRadiusCircle.setAdapter(valueRadiusCircleAdapter);

spValueRadiusCircle.setPrompt(getString(R.string.tv_settings_circle_radius));

spValueRadiusCircle.setSelection(SaveStaticValue.selectionCircleRadius);

            spValueRadiusCircle.setOnItemSelectedListener(new
AdapterView.OnItemSelectedListener() {
                @Override
                public void onItemSelected(AdapterView<?> parent, View view, int
position, long id) {

                    int choose = position;
                    switch (choose){
                        case    0:    SaveStaticValue.valueCircleRadius    =    1f;
SaveStaticValue.selectionCircleRadius = 0; break;
                        case    1:    SaveStaticValue.valueCircleRadius    =    2f;
SaveStaticValue.selectionCircleRadius = 1; break;
                        case    2:    SaveStaticValue.valueCircleRadius    =    3f;
SaveStaticValue.selectionCircleRadius = 2; break;
                        case    3:    SaveStaticValue.valueCircleRadius    =    4f;
SaveStaticValue.selectionCircleRadius = 3; break;
                        case    4:    SaveStaticValue.valueCircleRadius    =    5f;
SaveStaticValue.selectionCircleRadius = 4; break;
                        default: Toast.makeText(getActivity(), "Радіус точки null",
Toast.LENGTH_SHORT).show(); break;
                    }
                }

                @Override
```

```
                public void onNothingSelected(AdapterView<?> parent) {
                    Toast.makeText(getActivity(), "Радіус точки не було змінено",
Toast.LENGTH_SHORT).show();
                }
            });



            //Set circle color adapter
            ArrayAdapter<String>        valueCircleColorAdapter        =        new
ArrayAdapter<String>(getActivity(),             android.R.layout.simple_spinner_item,
valueCircleColor);

valueCircleColorAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropd
own_item);
            Spinner                        spValueCircleColor                        =
view.findViewById(R.id.sp_set_circle_color);
            spValueCircleColor.setAdapter(valueCircleColorAdapter);

spValueCircleColor.setPrompt(getString(R.string.tv_settings_circle_color));
            spValueCircleColor.setSelection(SaveStaticValue.selectionCircleColor);

            spValueCircleColor.setOnItemSelectedListener(new
AdapterView.OnItemSelectedListener() {
                @Override
                public void onItemSelected(AdapterView<?> parent, View view, int
position, long id) {

                    int choose = position;
                    switch (choose){
                        case  0:  SaveStaticValue.valueCircleColor  =  "#000000";
SaveStaticValue.selectionCircleColor = 0; break;
                        case  1:  SaveStaticValue.valueCircleColor  =  "#FF0000";
SaveStaticValue.selectionCircleColor = 1; break;
                        case  2:  SaveStaticValue.valueCircleColor  =  "#00FF00";
SaveStaticValue.selectionCircleColor = 2; break;
                        case  3:  SaveStaticValue.valueCircleColor  =  "#0000FF";
SaveStaticValue.selectionCircleColor = 3; break;
                        case  4:  SaveStaticValue.valueCircleColor  =  "#FFFF00";
SaveStaticValue.selectionCircleColor = 4; break;
                        case  5:  SaveStaticValue.valueCircleColor  =  "#3F487F";
SaveStaticValue.selectionCircleColor = 5; break;
                        default: Toast.makeText(getActivity(), "Колір точки null",
Toast.LENGTH_SHORT).show(); break;
                    }
                }

                @Override
                public void onNothingSelected(AdapterView<?> parent) {
                    Toast.makeText(getActivity(), "Колір точки не було змінено",
Toast.LENGTH_SHORT).show();
                }
            });

            return view;
        }
    }
```

ДОДАТОК Д

Мобільний додаток моніторингу електрохімічних характеристик електродів для літій-іонних батарей

Кобцов В.Ю., *студент*; Алексенко О.В., *доцент*,
Нагорний В.В., *старший викладач*; Парфененко Ю.В., *доцент*
Сумський державний університет, м. Суми, Україна

Літій-іонні батареї широко використовуються в якості джерела енергії для широкого ряду електронних пристроїв але мають істотний недолік – у разі їх «перезарядки» може виникнути коротке замикання, що призводить до подальшого згоряння. Щоб цього уникнути розробляються алгоритми їх автоматичного відключення від мережі при досягненні критичних режимів. Ці алгоритми потребують детальних даних щодо температури, при якій починається перегрів у процесі зарядки-розрядки акумуляторів [1]. Щоб отримати цю інформацію розробникам літій-іонних батарей доводиться проводити багато експериментів, які генерують великі обсяги вихідних даних, що потребують ретельного аналізу.

Потрібно зауважити, що аналіз інформації, представленої у графічній формі, потребує значно менше часу та зусиль дослідників порівняно із табличним представленням. Тому створення зручного інструменту для візуалізації цифрових даних є актуальним завданням.

Виходячи із вищезазначеного була визначена мета проекту – розробка мобільного додатку моніторингу електрохімічних характеристик електродів для літій-іонних батарей.

Для досягнення мети проекту було проведено аналіз систем-аналогів на основі дослідження предметної області, здійснено вибір та налаштування інструментів реалізації, розроблено макет користувацького інтерфейсу, спроектовано алгоритми візуалізації даних. Показники для графічного відображення надходять з бази даних.

Результатом виконання проекту є розроблений мобільний додаток на платформі Android, використання якого полегшить аналіз вольт-амперних характеристик та кривих зарядки-розрядки акумуляторних батарей при проведенні експериментів з оцінки їх властивостей, а також дозволить проводити експерименти віддалено, без прив'язки до робочого місця.

Finance : веб-сайт. URL: https://news.finance.ua/ru/news/-/421013.