МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМ. В. ДАЛЯ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЕЛЕКТРОНІКИ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК ТА ІНЖЕНЕРІЇ

До захисту допускається
Завідувач кафедри
_____ Скарга-Бандурова І. С.
«____»_____ 2019 р.

ДИПЛОМНИЙ ПРОЕКТ БАКАЛАВРА
ПОЯСНЮВАЛЬНА ЗАПИСКА

НА ТЕМУ:

Комплексна тема. Мобільний додаток обліку фінансових операцій: стаціонарний модуль

Освітньо-кваліфікаційний рівень – бакалавр
Напрям підготовки 6.050102 «Комп'ютерна інженерія»

Керівник проекту:	_____ (підпис)	Скарга-Бандурова І.С. (ініціали, прізвище)
Консультант з охорони праці:	_____ (підпис)	Критська Я. О. (ініціали, прізвище)
Студент:	_____ (підпис)	Рибальченко В.Р. (ініціали, прізвище)
Група:		КІ-15д

Сєвєродонецьк-2019

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Факультет інформаційних технологій та електроніки
Кафедра комп'ютерних наук та інженерії
Освітньо-кваліфікаційний рівень бакалавр
Спеціальність 6.050102 "Комп'ютерна інженерія"

ЗАТВЕРДЖУЮ

Завідувач кафедри КНІ
_____Скарга-Бандурова І. С.
"____" _____2019 р.

ЗАВДАННЯ

НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) БАКАЛАВРА

Рибальченку Володимиру Романовичу
(прізвище, ім'я, по батькові)

1.Тема проекту (роботи) Комплексна тема. Мобільний додаток обліку фінансових операцій: стаціонарний модуль

керівник проекту (роботи) Скарга-Бандурова І.С., д.т.н., проф.
(прізвище, ініціали, науковий ступінь, вчене звання)

затверджені розпорядженням по кафедрі від "__"_____20____року

2.Строк подання студентом проекту (роботи) _____

3.Вихідні дані до проекту (роботи) матеріали переддипломної практики

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Аналіз систем обліку фінансів та формулювання технічного завдання; проектування системи; розробка інтерфейсу користувача; охорона праці та безпека в надзвичайних ситуаціях

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслеників) Електронні плакати

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці та безпека в надзвичайних ситуаціях	Критська Я. О., ст. викл. кафедри комп'ютерних наук та інженерії		

7. Дата видачі завдання: _____

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту	Строк виконання етапів	Примітка
1.	Отримання завдання, збір матеріалів		
2.	Вивчення відповідної літератури		
3.	Розроблення технічного завдання		
4.	Розгляд існуючих рішень щодо вирішення задачі.		
5.	Розробка інтерфейсу.		
6.	Розробка програмного коду і тестування		
7.	Охорона праці та безпека в надзвичайних ситуаціях		
8.	Оформлення пояснювальної записки		

Студент _____ Рибальченко В. Р.
(підпис) (прізвище та ініціали)

Керівник _____ Скарга-Бандурова І.С.
(підпис) (прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до дипломного проекту містить 82 стор., 22 рис., 5 табл., 2 додатки, 21 посилань.

Мета роботи – розробка мобільного додатку для контролю грошей за допомогою ведення своїх витрат і доходів кожного дня.

Об’єктом дослідження – побудова мобільного клієнту для зручного обліку фінансів.

У першому розділі надано огляд предметної області, проаналізовано існуючі рішення, визначено задачі роботи.

У другому розділі представлена архітектура системи, описана взаємодія користувача з системою, вибрано середовище для розробки та описано основний функціонал системи.

Третій розділ присвячено розробці інтерфейсу користувача.

Четвертий розділ присвячений охороні праці та безпеці в надзвичайних ситуаціях.

Ключові слова: додаток, фінансові операції, Android.

ЗМІСТ

ВСТУП

Гроші – це важливий ресурс в житті кожної людини. Більшості людей не вистачає своїх грошей, бо вони навіть не уявляють, скільки їм потрібно для життя. Саме із-за цього через декілька днів після отримання зарплати або інших доходів у людей закінчуються більша частина грошей. Для того, щоб цього не траплялось потрібно контролювати свої витрати та доходи. В першу чергу це не тільки допоможе більш вдало розподілити гроші на місяць, але і заощадити.

На сьогоднішній день більшість людей мають смартфони з доступом в інтернет. Слідкувати за своїми грошима вже не так складно. Не потрібно заводити блокнот і кожен раз дописувати витрати або доходи. Все це можна помістити у свій смартфон.

Користування мобільним додатком для обліку своїх фінансів в наш час дуже зручно, бо смартфон завжди під рукою. Зробити пару дотиків для запису не складно і це не буде відволікати від буденних справ.

Якщо ж користування мобільним додатком з якоїсь причини неможливе, можна скористатися веб-версією додатку. Веб-версія найчастіше майже нічим не відрізняється, крім більшого використання інтернет трафіку та менш зручним використанням деяких функцій.

Нажаль більшість сучасних додатків для контролю своїх грошей мають перевантажений інтерфейс, багато функцій відкривається тільки за додаткову плату, чи навіть додаток повністю платний.

Таким чином метою даного дипломного проекту є розробка зручного мобільного додатку для обліку своїх фінансів.

Об’єктом розробки є мобільний додаток класу money manager, який дозволяє користувачеві зібрати всі персональні фінансові операції в одному місті.

1 АНАЛІЗ СИСТЕМ ТА ФОРМУЛЮВАННЯ ТЕХНІЧНОГО ЗАВДАННЯ

За останні роки мобільні пристрої набули чималу популярність. Дуже багато людей вже не уявляють своє життя без смартфону чи планшету. В наш час найбільш популярними мобільними ОС є Android компанії Goggle ТА IOS виробництва компанії Apple. В цьому розділі роботи надано огляд предметної області, проаналізовано існуючі рішення, визначено задачі роботи.

1.1 Огляд предметної області

За статистичними даними всесвітньої частки ринку провідних операційних систем смартфонів з точки зору продажів кінцевим користувачам за перші два квартали 2018 року ОС Android займає 88% ринку, а на другому місті знаходиться ОС IOS з 11.9% ринку [1].

На рис. 1.1 представлена статистика, що показує глобальну частку ринку провідних операційних систем смартфонів, з точки зору продажів кінцевим користувачам, з 2009 по 2018 рік.

Виходячи з наведених статистичних даних, можна зробити висновок ,що мобільні додатки розроблені для операційної системи Android мають високу популярність.

Ідея відстеження витрат хоча б раз приходить в голову майже кожній людині. Згідно зі звітом [2], 70% дорослих жителів США використовують програми особистих фінансів для відстеження своїх витрат і дотримання бюджету. Сімейний бюджет - це той самий облік, як і в бізнесі, і не можна заперечувати, що програмне забезпечення для фінансового управління значно полегшує роботу.

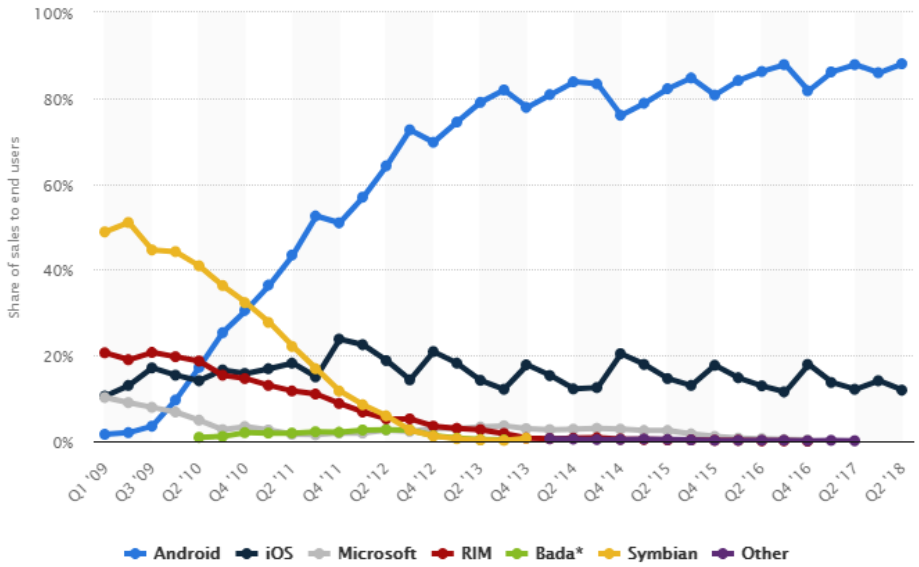


Рисунок 1.1 – Частка світового ринку провідних операційних систем смартфонів у продажу кінцевим споживачам з 1 кварталу 2009 року до 2-го кварталу 2018 року

Основними перевагами мобільного додатку є:

– зручність використання, смартфон завжди під рукою і не буде складно додати ту чи іншу операцію;

– легкий в освоєнні та використанні інтерфейс з дуже низьким порогом входження, навить людина з малим досвідом використання смартфону зможе впоратися;

– дешева у вживанні ресурсів смартфону;

– можливість зберігати данні в хмарі, навіть якщо загубити смартфон можна легко перенести данні до іншого смартфону.

Непоганим доповненням функціоналу є також можливість синхронізувати свої данні через спільний акаунт. Це відкриває можливість створення веб-версії додатку, користування однією базою на різних пристроях, бо не завжди є можливість користуватися смартфоном. Також це може зберегти данні якщо ви загубите телефон або він зламається.

На даний момент розроблено сотні фінансових програм для обліку і ведення особистих фінансів. Вони відрізняються інтерфейсом, набором

функцій, можливістю синхронізації. Деякі програми допомагають зберегти та накопичити капітал, інші дозволяють користувачам організовувати витрати або збирати інформацію про банки та курси валют. Разом з тим, питання їх зручності й тривалого використання залишається відкритим.

1.2 Огляд існуючих рішень

Нижче, розглядаються деякі популярні безкоштовні аналоги додатку з Google Play.

1.2.1 Monefy

Monefy один з найпопулярніших додатків для обліку фінансів авторами якого являються Aimbity AS. Додаток підтримується на ОС Android, iOS та Windows Phone. Monefy немає великої кількості функціоналу та складного інтерфейсу, але саме це і дозволяє йому залишатися одним із лідером на ринку мобільних додатків для обліку фінансів.

Основними перевагами даного додатку являються наявність резервного копіювання даних, що забезпечує можливість перенесення даних на інший пристрій чи відновлення втрачених даних. Велика кількість категорій та можливість їхнього редагування. Синхронізація даних через акаунт Google Drive чи Dropbox. Інтуїтивно зрозумілий інтерфейс користувача. Зручний віджет для створення записів з головного екрану.

Недоліків у додатку небагато, але вони є. Більша частина його переваг відкривається лише у платній версії. Безкоштовна версія душе обмежена. Неможливе редагування категорій, створення нових категорій та редагування вже існуючих, використання віджету на головній сторінці. Відсутня синхронізація зі стаціонарними пристроями. Також один із недоліків це відсутність веб-версії додатку

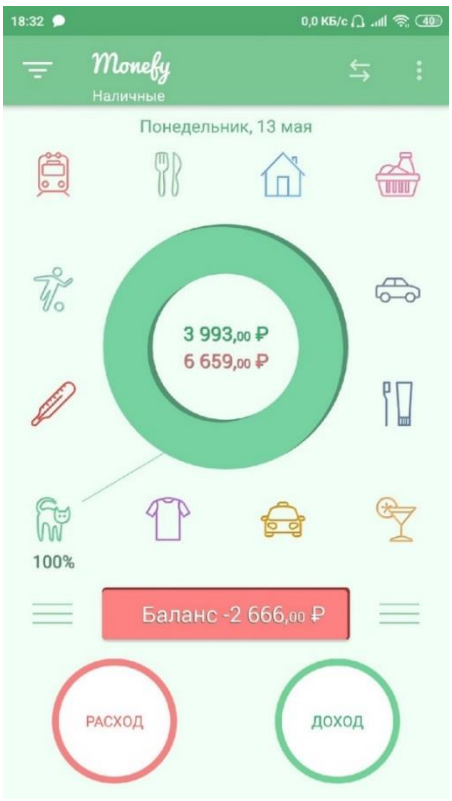


Рисунок 1.2 – Головна сторінка додатку Monefy

1.2.2 MoneyLover

MoneyLover – це додаток з дуже великим функціоналом. Крім звичайних фінансових операцій можливо додати багато різних відгалужень. Додаток підтримується на ОС Android, iOS, Windows Phone та має веб версію додатку на сайті.

Основними перевагами додатку є підключення к банку для автоматизованого відстеження операцій. Сканер квитанцій дозволяє швидко додавати нові операції. Створення операцій, які будуть автоматично повторюватися з певною періодичністю. Повна можливість редагування категорії, або створення своїх. Синхронізація через акаунт Google, Facebook або через електрону пошту.

Недоліками додатку є перевантажений інтерфейс, багато з функціоналу непотрібно користувачу і лише заважає. Велика кількість реклами яку можливо вимкнути лиже за окрему плату.

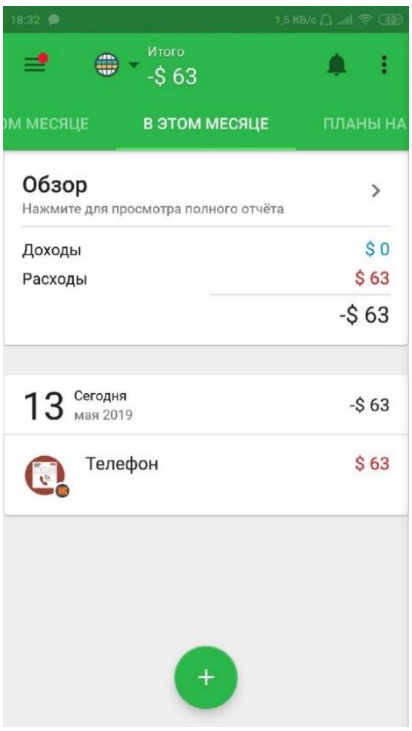


Рисунок 1.3 – Головна сторінка додатку MoneyLover

1.2.3 1Money

1Money – це додаток який набув популярність не так давно, але вже знаходиться у вершині списку Google Play. Розроблений компанією PixelRush. На сам перед це дуже простий додаток в якому немає нічого зайвого. Він має тільки ті дані, що потрібні користувачу.

Основними перевагами додатку є зручність та простота інтерфейсу. Редагування та додавання нових категорій. Зручна і зрозуміла статистика, яка відображує всі операції за останній місяць у виді діаграми для доходів та витрат. Підтримка великої кількості різних валют.

Недоліками додатку є обмеження створення категорій у безкоштовній версії, десять категорій витрат та одна доходів. Синхронізація доступна тільки у платній версії. Резервне копіювання теж тільки у платній версії. Немає веб-версії.

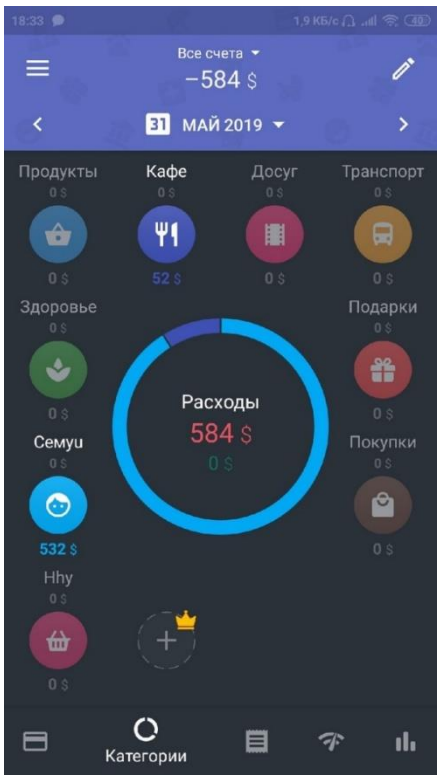


Рисунок 1.4 – Головна сторінка додатку 1Money

1.2.4 CoinKeeper

CoinKeeper – додаток, що на перший погляд справляє душе гарні враження, але вже після встановлення на телефон тобі майже через кожен клік хочуть щось продати.

Основні переваги додатку є зчитування банківських смс и додавання операцій автоматично. Редагування та додавання нових категорій. Наявність веб-версії додатку.

Недоліків у додатку дуже багато:

По-перше не зручний спосіб додавання витрат та доходів. Навіть при середніх розмірах екрану, дуже важко дотягнутися до потрібної категорії однією рукою.

По-друге багато нав’язливої реклами преміум-версії в котрій і знаходиться основний функціонал, а саме: синхронізація між веб-версією та мобільним пристроєм, додавання скільки завгодно категорій.

По-третє додаток зроблений не якісно і має багато технічних проблем.

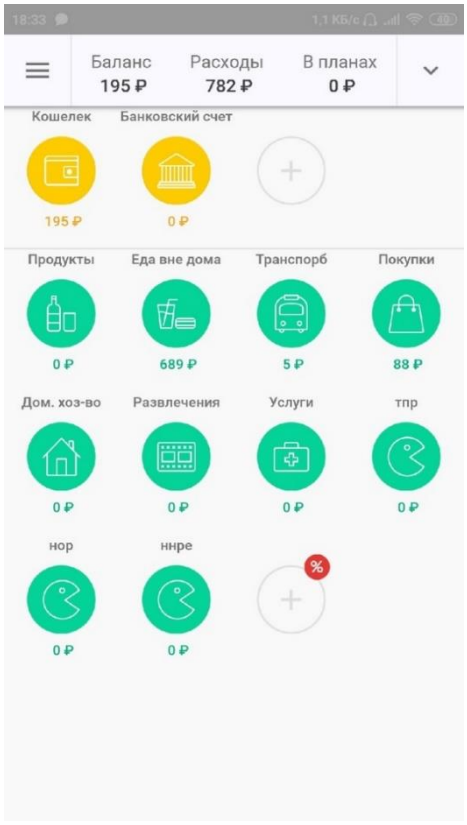


Рисунок 1.5 – Головна сторінка додатку CoinKeeper

За результатом проведеного аналізу конкурентів можна зробити висновок, що багато розробників додатків забувають про користувачів і перевантажують інтерфейс у гонитві за більшим функціоналом у преміум версії свого додатку. Це говорить про необхідність розробки зручного додатку без зайвих функцій та без обмежень користувача.

Таблиця 1.1 – Порівняльна таблиця конкурентів

Вимоги	Monefy	MoneyLover	1Money	CoinKeeper
Редагування категорій	ні	так	так	так
Синхронізація пристроїв	ні	так	ні	ні
Створення графіків	так	так	так	так

Вимоги	Monefy	MoneyLover	1Money	CoinKeeper
Підключення до банків	ні	так	ні	ні
Наявність платної версії	так	так	так	так
Наявність реклами	ні	так	так	так

1.3 Формулювання завдання роботи

1.3.1 Назва проекту

Додаток обліку фінансових операцій для мобільних пристроїв на базі ОС Android “Cash Eater”.

1.3.2 Призначення розробки

Розробка створена для зручного обліку фінансів на смартфонах на базі ОС Android та забезпечення синхронізації мобільного додатку с веб-версією.

1.3.3 Короткий опис

Для зручності користування фінансові операції повинні бути поділенні на категорії. Категорії насамперед повинні мати можливість редагування на розсуд користувача. Кожна операція має запам’ятовуватися та відображуватися у зручному вигляді для того, щоб було зрозуміло коли і на, що були витрачені гроші, чи одержані. Також необхідно відображувати зручну статистику витрат та доходів за певний період часу. Це мінімум функціоналу

який потрібен користувачу і на мою думку все останнє буде лише навантажувати інтерфейс.

1.3.4 Вимоги до функціоналу

Додаток повинен виконувати наступні функції:

– Можливість вести облік фінансових операцій, проглядати, редагувати да заповнювати історію грошових транзакцій.

– Можливість створення та редагування особистих категорій, аналіз графіків витрат, прибутків.

– Мінімізація займаної постійної пам’яті на пристрої, високий рівень плавності роботи.

– Синхронізація зі стаціонарним модулем за допомогою облікового запису.

1.3.5 Вимоги до інтерфейсу

Головні вимоги до інтерфейсу – це практичність, інтуїтивна зрозумілість і зручність. Користувач повинен знайти будь-яку доступну для нього інформацію не більш, ніж за 3 кроки.

На головній сторінці додатку повинні явно виділятися кнопки додавання та віднімання фінансів. Для зручного керування однією рукою вони повинні знаходитися у нижній частині екрану. Також на головній сторінці має відображуватися зрозуміла статистика доходів та витрат у вигляді кругової діаграми.

На сторінці додавання нової операції повинен знаходитися зручний калькулятор, щоб користувач не витрачав час на рахування своїх грошей самостійно.

1.3.6 Вимоги до програмного забезпечення

Мобільний модуль має працювати на мобільних пристроях під керівництвом ОС Android версії не нижче ніж 5.0 Lolipop.

1.3.7 Вимоги до апаратного забезпечення

Мінімальні системні вимоги для роботи з мобільним модулем:

- Процесор ARM Cortex-A8 з частотою не менш ніж 600 МГц;Є
- Оперативна пам’ять – не менше 1 ГБ;
- Розмір екрану – не менше ніж 4.5 дюймів;
- Постійна пам’ять – не менше ніж 25 МБ.

Висновки до розділу 1

У першому розділі було розглянуто предметна область розробки, кілька популярних аналогів та зроблений аналіз їх переваг та недоліків. Було визначено призначення розробки та сформульовані вимоги до майбутнього додатку.

2 ПРОЕКТУВАННЯ СИСТЕМИ

У Розділі представлена архітектура розроблюваної системи. Наведені результати проектування бази даних. Описано взаємодію користувачів системи. Надані відомості про середовище розробки. Описано основний функціонал системи.

2.1 Архітектура системи

При розробки програми в першу чергу потрібно розробити архітектуру системи. Архітектура системи – принципова організація системи, втілена в її елементах, їх взаємини один з одним і з середовищем, а також принципи, що направляють її проектування і еволюцію [7].

2.1.1 Компоненти системи

Система складається з чотирьох компонентів (рис 2.1) – мобільного додатку, серверу додатка, веб-клієнту та серверу бази даних.

Сервер додатку є посередником між мобільним клієнтом, веб-клієнтом та сервером бази даних. Він поєднує у собі сервер автентифікації і авторизації та сервер синхронізації даних.

Сервер додатку відповідає за прийом інформації, для подальшої її обробки і передачі на сервер бази даних. Він має підсистему автентифікації, підсистему синхронізації та підсистему керування користувачами. Підсистема синхронізації відповідає за синхронізацію даних на сервері бази даних при використанні одного акаунта на мобільному додатку та веб-клієнті. Вона тісно зв’язана з підсистемою автентифікації. Підсистема автентифікації перевіряє справжність користувача шляхом порівняння введеного їм логіна з паролем, збереженим в базі даних користувачів. Підсистема керування користувачами відповідає за створення, збереження, та зміну особистих даних користувача.

Забезпечення доступу к підсистемі керування користувача відбувається за допомогою клієнтських додатків.

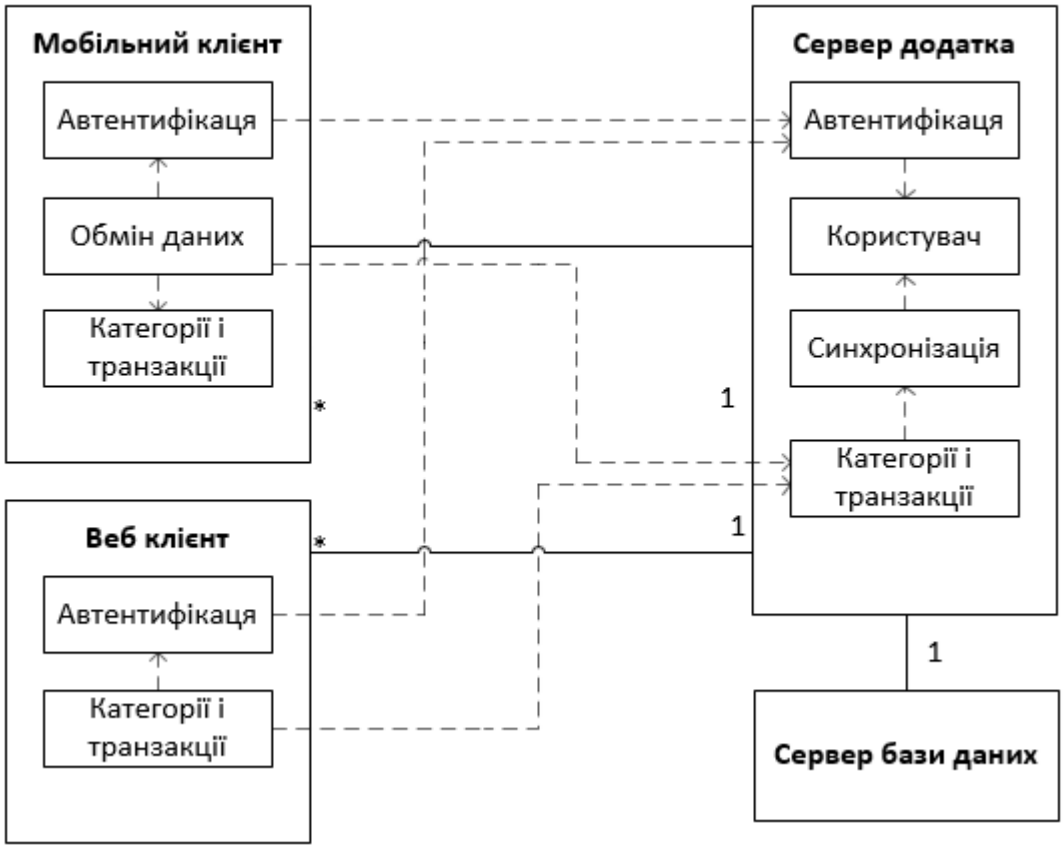


Рисунок 2.1 – Схема розгортання системи

Мобільний клієнт має можливість працювати у двох режимах. Режимі онлайн і в режимі офлайн. У режимі онлайн клієнт відповідає за передачу запитів користувача до серверу додатку за допомогою інтерфейсу. Для додавання, зміни, видалення, даних з серверу бази даних. Та відображення цих даних у зручному вигляді. Офлайн режим робить теж саме, але вже на локальній базі даних. При роботі офлайн весь функціонал зберігається. Для перенесення даних до серверу бази даних потрібно підключитися до інтернету та увійти або створити акаунт за допомогою підсистеми автентифікації. Після цього дані будуть синхронізовані з даними на сервері бази даних. За це відповідає підсистема обміну даних.

Веб-клієнт має той же функціонал, що і мольний клієнт крім можливості роботи у офлайн режими.

2.1.2 Проектування бази даних

Проектування бази даних - це процес створення схеми бази даних і визначення необхідних обмежень цілісності.

Процес проектування даних можна умовно розділити на два етапи: логічне моделювання і фізичне проектування. Результатом першого з них є так звана логічна (або концептуальна) модель даних, що виражається зазвичай діаграмою «сутність-зв’язок» або ER (Entity-Relationship) діаграмою, яка представлена в одній зі стандартних нотацій, прийнятих для відображення подібних діаграм. Результатом другого етапу є готова база даних.

Логічна модель даних описує факти і об’єкти, що підлягають реєстрації в майбутній базі даних. Основними компонентами такої моделі є сутності, їх атрибути та зв’язки між ними. Як правило, фізичним аналогом сутності в майбутній базі даних є таблиця, а фізичним аналогом атрибута - поле цієї таблиці. З логічної точки зору сутність являє собою сукупність однотипних об’єктів або фактів, які називаються екземплярами цієї сутності. Фізичним аналогом екземпляру зазвичай є запис в таблиці бази даних. Як і записи в таблиці реляційної СУБД, екземпляри сутності повинні бути унікальними, тобто повний набір значень їх атрибутів не повинен дублюватися. І так само, як і поля в таблиці, атрибути можуть бути ключовими і неключовими. На етапі логічного проектування для кожного атрибута зазвичай визначається приблизний тип даних (строковий, числовий, BLOB і ін.). Конкретизація відбувається на етапі фізичного проектування, так як різні СУБД підтримують різні типи даних і обмеження на їх довжину або точність.

Для логічного проектування бази даних використовуються різні методи, один з таких методів називається метод «сутність-зв’язок» також відомий як ER-модель (англ. Entity-relationship model або entity-relationship diagram).

Модель «сутність-зв’язок» це модель даних, яка дозволяє описувати концептуальні схеми за допомогою узагальнених конструкцій блоків. ER-модель — це мета-модель даних, тобто засіб опису моделей даних. Існує ряд моделей для представлення знань, але одним з найзручніших інструментів уніфікованого представлення даних, незалежного від програмного забезпечення, що його реалізує, є модель «сутність-зв’язок». Важливим є той факт, що з моделі «сутність-зв’язок» можуть бути породжені всі існуючі моделі даних (ієрархічна, мережева, реляційна, об’єктна), тому вона є найзагальнішою[8].

Модель «сутність-зв’язок» є результатом систематичного процесу, який описує та визначає деяку предметну область. Вона не визначає сам процес, а лише візуалізує його.

У процесі проектування були визначені наступні сутності: “Користувачі”, “Операції”, “Категорії”.

Для кожної сутності були визначені її атрибути.

Атрибути сутності “Користувачі”:

– Код користувача
– Пошта користувача
– Пароль користувача
– Баланс користувача
– Дата оновлення

Атрибути сутності “Операції”:

– Код операції
– Код категорії
– Дата операції
– Сума операції
– Примітка операції
– Дата оновлення

Атрибути сутності “ Категорії”:

– Код категорії

- Код користувача
- Назва категорії
- Колір категорії
- Зображення категорії
- Дохід чи витрати
- Дата оновлення

Між об’єктами існують наступні типи відносин:

Один-до-одного (1:1). Кожному екземпляру першого інформаційного об’єкта відповідає тільки один екземпляр другого інформаційного об’єкта.

Один-до-багатьох (1:M). Кожному екземпляру одного інформаційного об’єкта відповідає кілька екземплярів іншого інформаційного об’єкта, а кожному примірнику другого інформаційного об’єкта відповідає не більше одного примірника першого інформаційного об’єкта.

Багато-до-багатьох (M:M). Кожному екземпляру одного інформаційного об’єкта відповідає кілька екземплярів іншого інформаційного об’єкта і кожному екземпляру другого інформаційного об'єкта може відповідати кілька екземплярів першого.

Зв’язок “Користувачі” - “Категорії”: Тип зв’язку 1:M. До одного користувача відноситься багато категорій, а категорія має лише одного користувача. Зв’язок між цими об'єктами забезпечується за допомогою атрибуту Код користувача.

Зв’язок “Користувачі” - “Операції”: Тип зв’язку 1:M. До одного користувача відноситься багато операцій, а операція має лише одного користувача. Зв’язок між цими об'єктами забезпечується за допомогою атрибуту Код користувача.

Зв'язок “Категорії” - “Операції”: Тип зв'язку 1:M. До одної категорії відноситься багато операцій, а операція має лише одну категорію. Зв'язок між цими об'єктами забезпечується за допомогою атрибуту Код категорії

Після проектування можна побудувати фізичну схему бази даних (Рис.2.2).

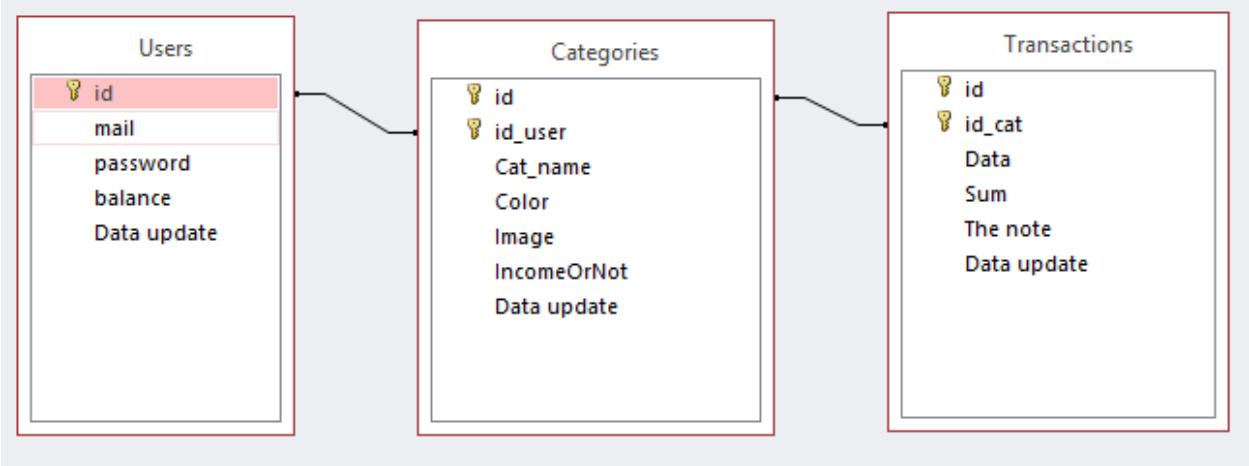


Рисунок 2.2 – Схема бази даних

Фізичне проектування - створення схеми бази даних для конкретної СУБД. Специфіка конкретної СУБД може включати в себе обмеження на іменування об'єктів бази даних, обмеження на підтримувані типи даних та інші. Крім того, специфіка конкретної СУБД при фізичному проектуванні включає вибір рішень, пов'язаних з фізичним середовищем зберігання даних (вибір методів управління дисковою пам'яттю, поділ БД по файлам і пристроям, методів доступу до даних), створення індексів та інші [8].

Таблиця Users – містить у собі основні дані зареєстрованих користувачів. В таблиці зберігаються такі дані як, унікальний код користувача, електронна адреса, пароль, поточний баланс користувача та дата останнього обновлення даних.

Таблиця Categories – містить у собі всі категорії користувачів. В таблиці зберігаються такі дані як, унікальний код категорії, унікальний код користувача, назва категорії її колір та зображення, чи відноситься категорія до доходів чи до витрат та дата останнього обновлення даних.

Таблиця Transaction – містить у собі операції, що додали користувачі. В таблиці зберігаються такі дані як, унікальний ключ операції, унікальний ключ категорії, дата створення операції, сума операції, примітка до операції та дата останнього обновлення даних.

2.1.3 SQLite

Для реалізації бази даних на мобільному клієнті було використано база даних SQLite. SQLite – це вбудована бібліотека, яка реалізує автономний, безсистемний, транзакційний механізм баз даних SQL з нульовою конфігурацією. Код для SQLite є відкритим доступом і, таким чином, є безкоштовним для використання в будь-яких цілях, комерційних або приватних. SQLite є одною із найбільш розповсюдженою базою даних у світі з великим числом програм, включаючи кілька проектів високого рівня.

SQLite - це вбудований движок баз даних SQL. У відмінності від більшості інших баз даних SQL, SQLite не має окремого серверного процесу. SQLite читає і пише безпосередньо в звичайні файли на диску. Повна база даних SQL з декількома таблицями, індексами, тригерами і уявленнями, міститься в одному файлі на диску. Формат файлу бази даних є кросплатформним - ви можете вільно копіювати базу даних між 32-розрядними та 64-розрядними системами або між архітектурою з прямим і молодшим порядком байтів. Ці функції роблять SQLite популярним вибором в якості формату файлу програми. Файли бази даних SQLite - це рекомендований формат зберігання Бібліотеки Конгресу США.

SQLite - це компактна бібліотека. При всіх включених функціях розмір бібліотеки може бути менше 600 КБ, в залежності від цільової платформи і налаштувань оптимізації компілятора. Існує компроміс між використанням пам'яті і швидкістю. SQLite зазвичай працює так, що чим більше пам'яті ви даєте тим швидше працює база. Проте продуктивність, як правило, досить хороша, навіть в умовах браку пам'яті. Залежно від того, як він використовується, SQLite може бути швидше, ніж пряме введення-виведення файлової системи.

SQLite дуже ретельно тестується перед кожним випуском і має репутацію дуже надійного. Велика частина вихідного коду SQLite присвячена виключно тестуванню і перевірці. Автоматизований набір тестів виконує мільйони і

мільйони тестових випадків, що включають сотні мільйонів окремих операторів SQL, і забезпечує 100% охоплення розгалужень тестування. SQLite витончено реагує на збої виділення пам'яті і помилки введення-виведення диска. Все це підтверджується автоматизованими тестами з використанням спеціальних тестових наборів, які імітують збої системи. Звичайно, навіть після всього цього тестування, все ще є помилки. Але на відміну від деяких подібних проектів (особливо комерційних конкурентів) SQLite відкритий і чесний по відношенню до всіх помилок і надає списки помилок і щохвилинні хронології змін коду.

База коду SQLite підтримується міжнародною командою розробників, які працюють над SQLite повний робочий день. Розробники продовжують розширювати можливості SQLite і підвищувати його надійність і продуктивність, зберігаючи сумісність з опублікованими специфікаціями інтерфейсу, синтаксисом SQL і форматом файлу бази даних. Вихідний код абсолютно безкоштовний для всіх, хто цього хоче [9].

2.2 Взаємодія користувача з системою

2.2.1 Реєстрація користувача та вхід до акаунту

Для користування мобільним додатком реєстрація не є обов’язковою, але для збереження даних на сервері бази даних потрібно зареєструватися. Для реєстрації або входу до вже існуючого акаунту потрібно відкрити бокове меню на головній сторінці мобільного додатку и натиснути кнопку “Вхід”. Після цього на екрані з’явиться форма входу до акаунту та кнопка “Реєстрації”. Натиснув цю кнопку з’явиться форма реєстрації. Заповнивши всі поля форми такі, як електронна пошта та пароль буде складено запит до сервера додатку для додавання нового акаунту до серверу бази даних. Після успішної реєстрації можна увійти до свого акаунта за допомогою електронної пошти та паролю котру вказували при реєстрації.

2.2.2 Додавання нових категорії та редагування вже існуючих

Для додавання нової категорії чи зміни все існуючої на головній сторінці мобільного додатку потрібно відкрити бокове меню и натиснути кнопка “Категорії”. Після цього користувач потрапляє на екран де відображаються всі існуючі категорії доходів та витрат і порожні категорії. Натиснув на існуючу категорію користувач буде перенесений на екран редагування категорій. Натиснув на порожню категорію користувач потрапить до екрану створення нової категорії. Заповнивши всі поля буде створений запит до бази даних для додавання чи зміни категорії

2.2.3 Додавання операцій доходів чи витрат

Для додавання нових доходів чи витрат потрібно натиснути на одну з категорій на головній сторінці, або на кнопку “+” для доходів і на кнопку “-” для витрат. Після цього відкриється екран додавання нових доходів чи витрат. Заповнивши суму та вибравши якщо потрібно категорію буде створений запит до бази даних для додавання нової операції.

2.2.4 Видалення операцій

Для видалення транзакції потрібно натиснути на кнопку історія на головному екрані. Вибрати потрібну операцію і пересунути її в будь-яку сторону. Після цього буде створений запит до бази даних для видалення потрібної операції.

2.3 Середовище розробки Android Studio

Android Studio є офіційним інтегрованим середовищем розробки (IDE) для операційної системи Android від Google, побудованої на програмному

забезпеченні IntelliJ IDEA від JetBrains і розробленої спеціально для розробки Android. Він доступний для завантаження в операційних системах на базі Windows, macOS і Linux. Це заміна Eclipse Android Development Tools (ADT) в якості основної IDE для розробки власних додатків Android [10].

Android Studio підтримує велику кількість мов програмування, наприклад, Java, C ++, Go. Android Studio 3.0 та пізнішими версіями, підтримують Kotlin і всі можливості мови Java 7 і підмножина мовних функцій Java 8, які відрізняються в залежності від версії платформи [10].

Основними особливостями Android Studio є:

– Підтримка побудови на основі Gradle
– Рефакторінг для Android та швидкі виправлення
– Шаблони основних макетів и компонентів Android.
– Розширений редактор макетів: WYSIWYG, здатність працювати з UI-компонентами при допомозі Drag-and-Drop, функція попереднього перегляду макету на декількох конфігураціях екрана.
– Статичний аналізатор коду (Lint), що дозволяє знаходити проблеми продуктивності, несумісності версій і інше.
– Вбудована підтримка Google Cloud Platform, яка включає в себе інтеграцію з сервісами Google Cloud Messaging і App Engine.
– Android Virtual Device (емулятор) для запуску та налагодження програм у студії Android.
– Підтримка розробки додатків для Android Wear і Android TV.

При розробці програми на Android корисно зрозуміти загальний підхід платформи до управління змінами API. Також важливо розуміти ідентифікатор рівня API і роль, яку воно відіграє в забезпеченні сумісності вашого застосування з пристроями, на яких воно може бути встановлено.

Рівень API - це цілочислове значення, однозначно ідентифікує версію API фреймворка, пропоновану версією платформи Android. Платформа Android надає інтерфейс API, який додатки можуть використовувати для взаємодії з базовою системою Android. API фреймворка складається з:

– Основний набір пакетів і класів
– Набір XML-елементів та атрибутів для оголошення файлу маніфесту
– Набір елементів і атрибутів XML для оголошення і доступу до ресурсів
– Набір Интентів
– Набір дозволів, які можуть запитувати додатки, а також примусове застосування дозволів, включених в систему

Кожна наступна версія платформи Android може містити поновлення API платформи додатків Android, які вона надає.

Оновлення API-інтерфейсу платформи розроблені таким чином, щоб новий API залишався сумісним з більш ранніми версіями API. Тобто більшість змін в API є адитивними і вводять нові або замінюють функціональні можливості. У міру відновлення частин API застарілі замінені частини застарівають, але не видаляються, тому існуючі програми все ще можуть їх використовувати. У дуже невеликій кількості випадків частини API можуть бути змінені або видалені, хоча зазвичай такі зміни необхідні тільки для забезпечення надійності API і безпеки програми або системи. Всі інші частини API з попередніх версій переносяться без змін.

API-інтерфейс платформи, що надається платформою Android, вказується за допомогою цілочислового ідентифікатора, званого «Рівень API». Кожна версія платформи Android підтримує рівно один рівень API, хоча підтримка неявна для всіх попередніх рівнів API (аж до рівня API 1). Початковий випуск платформи Android надав API Рівень 1, а наступні випуски збільшили рівень API.

У наступній таблиці зазначений рівень API, підтримуваний кожною версією платформи Android [11].

Таблиця 2.1 – Відповідність версії платформи та рівня API

Версія платформи	Рівень API
Android Q Beta	29
Android 9.0	28

Версія платформи	Рівень API
Android 8.0	26
Android 7.0	24
Android 6.0	23
Android 5.0	21
Android 4.4	19
Android 4.0, 4.0.1, 4.0.2	14
Android 3.0.x	11
Android 2.0	5
Android 1.0	1

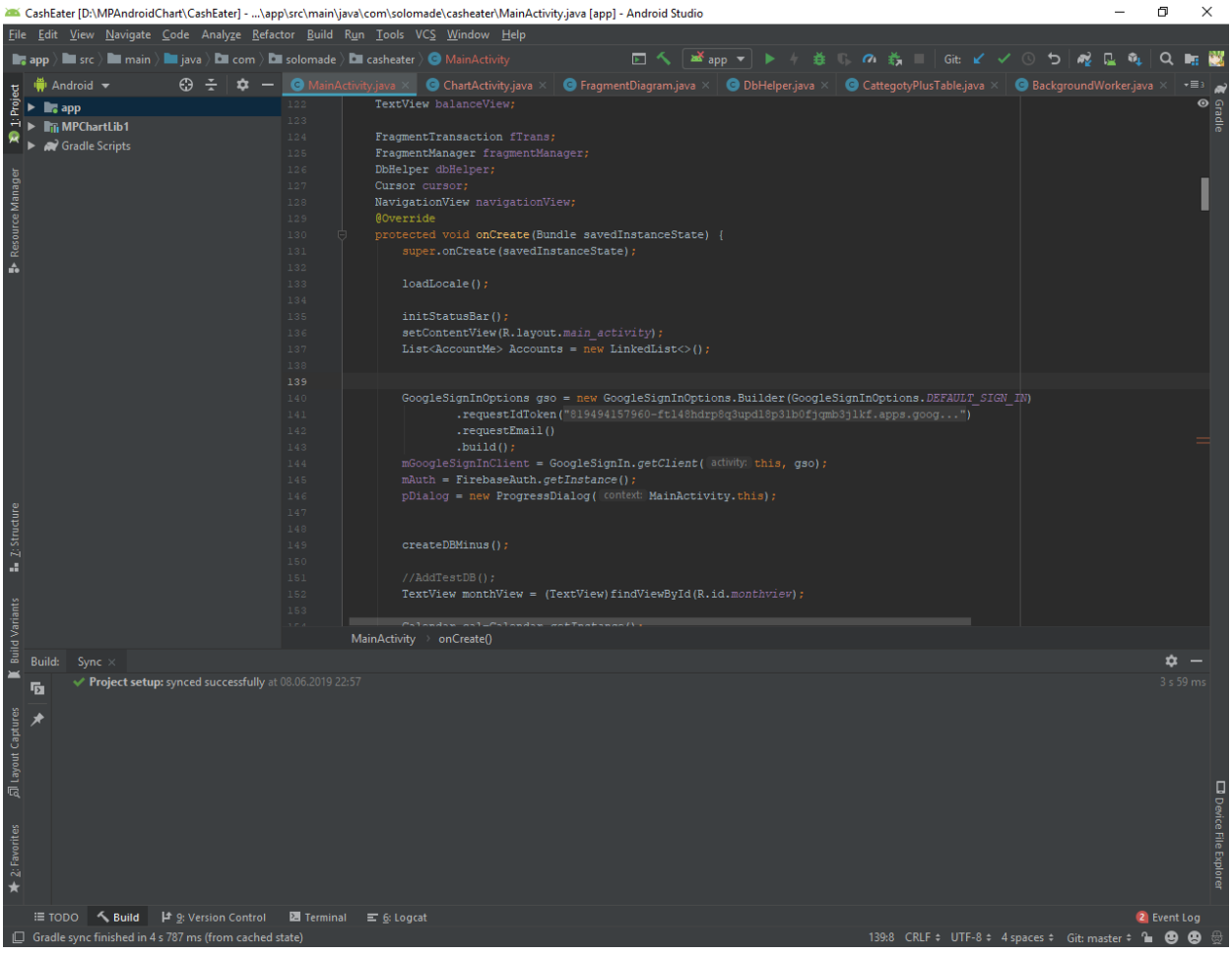


Рисунок 2.3 – Знімок екрану Android studio з відкритим проектом

2.4 Розробка програмного забезпечення

Розробка додатку для операційної системи Android в середовищі розробки Android Studio поділяється на дві частини. Перша частина це верстка екранів за допомогою мови розмітки XML. Друга частина написання всієї логіки додатку за допомогою мови програмування Java.

XML (EXtensible Markup Language) – розширювана мова розмітки. Рекомендований Консорціумом Всесвітньої павутини (W3C). XML розроблявся як мова з простим формальним синтаксисом, зручний для створення і обробки документів програмами і одночасно зручний для читання і створення документів людиною, з підкресленням націленості на використання в Інтернеті. Мова називається розширюваною, оскільки ним не фіксується розмітку, яка використовується в документах: розробник вільний створити розмітку відповідно до потреб конкретної області, будучи обмеженим лише синтаксичними правилами мови [12].

Java — об'єктно-орієнтована мова програмування, випущена 1995 року компанією “Sun Microsystems” як основний компонент платформи Java. В офіційній реалізації Java-програми компілюються у байт-код, який при виконанні інтерпретується віртуальною машиною для конкретної платформи.

Мова значно запозичила синтаксис із C і C++. Зокрема, взято за основу об'єктну модель C++, проте її модифіковано. Усунуто можливість появи деяких конфліктних ситуацій, що могли виникнути через помилки програміста та полегшено сам процес розробки об'єктно-орієнтованих програм. Ряд дій, які в C/C++ повинні здійснювати програмісти, доручено віртуальній машині. Передусім Java розроблялась як платформо-незалежна мова, тому вона має менше низькорівневих можливостей для роботи з апаратним забезпеченням, що в порівнянні, наприклад, з C++ зменшує швидкість роботи програм. За необхідності таких дій Java дозволяє викликати підпрограми, написані іншими мовами програмування.

2.4.1 Підключення до серверу бази даних

Для підключення до серверу бази даних набув створений скрипт до якого буде звертатися додаток. Скрипт пишеться на мові PHP.

PHP (Hypertext Preprocessor) - Інструменти для створення персональних веб-сторінок - скриптова мова загального призначення, інтенсивно застосовується для розробки веб-додатків. В даний час підтримується переважною більшістю хостинг-провайдерів і є одним з лідерів серед мов, що застосовуються для створення динамічних веб-сайтів [15].

Підключення до скрипта здійснюється за допомогою класу HttpURLConnection через URL адресу серверу. Після вдалого підключення на сервер відсилаються дані для перевірки справжності користувача. При вдалій перевірці локальні дані та дані з серверу обновлюються до актуальних.

При невдалій перевірці користувачу буде запропоновано перевірити правильність введених даних або зареєструватися.

При реєстрації додаток теж буде звертатися до скрипту на сервері але вже до іншого. Цей скрипт бере дані с форми реєстрації и додає їх до серверу бази даних.

2.4.2 Діаграма витрат

MPAndroidChart - це потужна і проста у використанні бібліотека для Android. Яка позволяє створювати різні види діаграм. MPAndroidChart має безліч налаштувань. Таким чином можна створити унікальну діаграму яка підійде вашому проекту.

Для зручного відображення даних про витрати на головній сторінки була обрана кругова діаграма. Щоб діаграма відображалась потрібно звернутися до бази даних і отримати всі стовпці таблиці Transactions та знайти суму котра була витрачена на певну категорію.

Далі потрібно звернутися до таблиці Categories, щоб отримати колір категорії та її назву. Отримавши всі потрібні дані можна відображувати діаграму.

Рисунок 2.4 – Діаграма витрат

Також на діаграму встановлений відстежувач дотику. За допомогою цієї функції, яка використовує під час роботи звичайну систему координат на основі розміру екрана, система відстежує місце дотику користувача, після чого виділяє потрібну категорію.

2.4.3 Графік доходів та витрат

За допомогою бібліотеки MPAndroidChart також були побудовані графіки доходів та витрат. Для відображення графіки спершу відбувається звернення до таблиці Transactions бази даних та отримання стовпців з сумою операції та датою її додавання. Після цього ці данні упорядковуються по даті додавання і в залежності від кількості днів сортируються по дням, тижням або місяцям.

Якщо в базі даних немає інформації для побудови графіків то замість графіку відображається попередження про відсутність даних.

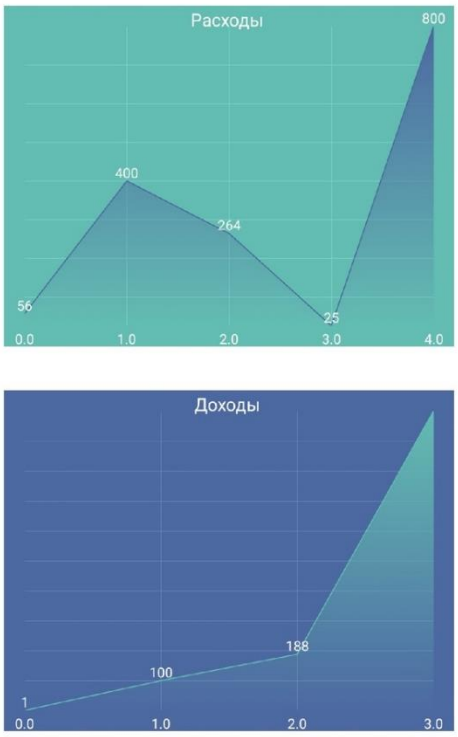


Рисунок 2.5 – Графік доходів та витрат

2.4.4 Синхронізація даних з сервером бази даних

Після запуску, додаток перевіряє чи є підключення до інтернету і увійшов користувач в свій акаунт. Після вдалої перевірки додаток порівнює стовпці локальної бази з базою даних на сервері. В першу чергу перевіряється дата поновлення даних. Якщо дата не збігається дані обновляються до тих, що є більш новими судячи з дати.

Якщо при порівнянні стовпців аналога на локальній базі або в базі даних серверу не було виявлено, то ці дані додаються відповідно.

Також при зміні або додаванню нових даних на локальній базі вони також змінюються або додаються на сервері бази даних за умови, що користувач має підключення до інтернету и увійшов у свій акаунт.

При видаленні даних на мобільному пристрої без підключення інтернету вони видаляться на локальному пристрої, але після підключення до інтернету вони видаляться і на сервері бази даних

2.4.5 Редагування та створення нових категорій

На екрані з відображення всіх категорій кожній кнопці присвоєно тег з унікальним кодом категорії котрий отримується з таблиці бази даних Categories. Якщо категорія пуста їй присвоюється тег -1 для категорій доходу і -2 для категорій витрат. Після натискання на одну з категорій тег передається до іншого класу де виконується пошук в базі даних потрібної категорії по цьому тегу для подальшої зміни. Якщо тег -1 або -2 буде створена нова категорія.

Висновок до розділу 2

У 2 розділі було розглянуто архітектура системи, процес проектування бази даних, описані основні моменти взаємодії користувача з додатком. Було описано середовище розробки та основні моменти розробки.

3 ІНТЕРФЕЙС КОРИСТУВАЧА

У Розділі надана інформація про розробку інтерфейсу. Представлені результати розробки інтерфейсу. Описаний інтерфейс користувача.

3.1 Розробка інтерфейсу

Кожен екран, що бачить користувач в додатку називається Activity. Зазвичай вони складаються з XML тегів.

Activity - це компонент додатка, який видає екран, і з яким користувачі можуть взаємодіяти для виконання будь-яких дій, наприклад набрати номер телефону, зробити фото, відправити лист або переглянути карту. Кожній операції присвоюється вікно для промальовування відповідного призначеного для користувача інтерфейсу. Зазвичай вікно відображається на весь екран, проте його розмір може бути менше, і воно може розміщуватися поверх інших вікон.

Як правило, програма містить кілька операцій, які слабо пов'язані один з одним. Зазвичай одна з операцій в додатку позначається як «основна», пропонована користувачеві при першому запуску програми. У свою чергу, кожна операція може запустити іншу операцію для виконання різних дій. Кожен раз, коли запускається нова операція, попередня операція зупиняється, однак система зберігає її в стек («стек переходів назад»). При запуску нової операції вона поміщається в стек переходів назад і нова операція відображається для користувача. Стек переходів назад працює за принципом «останнім увійшов - першим вийшов», тому після того як користувач завершив поточну операцію і натиснув кнопку Назад, поточна операція видаляється з стека (і знищується), і відновлюється попередня операція [12].

Активність має життєвий цикл - початок, коли Android створює екземпляр активності, проміжний стан, і кінець, коли екземпляр знищується системою і звільняє ресурси. Активність може перебувати в трьох станах:

- активна (active або running) - активність знаходиться на передньому плані екрана. Користувач може взаємодіяти з активним вікном;
- призупинена (paused) - активність втратила фокус, але все ще видима користувачу. Тобто активність знаходиться зверху і частково перекриває іншу активність. Призупинена активність може бути знищена системою в критичних ситуаціях при нестачі пам'яті;
- зупинена (stopped) - якщо ця діяльність повністю закрита іншою активністю. Вона більше не видима користувачу і може бути знищена системою, якщо пам'ять необхідна для більш важливого процесу.

Якщо активність, яка була знищена системою, потрібно знову показати на екрані, вона повинна бути повністю перезапущена і відновлена в своєму попередньому стані.

Для більшої гнучкості та динамічності інтерфейсу користувача були використані фрагменти

Фрагмент (клас Fragment) представляє поведінку або частина призначеного для користувача інтерфейсу в операції (клас Activity). Розробник може об'єднати кілька фрагментів в одну операцію для побудови багато панельного, призначеного для користувача інтерфейсу і повторного використання фрагмента в декількох операціях. Фрагмент можна розглядати як модульну частина операції. Така частина має свій життєвий цикл і самостійно обробляє події введення. Крім того, її можна додати або видалити безпосередньо під час виконання операції. Це щось на зразок вкладеної операції, яку можна багаторазово використовувати в різних операціях [13].

Фрагмент завжди повинен бути вбудований в операцію, і на його життєвий цикл безпосередньо впливає життєвий цикл операції. Наприклад, коли операція припинена, в тому ж стані знаходяться і всі фрагменти в ній, і коли операцію знищується, знищуються і всі фрагменти. Однак поки операція виконується (це відповідає стану відновлено життєвого циклу), можна маніпулювати кожним фрагментом незалежно, наприклад додавати або видаляти їх [13].

Під час роботи над проектом було розроблено п’ять activity для відображення головного екрану, екрану додавання операцій, екрану відображення категорій, екрану додавання та зміни категорій, екрану відображення графіків. Також для зручності було розроблено декілька фрагментів.

Перед розробкою інтерфейсу був зроблений прототип. На рис 3.1 представлений прототип додавання нових категорій та відкриття історії цих операцій.

Під номером 1 зображений головний екран додатку. Після натискання на одну з двох кнопок знизу екрану користувач потрапляє до пункту під номером 2 який представляє собою екран введення суми для нової операції. Далі при натисканні кнопки “вибір категорії” користувач потрапляє до пункту під номером 3. Це екран вибору категорії для нової операції. Після вибору одної з категорії користувач знову потрапляє до пункту номер 1.

Для перегляду історії операції на зображенні номер 1 користувач повинен натиснути кнопку “Історія”. Після цього користувач потрапляє до пункту номер 4. Для потрапляння до пункту 1 користувач повинен знову натиснути кнопку “Історія”.

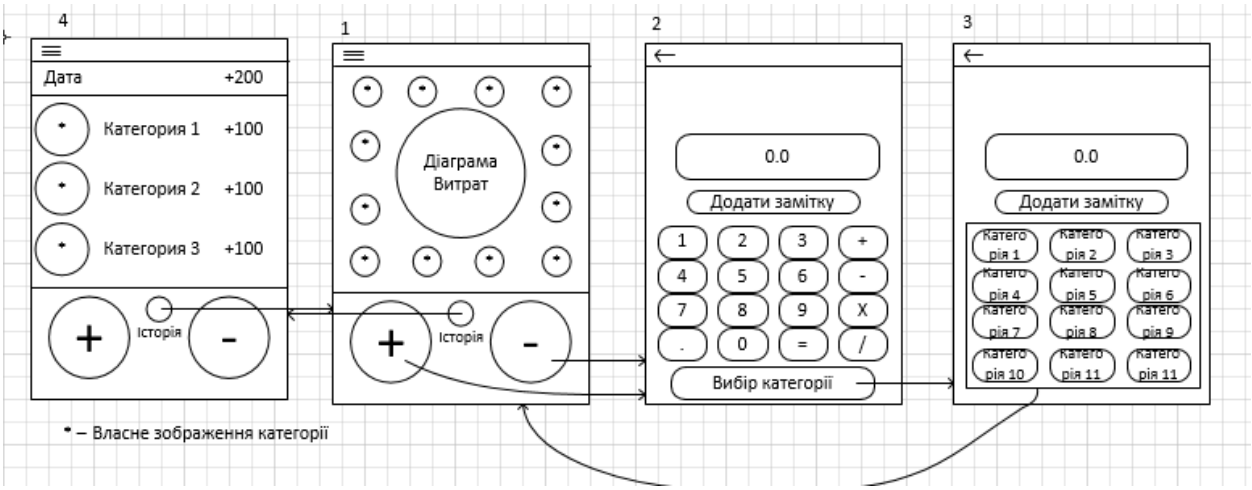


Рисунок 3.1 – Прототип додавання операції та відкриття історії

3.2 Головний екран

Головний екран це екран, що бачить користувач запустивши додаток (рис 3.2). Він містить у собі основну інформацію про витрати користувача. Більшу частину екрану займає кругова діаграма яка відображає усі витрати користувача. Всі категорії на діаграмі мають свій колір, щоб користувач міг швидку орієнтуватися.

Рисунок 3.2 – Головний екран додатку

В середині діаграми відображається зеленим кольором загальна сума доходів та червоним загальна сума витрат.

Навколо діаграми відображаються всі доступні категорії витрат кожна категорія має своє зображення та колір. Якщо категорії не вистачає, щоб заповнити простір навколо діаграми він заповнюється кнопками додавання нової категорії.

При натисканні на будь яку категорії навколо діаграми можна перейди одразу до екрану додавання операції. Також якщо натиснути та тримати на будь яку категорію або на колір категорії на діаграмі можна побачити у середині діаграми назву категорії та кошти витрачені на неї.

Фрагмент програмного коду відстеження натискання на діаграму.

```
if((Math.pow(event.getX()-hart.getCenter().getX(),2))+
(Math.pow(event.getY()-chart.getCenter().getY(),2))
<=Math.pow((chart.getRadius()*(chart.getHoleRadius() / 100)),2)){break;}
else {Highlight high = chart.getHighlightByTouchPoint(event.getX(),
event.getY());
if(high == null)break;
chart.highlightValue(high);
chart.setCenterText(str.get((int)high.getX())+"\n"+high.getY());
chart.setHoleColor(dataSet.getColor((int)high.getX()));
chart.setCenterTextColor(getResources().getColor(R.color.main_color_white
));
break;}
```

Під діаграмою відображається поточний баланс користувача та три великі кнопки. Кнопки з зображенням плюсу та мінусу слугують для переходу на екран додавання нової операції.

Кнопка з написом “История” потрібна для відображення історії операцій користувача (рис 3.3).

Фрагмент програмного коду відображення історії.

```
TransactionAdapter.setDataOneItem(Transactions);
if (Transactions.size() != 0) {
ArrayList<Float> sum_day = new ArrayList<>();
sum_day.add(Transactions.get(0).getSum());
String sd = Transactions.get(0).getData();
for (int i = 0; i < Transactions.size(); i++) {
if (i == 0) { }
else if (sd.equals(Transactions.get(i).getData())) {
sum_day.set(i - 1, sum_day.get(i - 1) + Transactions.get(i).getSum());
Transactions.remove(i);
i--;}
else {
sd = Transactions.get(i).getData();
sum_day.add(Transactions.get(i).getSum());}}
TransactionAdapter.notifyDataSetChanged();
```

```
TransactionAdapter.setData(Transactions, sum_day); }
```

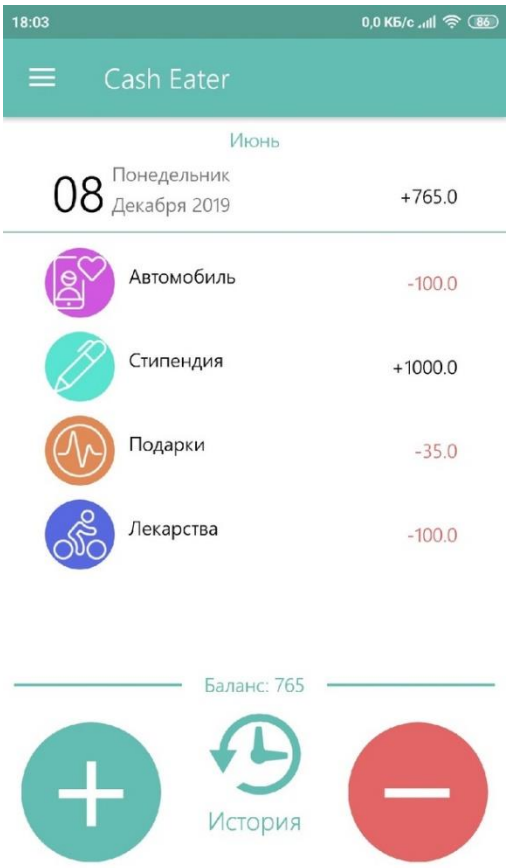


Рисунок 3.3 – Історія операцій

Історія транзакцій змінює собою діаграму та категорії на головній сторінці. Вона відображує всі проведені операції користувача по дням, сумуючи значення фінансових витрат та прибутків за відповідний день. Для видалення обраної користувачем операції потрібно натиснути на рядок з транзакцією та перетягнути у будь який бік. Після цього з’явиться повідомлення для підтвердження операції, на яке потрібно натиснути для видалення запису. Такий простий механізм видалення дозволяє уникнути помилки під час роботи з програмою.

У лівому верхньому куті екрану знаходиться кнопка для виклика бокового меню, яке відповідає за системні налаштування додатку (рис 3.4). У даному боковому меню знаходиться кнопка зміни мови, переходу до екрану редагування всіх категорій, відображення статистичної інформації у виді графіків та кнопка для авторизації облікового запису користувача.

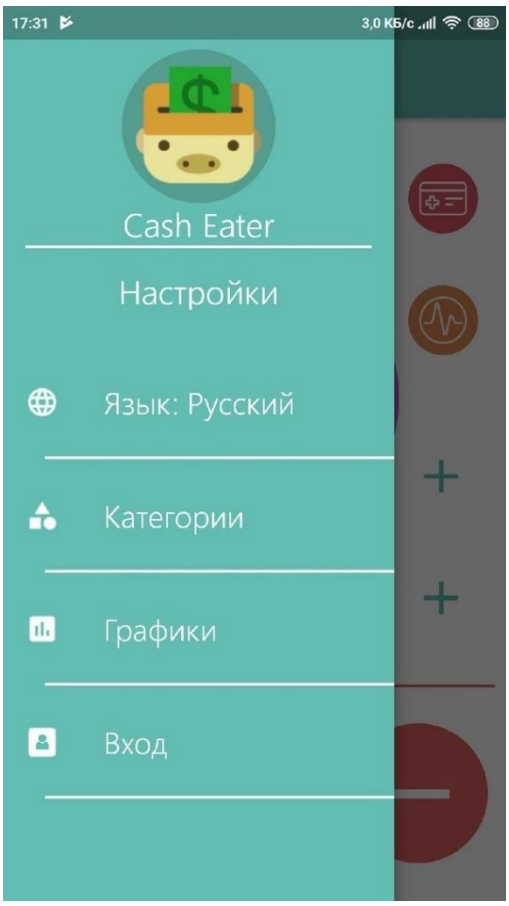


Рисунок 3.4 – Бокове меню додатку

3.3 Зміна мови додатку

Для охоплення більшої аудиторії в додатку присутня зміна мови. Багатомовність додатку створюється для відвідувачів, які розмовляють на різних мовах, навіть якщо вони живуть на території однієї країни. Так само для подальшого розвитку проекту та випуску його закордоном.

Фрагмент програмного коду зміни мови інтерфейсу.

```
Locale locale = new Locale(lang);
Locale.setDefault(locale);
Configuration configuration = new Configuration();
configuration.locale = locale;
getBaseContext().getResources().updateConfiguration(configuration,getBase
Context().getResources().getDisplayMetrics());
SharedPreferences.Editor
editor=getSharedPreferences("Settings",MODE_PRIVATE).edit();
editor.putString("My_lang",lang);
editor.apply();
```

На даний момент додаток підтримує три мови: українську, російську та англійську. Для зміни мови користувач повинен відкрити бокове меню та натиснути на перший пункт меню де буде відображатися поточна мова інтерфейсу додатку. При натисканні користувачу запропонує на вибір 3 мови (рис 3.5). Після вибору весь інтерфейс буде переведений на вибрану мову без перевантаження додатку.

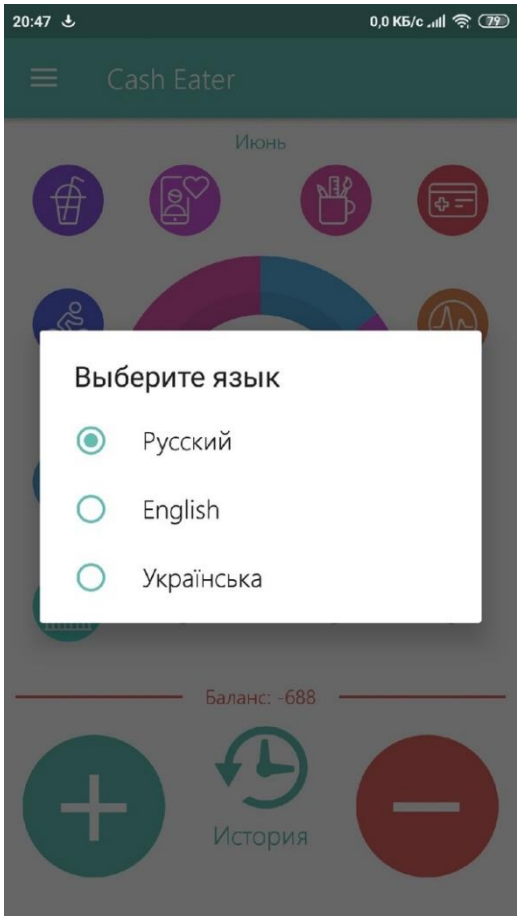


Рисунок 3.5 – Вікно вибору мови

3.4 Екран додавання операцій

Екран додавання операцій призначений для додавання доходів або витрат (рис 3.6). Екран містить у собі поле для відображення суми, що ввів користувач за допомогою інтегрованого калькулятору. Також поле з відображення суми має кнопку для видалення останньої дії будь то цифра чи якась арифметична операція . Поле для додавання заміток яке не є обов’язкове для заповнення.

Під калькулятором знаходиться кнопка для вибору категорій. При натисканні якої знизу екрану виїжджає поле з категоріями. Відображаються категорії доходів чи витрат відповідно з тим яка кнопка на головному екрані була натиснута. Якщо була натиснута кнопка певної категорії то вибору категорії не буде, замість неї буде кнопка додавання операції.

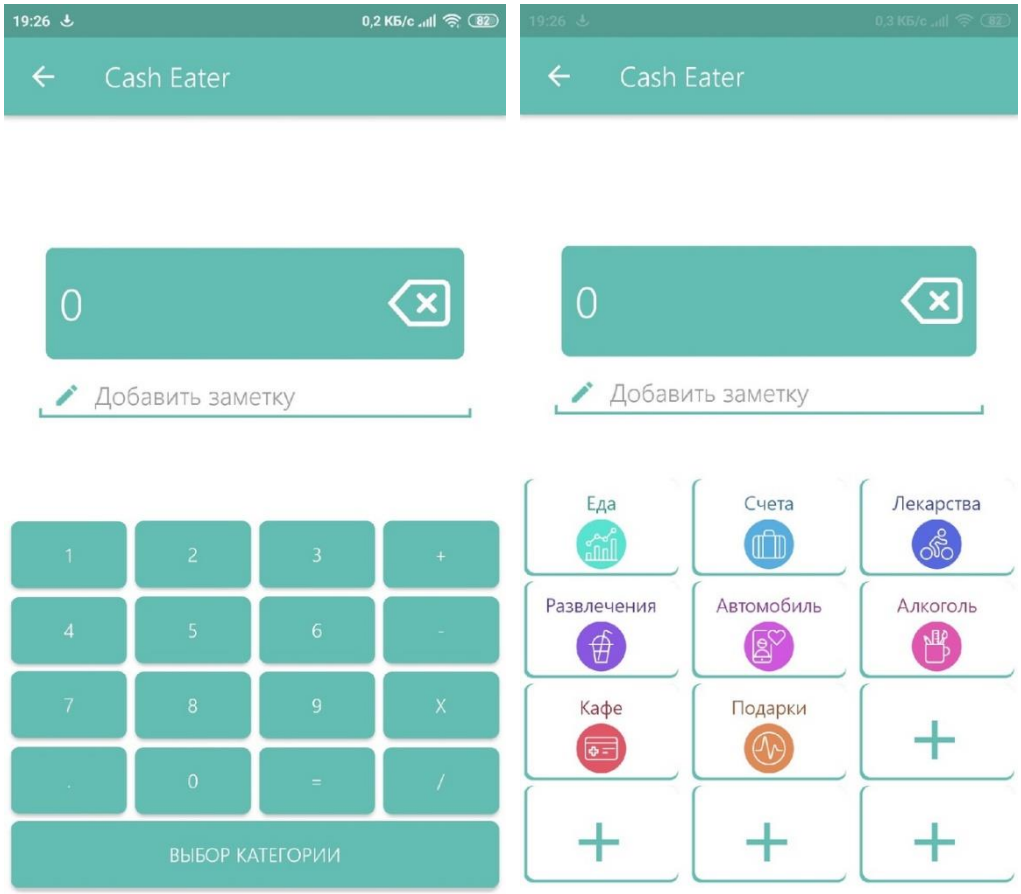


Рисунок 3.6 – Екран додавання операцій

Фрагмент програмного коду додавання нових операції.

```
DbHelper dbHelper = new DbHelper(AddTransactionActivity.this);
TransactionModel TransactionModel = new TransactionModel(dbHelper);
presenter = new TransactionPresenter(TransactionModel);
presenter.attachView(AddTransactionActivity.this);
cat_text_btn = cat_name;
if(presenter.add(tag_btn) == -1){}
else finish();
```

Якщо для операції, що ви хочете додати ще не існує категорії можна натиснути на кнопку с зображення плюсу та перейти до екрану додавання нової категорії.

При спробі додати операцію без суми вона не буде додана. Користувач буде попереджений про відсутність суми та йому буде запропоновано повторити спробу додати нову операцію.

3.5 Екран відображення графіків

Для зручності відображення доходів та витрат користувача на довгій дистанції були додані графіки. Для переходу до графіків користувач мусить відкрити бокове меню та перейти до пункту “Графики”. Після цього користувач потрапляє до екрану де відображаються два графіки один для доходів другий для витрат (рис 3.7).

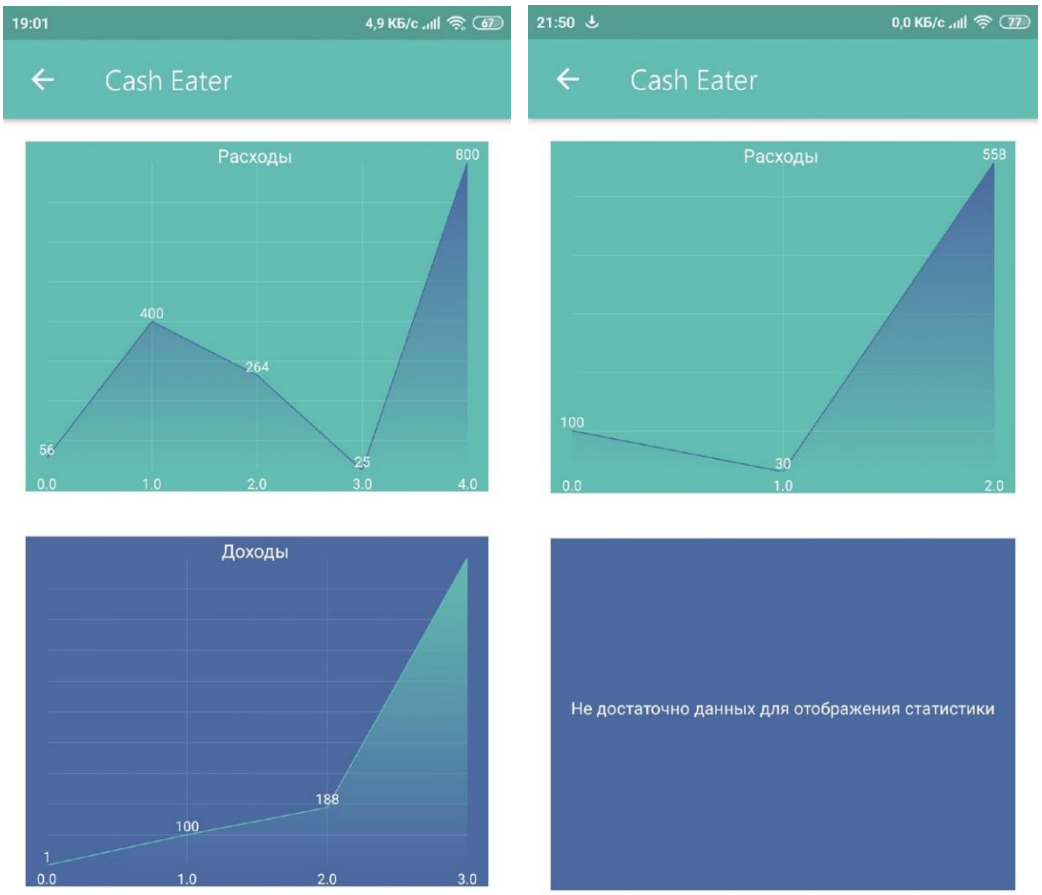


Рисунок 3.7 – Екран відображення графіків

Якщо для відображення графіку не вистачає даних то замість графіку буде показано попередження про те, що даних не вистачає.

Фрагмент програмного коду запиту до бази даних для відображення графіків

```
DbHelper dbHelper = new DbHelper(this);
ArrayList<Float> transaction = new ArrayList<>();
transactionAll = new ArrayList<>();
Cursor
cursor=dbHelper.getReadableDatabase().query(TransactionTable.TABLE,null,null,
null,null,null,null);
while (cursor.moveToNext()) {
transactionAll.add(cursor.getString(cursor.getColumnIndex(TransactionTabl
e.COLUMN.DATA)));
transaction.add(cursor.getFloat(cursor.getColumnIndex(TransactionTable.CO
LUMN.SUM)));}
```

3.6 Екран відображення категорій

При натисканні кнопки “Категории” у боковому меню користувач потрапляє до екрану де відображаються всі категорії (рис 3.8). Категорії поділені на групи спершу відображаються категорії витрат, а потім категорії доходів.

При натисканні на категорію користувач буде перенесений до екрану редагування категорій.

Фрагмент програмного коду відстеження натиску на категорію.

```
View.OnClickListener update_cat_click = new View.OnClickListener() {
@Override
public void onClick(View v) {
Intent myIntent = new Intent(update_cat_activity.this,
update_one_cat_activity.class);
String srt = v.getTag().toString();
myIntent.putExtra("id_cat", v.getTag().toString());
update_cat_activity.this.startActivity(myIntent); }};
```

Рисунок 3.8 – Екран відображення категорій

3.7 Екран редагування та додавання категорій

Екран редагування та додавання представляє собою один екран. При переході до редагування категорії всі поля на екрані редагування вже будуть вибрані, а при додаванні нової категорії всі поля будуть пусті (рис 3.9).

На цьому екрані відображається поле для введення назви категорії. Нижче представлено кнопка яка відкриває вікно з вибором кольору для категорії (рис 3.10). Все інше місце займає різні зображення категорії. Біля назви категорії є кнопка у вигляді плюсу яка зберігає зміни чи додає нову категорію.

Вікно з вибором кольору для категорій має на вибір п'ятнадцять кольорів. При натисканні на один із кольорів він буде позначений галкою. Знизу вікна знаходяться дві кнопки “Принять” яка зберігає вибраний колір та кнопка “Отмена” яка відміняє зміни кольору.

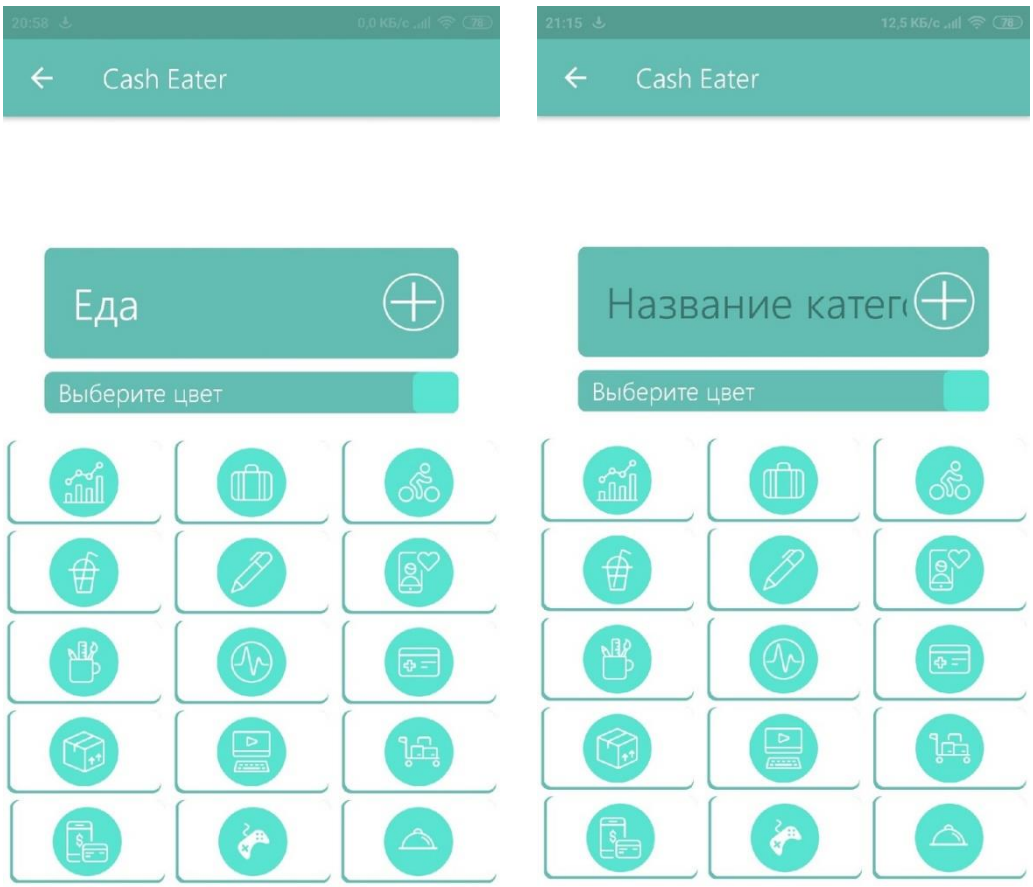


Рисунок 3.9 – Екран редагування та додавання категорій

Фрагмент програмного коду обновлення категорії.

```
if(id_cat.equals("-2") || id_cat.equals("-1")){
if(id_cat.equals("-2")){ cvAccount.put("PLUSORMINUS",-1);}
else { cvAccount.put("PLUSORMINUS",1); }
dbHelper.getWritableDatabase().insert(CategoryTable.TABLE,null,cvAccount)
;}else{
dbHelper.getWritableDatabase().update(CategoryTable.TABLE,cvAccount,"_id
= '"+id_cat+"' ",null);
dbHelper.getWritableDatabase().update(TransactionTable.TABLE,cvTrans,"cat
egory = '"+old_text_cat+"' ",null);}
```

Якщо при її натисканні користувач не вибрав колір то категорія не буде додана, а користувач буде попереджений про відсутність кольору. Також якщо користувач не вибрав зображення йому буде видано попередження про відсутність вибору зображення.

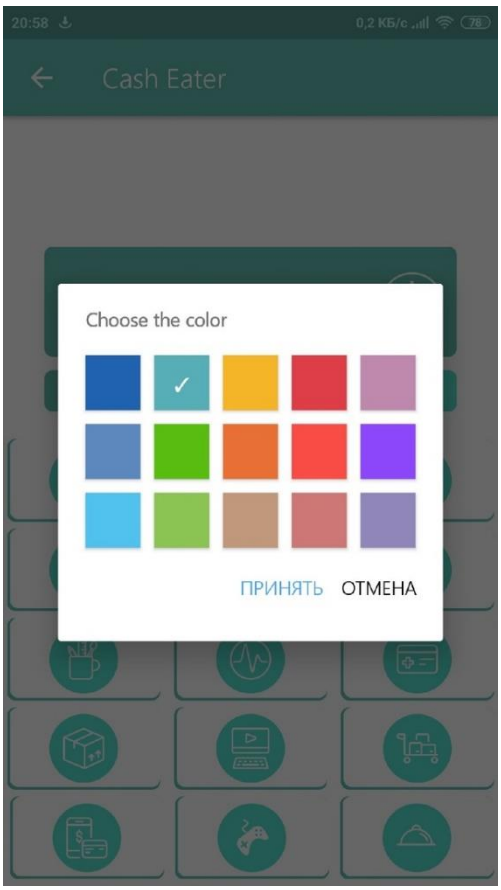


Рисунок 3.10 – Вікно зміни кольору

3.8 Вікно входу до акаунту

Для зберігання своїх даних в онлайн базі даних користувач повинен увійти до свого акаунту. Для входу до свого акаунту користувач повинен відкрити бокове меню та натиснути кнопку “Вход”. Після цього з’явиться вікно для входу у свій акаунт (рис 3.11).

Вікно складається з двох полів для вводу електронної адреси та паролю в яких користувачу дається підказка яку інформацію він має ввести та двох кнопок для входу та відкриття вікна реєстрації.

Після правильного заповнення полів для входу у свій акаунт користувач буде перенесений на головну сторінку.

Якщо ж поля були заповнені невірно то користувач отримає попередження про неправильність вводу даних та йому буде запропоновано повторити спробу входу або зареєструватися у системі.

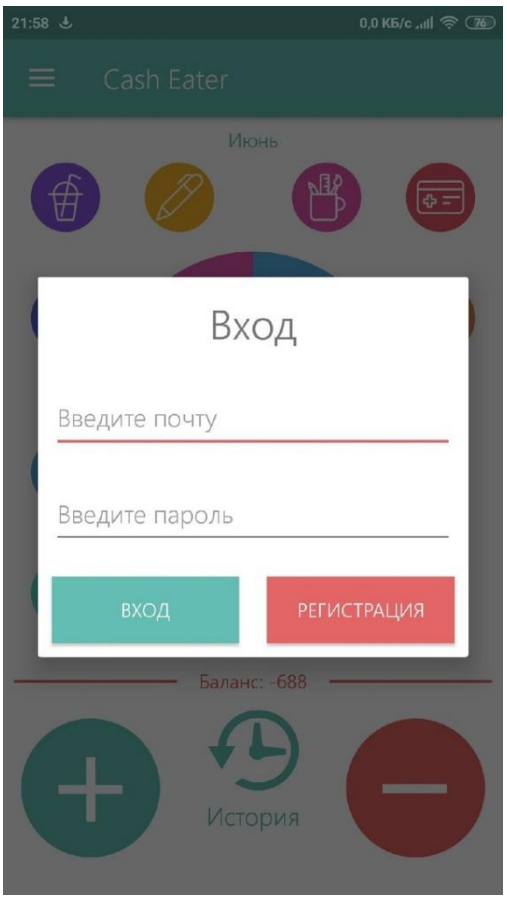


Рисунок 3.11 – Вхід до акаунту користувача

Фрагмент програмного коду запиту до бази даних для входу у акаунт

```
String user_mail = params[1];
String password = params[2];
URL url = new URL(login_url);
HttpURLConnection httpURLConnection = (HttpURLConnection)url.openConnection();
httpURLConnection.setRequestMethod("POST");
httpURLConnection.setDoInput(true);
httpURLConnection.setDoOutput(true);
OutputStream outputStream = httpURLConnection.getOutputStream();
BufferedWriter bufferedWriter = new BufferedWriter(new OutputStreamWriter(outputStream, StandardCharsets.UTF_8));
String post_data = URLEncoder.encode("user_mail","UTF-8")+"="+URLEncoder.encode(user_mail,"UTF-8")+"&"
+URLEncoder.encode("password","UTF-8")+"="+URLEncoder.encode(password,"UTF-8");
bufferedWriter.write(post_data);
bufferedWriter.flush();
bufferedWriter.close();
outputStream.close();
```

3.9 Вікно реєстрації нового користувача

Для реєстрації нового акаунту користувач має відкрити бокове меню та натиснути кнопку “Вход”. Після відкриття вікна входу користувач повинен натиснути кнопку “Регистрация” для відкриття вікна реєстрації (рис 3.12).

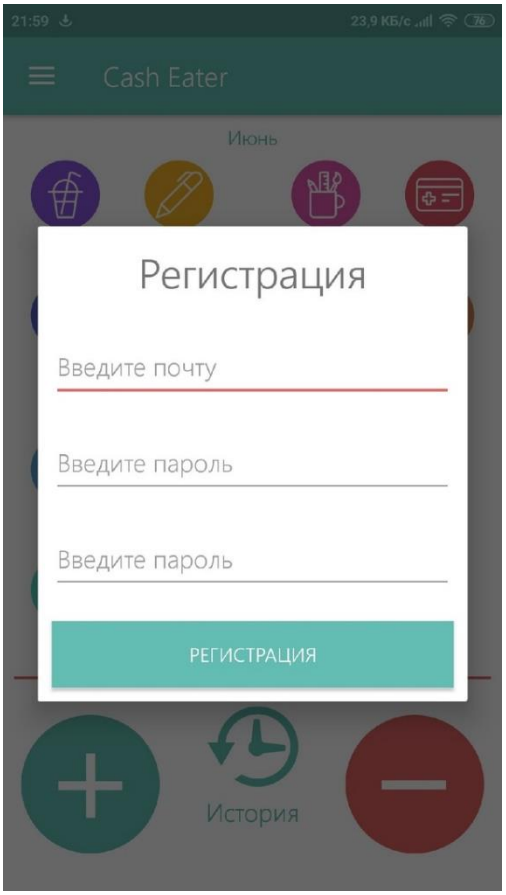


Рисунок 3.12 – Вікно реєстрації користувача

Вікно реєстрації має три поля для введення даних. Поле введення електронної пошти, поле для введення паролю та поле для повторного введення паролю. В кожному полі користувачу дається підказка яку інформацію він має ввести. Також вікно має кнопку “Регистарация” при натисканні якої у випадку якщо користувач заповнив всі поля коректною інформацією він буде зареєстрований.

Фрагмент програмного коду запиту до бази даних для входу у акаунт

```
String user_mail = params[1];
String password = params[2];
```

```
    String passwordRepeated= params[3];
    URL url = new URL(login_url);
    HttpURLConnection                    httpURLConnection                    =
(HttpURLConnection)url.openConnection();
    httpURLConnection.setRequestMethod("POST");
    httpURLConnection.setDoInput(true);
    httpURLConnection.setDoOutput(true);
    OutputStream outputStream = httpURLConnection.getOutputStream();
    BufferedWriter    bufferedWriter        =    new    BufferedWriter(new
OutputStreamWriter(outputStream, StandardCharsets.UTF_8));
    String    post_data    =    URLEncoder.encode("user_mail","UTF-8")+"="+
URLEncoder.encode(user_mail,"UTF-8")+"&"
    +URLEncoder.encode("password","UTF-8")+"="+
URLEncoder.encode(password,"UTF-8")+"&"
    +URLEncoder.encode("passwordRepeated","UTF-8")+"="+
URLEncoder.encode(passwordRepeated,"UTF-8");
    bufferedWriter.write(post_data);
    bufferedWriter.flush();
    bufferedWriter.close();
    outputStream.close();
```

Якщо користувач введе інформацію неправильно або взагалі залишить поля пустими йому буде показано попередження і прохання заповнити поля коректною інформацією

Висновок до розділу 3

У розділі була представлена інформація по розробці інтерфейсу користувача. Наведені результати розробки та короткий опис можливостей інтерфейсу.

4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

В даному розділі було розглянуто загальні питання з охорони праці. Були проаналізовані умови праці, вимоги до приміщення та організації. Розглянуті заходи, які дозволяють забезпечити гігієну праці та виробничу санітарію. Також були розглянуті рекомендації щодо пожежної безпеки, електробезпеки, мікроклімату та освітлення.

4.1 Загальні питання з охорони праці

Згідно з законом “Про охорону праці” [16] охорона праці це – система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження життя, здоров'я і працездатності людини у процесі трудової діяльності.

При роботі з обчислювальною технікою змінюються фізичні і хімічні фактори навколишнього середовища: виникає статична електрика, електромагнітне випромінювання, змінюється температура і вологість, рівень вміст кисню і озону в повітрі. Забезпечення цих умов покладається на власника або уповноважений ним орган (далі роботодавець). Умови праці на робочому місці, безпека технологічних процесів, машин, механізмів, устаткування та інших засобів виробництва, стан засобів колективного та індивідуального захисту, що використовуються працівником, а також санітарно-побутові умови повинні відповідати вимогам нормативних актів про охорону праці.

4.2 Аналіз стану та умов праці

Для роботи над створенням мобільного додатку “Cash Eater” достатньо однієї людини для якої надано робоче місце ці стаціонарним комп’ютером.

4.2.1 Вимоги до приміщення

Геометричні розміри приміщення зазначені в табл. 4.1.

Таблиця 4.1 – Розміри приміщення

Найменування	**Значення**
Довжина, м	6
Ширина, м	3
Висота, м	3
Площа, $м^2$	18
Об’єм, $м^3$	54

Згідно з ДСН 3.3.6.042-99 “Санітарні норми мікроклімату виробничих приміщень” [17] розмір площі для одного робочого місця оператора персонального комп’ютера має бути не менше 6 кв. м, а об’єм — не менше 20 куб. м. Отже, дане приміщення цілком відповідає зазначеним нормам.

Також для дотримання визначеного рівня мікроклімату в будівлі встановлено систему опалення та кондиціонування.

Для забезпечення потрібного рівного освітленості кімната має вікно та систему загального рівномірного освітлення, що встановлена на стелі. Для дотримання вимог пожежної безпеки встановлено порошковий вогнегасник та систему автоматичної пожежної сигналізації.

4.2.2 Вимоги до організації робочого місця

При порівнянні відповідності характеристик робочого місця к нормативним, основні вимоги до організації робочого місця за ДСанПіН 3.3.2.007-98 «Правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» [18] і відповідними

фактичними значеннями для робочого місця, констатуємо повну відповідність в таблиці 4.2.

Таблиця 4.2 - Характеристики робочого місця

Найменування параметра	**Фактичне значення**	**Нормативне значення**
Висота робочої поверхні, мм	700	680 ÷800
Висота простору для ніг, мм	680	не менше 600
Ширина простору для ніг, мм	550	не менше 500
Глибина простору для ніг, мм	700	не менше 650
Висота поверхні сидіння, мм	500	400 ÷500
Ширина сидіння, мм	400	не менше 400
Глибина сидіння, мм	400	не менше 400
Висота поверхні спинки, мм	600	не менше 300
Ширина опорної поверхні спинки, мм	400	не менше 380
Радіус кривини спинки в горизонтальній площині, мм	400	400
Відстань від очей до екрану дисплея, мм	800	700 ÷800

Приміщення кабінету знаходиться у вчасній одноповерховій будівлі і має об’єм 54 м3, площу — 18 м2.

Температура в приміщенні протягом року коливається у межах 18–24°С, відносна вологість — близько 50%. Система вентилювання приміщення — природна неорганізована, а опалення — централізоване.

Розміщення вікон забезпечує природне освітлення з коефіцієнтом природного освітлення не менше 1,5%, а загальне штучне освітлення, яке здійснюється за допомогою однієї люмінесцентної лампи, забезпечує рівень

освітленості не менше 200 Лк. За ступенем пожежної безпеки приміщення належить до категорії В.

4.2.3 Навантаження та напруженість процесу праці

Під час виконання робіт використовують ПК та периферійні пристрої (лазерні та струменеві), що призводить до навантаження на окремі системи організму. Такі перекоси у напруженні різних систем організму, що трапляються під час роботи з ПК, зокрема, значна напруженість зорового аналізатора і довготривале малорухоме положення перед екраном, не тільки не зменшують загального напруження, а навпаки, призводять до його посилення і появи стресових реакцій.

Найбільшому ризику виникнення різноманітних порушень піддаються: органи зору, м’язово-скелетна система, нервово-психічна діяльність, репродуктивна функція у жінок.

Тобто наявні психофізіологічні небезпечні та шкідливі фактори:

а) фізичного перевантаження:

– статичного;

– динамічного;

б) нервово-психічного перевантаження:

– розумового перенапруження;

– монотонності праці;

– перенапруження аналізаторів;

– емоційних перевантажень.

Роботу за дипломним проектом визнано, таку, що займає 50% часу робочого дня та за восьмигодинної робочої зміни рекомендовано встановити додаткові регламентовані перерви тривалістю 15 хв через кожну годину роботи.

4.3 Виробнича санітарія

На підставі аналізу небезпечних та шкідливих факторів при виробництві (експлуатації), пожежної безпеки можуть бути надалі вирішені питання необхідності забезпечення працюючих достатньою кількістю освітлення, вентиляції повітря, організації заземлення, тощо.

4.3.1 Загальні заходи безпеки

Відповідно до санітарно-гігієнічних нормативів та правил експлуатації обладнання приклад деяких заходів безпеки:

1. Заходи безпеки під час експлуатації персонального комп’ютера та периферійних пристроїв передбачають:

– правильне організування місця праці та дотримання оптимальних режимів праці та відпочинку під час роботи з ПК;
– експлуатацію сертифікованого обладнання;
– дотримання заходів електробезпеки;
– забезпечення оптимальних параметрів мікроклімату;
– забезпечення раціонального освітлення місця праці (освітленість робочого місця не перевищувала 2/3 нормальної освітленості приміщення);
– облаштовуючи приміщення для роботи з ПК, потрібно передбачити припливно-витяжну вентиляцію або кондиціювання повітря;
– зниження рівня шуму та вібрації.

2. Заходи безпеки під час експлуатації інших електричних приладів передбачають дотримання таких правил:

– постійно стежити за справним станом електромережі, розподільних щитків, вимикачів, штепсельних розеток, лампових патронів, а також мережевих кабелів живлення, за допомогою яких електроприлади під’єднають до електромережі;

- постійно стежити за справністю ізоляції електромережі та мережевих кабелів, не допускаючи їхньої експлуатації з пошкодженою ізоляцією;
- не тягнути за мережевий кабель, щоб витягти вилку з розетки;
- не закривати меблями, різноманітним інвентарем вимикачі, штепсельні розетки;
- не підключати одночасно декілька потужних електропристроїв до однієї розетки, що може викликати надмірне нагрівання провідників, руйнування їхньої ізоляції, розплавлення і загоряння полімерних матеріалів;
- не залишати включені електроприлади без нагляду;
- не допускати потрапляння всередину електроприладів крізь вентиляційні отвори рідин або металевих предметів, а також не закривати їх та підтримувати в належній чистоті, щоб уникнути перегрівання та займання приладу;
- не ставити на електроприлади матеріали, які можуть під дією теплоти, що виділяється, загорітися (канцелярські товари, сувенірну продукцію тощо).

4.3.2 Електробезпека

Основним небезпечним фактором при роботі з ЕОМ є небезпека ураження людини електричним струмом, яка посилюється тим, що органи чуття людини не можуть на відстані виявити наявність електричної напруги на обладнанні.

Проходячи через тіло людини, електричний струм чинить на нього складний вплив, що є сукупністю термічної (нагрів тканин і біологічних середовищ), електролітичної (розкладання крові і плазми) і біологічної (роздратування і збудження нервових волокон та інших органів тканин організму) дій.

Тяжкість ураження людини електричним струмом залежить від цілого ряду чинників:

1) значення сили струму;

2) електричного опору тіла людини і тривалості протікання через нього струму;

3) типу і частоти струму;

4) індивідуальних властивостей людини і навколишнього середовища.

Приміщення для ЕОМ відноситься до приміщень без підвищеної небезпеки, тобто в приміщення, в яких відсутні умови, що створюють підвищену або особливу небезпеку. Небезпека ураження електричним струмом існує всюди, де використовуються електроустановки, тому приміщення без підвищеної небезпеки не можна назвати безпечними.

Електробезпека забезпечується:

1) відповідною конструкцією електроустановок;

2) застосуванням технічних способів і засобів захисту;

3) організаційними і технічними заходами.

Конструкція електроустановок відповідає умовам їх експлуатації та забезпечує захист персоналу від дотику до струмоведучих частин.

Основними технічними способами і засобами захисту від ураження електричним струмом, що використовуються окремо або в поєднанні один з одним, є:

1) захисне заземлення;

2) занулення;

3) вирівнювання потенціалів;

4) мале напруга;

5) електричне поділ мереж;

6) захисне відключення;

7) ізоляція струмоведучих частин;

8) компенсація струмів замикання на землю;

9) захисні пристрої;

10) попереджувальна сигналізація, блокування, знаки безпеки;

11) ізолюючі захисні та запобіжні пристосування.

4.3.3 Мікроклімат

Мікроклімат робочих приміщень – це клімат внутрішнього середовища цих приміщень, що визначається діючої на організм людини з'єднанням температури, вологості, швидкості переміщення повітря. В даному приміщенні проводяться роботи, що виконуються сидячи і не потребують динамічного фізичного напруження, то для нього відповідає категорія робіт Іа. Отже оптимальні значення для температури, відносної вологості й рухливості повітря для зазначеного робочого місця відповідають ДСН 3.3.6.042-99 «Санітарні норми мікроклімату виробничих приміщень» [17] і наведені в таблиці 4.3.

Таблиця 4.3 – Норми мікроклімату робочої зони об’єкту

Період рок	Температура, ^{o}C	Відносна вологість,%	Швидкість вітру, м/с, не більше
Холодний	21 - 23	60 – 40	0.1
Теплий	22 - 24	60 - 40	0.2

4.3.4 Освітлення

Світло є природною умовою існування людини. Воно впливає на стан вищих психічних функцій і фізіологічні процеси в організмі. Хороше освітлення діє тонізуюче, створює гарний настрій, покращує протікання основних процесів вищої нервової діяльності.

Збільшення освітленості сприяє поліпшенню працездатності навіть в тих випадках, коли процес праці практично не залежить від зорового сприйняття. При поганому освітленні людина швидко втомлюється, працює менш

продуктивно, виникає потенційна небезпека помилкових дій і нещасних випадків.

У проекті, що розробляється, передбачається використовувати суміщене освітлення. У світлий час доби використовуватиметься природне освітлення приміщення через віконні отвори, в решту часу використовуватиметься штучне освітлення.

Штучне освітлення в робочому приміщенні передбачається здійснювати з використанням люмінесцентних джерел світла у світильниках загального освітлення, оскільки люмінесцентні лампи мають високу потужність (80 Вт), тривалий термін служби (до 10000 годин), спектральним складом випромінюваного світла, близький до сонячного. При експлуатації ПК виконується зорова робота IV в розряду точності (середня точність). При цьому нормована освітленість на робочому місці (Ен) рівна 200 лк. Джерелом природного освітлення є сонячне світло. У приміщенні, де розташовані ЕОМ передбачається природне бічне освітлення, рівень якого відповідає ДБН В.2.5-28:2018 Природне і штучне освітлення [19]

Регулярно повинен проводитися контроль освітленості, який підтверджує, що рівень освітленості задовольняє [19]

Розрахунок освітлення.

Для виробничих та адміністративних приміщень світловий коефіцієнт приймається не менше -1/8, в побутових – 1/10:

$$S_b = \left(\frac{1}{5} \div \frac{1}{10}\right) \cdot S_n \qquad (4.9)$$

де Sb – площа віконних прорізів, м2;

Sn – площа підлоги, м2.

Sn = a × b = 3 × 6 = 18 м2 ,

S = 1/8 × 18 = 2.25м2 .

Приймаємо 1 вікно площею S = 2.25м2. Світильники загального освітлення розташовуються над робочими поверхнями в рівномірно-прямокутному порядку. Для організації освітлення в темний час доби передбачається обладнати приміщення, довжина якого складає 4м, ширина 4м, світильниками ЛПО2П, оснащеними лампами типу ЛБ (дві по 80Вт) з світловим потоком 5400лм кожна. Розрахунок штучного освітлення виробляється по коефіцієнтах використання світлового потоку, яким визначається потік, необхідний для створення заданої освітленості при загальному рівномірному освітленні.

Розрахунок кількості світильників n визначається по формулі (4.10):

$$n = \frac{E \times S \times Z \times K}{F \times U \times M} \qquad (4.10)$$

Де Е нормована освітленість робочої поверхні, визначається нормами – 300лк;

S – освітлювана площа, м2; S = 18 м2;

Z – поправочний коефіцієнт світильника (Z = 1.15 для ламп розжарювання та ДРЛ;

Z = 1,1 для люмінесцентних ламп) приймаємо рівним 1.1;

K – коефіцієнт запасу, що враховує зниження освітленості в процесі експлуатації – 1.5;

U – коефіцієнт використання, залежний від типу світильника, показника індексу приміщення і т.п. – 0.575;

M – число люмінесцентних ламп в світильнику – 2;

F – світловий потік лампи – 5400лм (для ЛБ-80). Підставивши числові значення у формулу (4.10), отримуємо:

$$n = \frac{300 \times 18 \times 1.1 \times 1.5}{5400 \times 0.575 \times 2} \approx 1.43 \qquad (4.11)$$

Приймаємо освітлювальну установку, яка складається з 2-х світильників, які складаються з двох люмінесцентних ламп загальною потужністю 160Вт, напругою 220В.

4.3.5 Рекомендації щодо пожежної безпеки

Пожежна безпека при застосуванні ПК забезпечується:

– системою запобігання пожежі,
– системою протипожежного захисту,
– організаційно-технічними заходами.

Згідно ДСТУ Б В.1.1-36:2016 [20] таке приміщення, площею 25 м2, відноситься до категорії "В" (пожежонебезпечної) та для протипожежного захисту в ньому проектом передбачено устаткування автоматичною пожежною сигналізацією із застосуванням датчиків-сповіщувачів РІД-1 (сповіщувач димовий ізоляційний) в кількості 1 шт., і застосуванням первинних засобів пожежогасіння. Відповідно до норм первинних засобів пожежогасінні пропонується використовувати:

– ручний вуглекислий вогнегасник ОУ-5 в кількості 1 шт,
– повсть 1 1 м2, кошму 2×1,5 м2 або азбестове полотно 2×2 м2 в кількості 1 шт.

Виникнення пожежі можливе, якщо на об'єкті є горючі речовини, окислювач і джерела запалювання. Вірогідність пожежної небезпеки приймається значною, якщо ймовірна взаємодія цих трьох чинників. Горючими компонентами є: будівельні матеріали для акустичної і естетичної обробки приміщень, перегородки, підлоги, двері, ізоляція силових, сигнальних кабелів і т.д.

Горючими матеріалами в приміщенні, де розташовані ПК, є:

– поліамід – матеріал корпусу мікросхем, горюча речовина, температура самозаймання 420 ˚С,

– полівінілхлорид – ізоляційний матеріал, горюча речовина, температура запалювання 335˚С, температура самозаймання 530 ˚С,
– склотекстоліт ДЦ – матеріал друкарських плат, важкогорючий матеріал, показник горючості 1.74, не схильний до температурного самозаймання,
– пластикат кабельний №.489 – матеріал ізоляції кабелів, горючий матеріал, показник горючості більше 2.1,
– деревина – будівельний і обробний матеріал, з якого виготовлені меблі, горючий матеріал, показник горючості більше 2.1, температура запалювання 255 ℃, температура самозаймання 399 ˚С.

Для відводу теплоти від ПК діє система кондиціонування. Тому кисень, як окиснювач процесів горіння, є в будь-якій точці приміщень ВЦ.

Простори усередині приміщень в межах, яких можуть утворюватися або знаходиться пожежонебезпечні речовини і матеріали відповідно до [20] відносяться до пожежонебезпечної зони класу П-IIа. Це обумовлено тим, що в приміщенні знаходяться тверді горючі та важко займисті речовини та матеріали. Приміщенню, у якому розташоване робоче місце, присвоюється II ступень вогнестійкості.

Потенційними джерелами запалювання можуть бути:

– іскри і дуги короткого замикання;
– електрична іскра при замиканні і розмиканні ланцюгів;
– перегріви від тривалого перевантаження,
– відкритий вогонь і продукти горіння,
– наявність речовин, нагрітих вище за температуру самозаймання,
– розрядна статична електрика.

Причинами можливого загоряння і пожежі можуть бути:

– несправність електроустановки,
– конструктивні недоліки устаткування,
– коротке замикання в електричних мережах,

– запалювання горючих матеріалів, що знаходяться в безпосередній близькості від електроустановки.

Продуктами згорання, що виділяються на пожежі, є: окис вуглецю; сірчистий газ; окис азоту; синильна кислота; акромін; фосген; хлор і ін. При горінні пластмас, окрім звичних продуктів згорання, виділяються різні продукти термічного розкладання: хлорангідриді кислоти, формальдегіди, хлористий водень, фосген, синильна кислота, аміак, фенол, ацетон, стирол. (ГОСТ 12.1.044-89) [21].

Для захисту персоналу від дії небезпечних і шкідливих чинників пожежі проектом передбачається застосування промислового протигаза, що фільтрує, з коробкою марки «В» із сірою відміткою забарвлення – захист від неорганічних газів (хлор, фтор, бром, сірководень, сірковуглець, хлорциан, галогени), а цей фільтр не захистить від СО (тобто від чадного газу).

Можливе також відповідне застосування фільтрувальної коробки з маркуванням «СО» із фіолетовим забарвленням на фільтрі означає, що він захищає від Чадного газу. Або фільтру для протигазу з літерним маркуванням «SX» із фіолетовим забарвленням захистить від спец речовин таких як (зарин, зоман та фосген).

Висновок до розділу 4

У розділі "Охорона праці" виконаний аналіз потенційних небезпек при роботі із засобами обчислювальної техніки. Приведені рекомендації щодо організації робочого місця, електробезпеки та пожежної безпеки. Наведені розміри приміщення та наведено значення температури, вологості й рухливості повітря, необхідна кількість і потужність ламп та інші параметри, значення яких впливає на умови праці, рекомендації з охорони праці, техніки безпеки при роботі на комп’ютері.

ВИСНОВКИ

Метою даної роботи було створення мобільного додатку для контролю грошей за допомогою ведення своїх витрат і доходів кожного дня.

У першому розділі була розглянута предметна область для визначення актуальності теми проекту. Також були проаналізовані переваги та недоліки конкурентів, що дозволило зрозуміти, що потрібно користувачеві і допомогло визначиться з дизайном і функціоналом додатку. Насамкінець були сформульовані цілі та завдання для реалізації проекту.

У другому розділі було описано архітектуру проекту та її компоненти. Спроектовано базу даних та описана взаємодія користувача з системою. Вибрано середовище для розробки проекту і описано основний функціонал системи

У третьому розділі був представлений розроблений інтерфейс користувача та описана взаємодія користувача з ним.

У четвертому розділі було визначено параметри і характеристики приміщення для роботи над проектом, заходи, які потрібно зробити для того, щоб дане приміщення відповідало необхідним нормам і було комфортним і безпечним для роботи. Приведені рекомендації щодо організації робочого місця, електробезпеки та пожежної безпеки. Проаналізовано основні навантаження та напруженості у процесу праці над проектом.

У результаті роботи була створена основна частина функціоналу мобільного додатку для контролю грошей. Через брак часу не було реалізовано весь запланований функціонал додатку, а саме синхронізація з онлайн базою даних.

У перспективі розвитку проекту буде дороблена синхронізація, доданий сканер чеків який дозволить швидше додавати нову інформацію та буде додано більш детальну статистику доходів та витрат.

ПЕРЕЛІК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Global market share held by the leading smartphone operating systems in sales to end users from 1st quarter 2009 to 2nd quarter 2018. [Електронний ресурс] https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/ (дата звернення 13.05.2019).
2. Report on the Economic Well-Being of U.S. Households in 2017. May 2018. [Електронний ресурс] https://www.federalreserve.gov/publications/files/2017-report-economic-well-being-us-households-201805.pdf (дата звернення 15.05.2019).
3. Monefy [Електронний ресурс] http://www.monefy.me (дата звернення 14.05.2019).
4. MoneyLover [Електронний ресурс] https://moneylover.me (дата звернення 14.05.2019).
5. 1Money [Електронний ресурс] https://play.google.com/store/apps/details?id=org.pixelrush.moneyiq (дата звернення 14.05.2019).
6. CoinKeeper [Електронний ресурс] https://coinkeeper.me (дата звернення 14.05.2019).
7. Архітектура системи [Електронний ресурс] https://ru.wikipedia.org/wiki/Архитектура_системы (дата звернення 04.06.2019)
8. Проектування бази даних [Електронний ресурс] https://.wikipedia.org/wiki/Проектування_бази_даних (дата звернення 04.06.2019)
9. SQLite [Електронний ресурс] https://www.sqlite.org./about.html (дата звернення 05.06.2019).
10. Android Studio [Електронний ресурс] https://en.wikipedia.org/wiki/Android_Studio (дата звернення 05.06.2019)

11. Android lvl api [Електронний ресурс] https://developer.android.com/guide/topics/manifest/uses-sdk-element (дата звернення 05.06.2019)

12. XML [Електронний ресурс] https://ru.wikipedia.org/wiki/XML (дата звернення 06.06.2019)

13. Activity [Електронний ресурс] https://developer.android.com/guide/components/activities/?hl=ru (дата звернення 07.06.2019)

14. Fragment [Електронний ресурс] https://developer.android.com/guide/components/fragments?hl=RU (дата звернення 07.06.2019)

15. PHP [Електронний ресурс] https://ru.wikipedia.org/wiki/PHP (дата звернення 07.06.2019)

16. Закон України «Про охорону праці». [Електронний ресурс] https://zakon.rada.gov.ua/laws/show/2694-12 (дата звернення 10.06.2019)

17. Санітарні норми мікроклімату виробничих приміщень ДСН 3.3.6.042-99. [Електронний ресурс] https://zakon.rada.gov.ua/rada/show/va042282-99 (дата звернення 10.06.2019)

18. Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин ДСанПІН 3.3.2.007-98. [Електронний ресурс] https://zakon.rada.gov.ua/rada/show/v0007282-98 (дата звернення 10.06.2019)

19. Природне і штучне освітлення ДБН В.2.5-28:2018. [Електронний ресурс] https://dbn.co.ua/load/normativy/dbn/dbn_v_2_5_28/1-1-0-1188 (дата звернення 10.06.2019)

20. Визначення категорій приміщень, будинків, установок за вибухопожежною та пожежною небезпекою ДСТУ Б В.1.1-36:2016

2018. [Електронний ресурс] https://dbn.co.ua/load/normativy/dstu/dstu_b_v_1_1_36/5-1-0-1759 (дата звернення 10.06.2019)

21. ГОСТ 12.1.044-89 [Електронний ресурс] http://docs.cntd.ru/document/gost-12-1-044-89 (дата звернення 10.06.2019)

Додаток А.

Лістинг класу реалізації діаграми витрат

```
package com.solomade.casheater;
import android.content.Intent;
import android.database.Cursor;
import android.graphics.Color;
import android.graphics.PorterDuff;
import android.graphics.PorterDuffColorFilter;
import android.graphics.drawable.Drawable;
import android.os.Bundle;
import android.app.Fragment;
import android.text.SpannableString;
import android.text.SpannableStringBuilder;
import android.text.style.ForegroundColorSpan;
import android.view.GestureDetector;
import android.view.LayoutInflater;
import android.view.MotionEvent;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageButton;
import android.widget.ImageView;
import com.github.mikephil.charting.charts.PieChart;
import com.github.mikephil.charting.components.Legend;
import com.github.mikephil.charting.data.PieData;
import com.github.mikephil.charting.data.PieDataSet;
import com.github.mikephil.charting.data.PieEntry;
import com.github.mikephil.charting.highlight.Highlight;
import com.solomade.casheater.database.CategoryTable;
import com.solomade.casheater.database.DbHelper;
import com.solomade.casheater.database.TransactionTable;
import com.solomade.casheater.transactionMVP.AddTransactionActivity;
import java.text.DecimalFormat;
import java.text.DecimalFormatSymbols;
import java.util.ArrayList;
import java.util.List;
/**
 * A simple {@link Fragment} subclass.
 */
public class FragmentDiagram extends Fragment {
    PieChart chart;
```

```
        ArrayList<Integer> num = new ArrayList<>();
        ArrayList<String> str;
        ArrayList<Integer> id_cat_plus  = new ArrayList<>();
        ArrayList<Integer> id_button_plus  = new ArrayList<>();
        ArrayList<Integer> colorCat = new ArrayList<>();
        DbHelper dbHelper;
        Cursor cursor;
        PieDataSet dataSet = new PieDataSet(null,"");
        float plusBalance = 0;
        float minusBalance = 0;
        private GestureDetector gestureDetector;
        public FragmentDiagram() { }
        @Override
        public View onCreateView(LayoutInflater inflater, ViewGroup
container,
                                 Bundle savedInstanceState) {
            View view = inflater.inflate(R.layout.fragment_diagram,
container, false);
            initCategoryBtn(view);
            chart = (PieChart) view.findViewById(R.id.chart);
            PlusAndMinusBalane(container);
            setupPieChart(container);
            return view;
        }
        private void PlusAndMinusBalane(ViewGroup container){
            plusBalance = 0;
            minusBalance = 0;
            dbHelper = new DbHelper(container.getContext());
            cursor =
dbHelper.getReadableDatabase().query(TransactionTable.TABLE, null, null,
null, null, null, null);
            while (cursor.moveToNext()) {
                 float cursorBalance =
cursor.getFloat(cursor.getColumnIndex(TransactionTable.COLUMN.SUM));
                 if (cursorBalance > 0) {
                     plusBalance+= cursorBalance;
                 }
                 else {
                     minusBalance -= cursorBalance;
                 }
            }
        }
        private void setupPieChart(ViewGroup container)
```

```
        {
            List<PieEntry> pieEntrie = new ArrayList<>();
            dbHelper = new DbHelper(container.getContext());
            str = new ArrayList<>();
            num = new ArrayList<>();
            colorCat = new ArrayList<>();
            boolean doubleCat;
            cursor =
dbHelper.getReadableDatabase().query(TransactionTable.TABLE, null, null,
null, null, null, null);
            while (cursor.moveToNext()) {
                doubleCat = false;
                    for(int i = 0; i < str.size(); i++){

if(str.get(i).equals(cursor.getString(cursor.getColumnIndex(TransactionTable.
COLUMN.CATEGORY)))){
                            doubleCat = true;

num.set(i,num.get(i)+cursor.getInt(cursor.getColumnIndex(TransactionTable.COL
UMN.SUM)));
                        }
                    }
                    if(doubleCat == false) {

num.add(cursor.getInt(cursor.getColumnIndex(TransactionTable.COLUMN.SUM)));

str.add(cursor.getString(cursor.getColumnIndex(TransactionTable.COLUMN.CATEGO
RY)));
                    }
                }
            for (int i = 0; i < num.size(); i++){
                if(num.get(i) < 0)
                    pieEntrie.add(new PieEntry(num.get(i)*-1,str.get(i)));
                else{
                    num.remove(i);
                    str.remove(i);
                    i--;
                }
            }
            cursor.close();
            for(int i = 0; i < str.size(); i++){
```

```
                cursor = dbHelper.getReadableDatabase().rawQuery("SELECT *
FROM " + CategoryTable.TABLE + " WHERE " + CategoryTable.COLUMN.PLUSORMINUS+
" = '" +-1+"'",null);
                while (cursor.moveToNext()) {

if(cursor.getString(cursor.getColumnIndex(CategoryTable.COLUMN.NAME)).equals(
str.get(i))){
                        String srt1 =
cursor.getString(cursor.getColumnIndex(CategoryTable.COLUMN.COLOR));
                        int qwe = Color.parseColor(srt1);
                        colorCat.add(qwe);
                    }
                }
            }
            cursor.close();
            dataSet = new PieDataSet(pieEntrie,"");
            dataSet.setColors(colorCat);
            PieData data = new PieData(dataSet);
            data.setDrawValues(false);
            data.setValueTextColor(Color.WHITE); // цвет значений
            data.setValueTextSize(10f);
            data.setValueFormatter(new MyYAxisValueFormatter("",num));
            data.getYValueSum();
            DecimalFormatSymbols decimalSymbols =
DecimalFormatSymbols.getInstance();
            decimalSymbols.setDecimalSeparator('.');
            DecimalFormat df = new DecimalFormat("0.00",decimalSymbols);
            String plus = df.format(plusBalance);
            String minus = df.format(minusBalance);
            SpannableStringBuilder builder = new SpannableStringBuilder();
            SpannableString plusBalanceColor= new SpannableString(plus);
            plusBalanceColor.setSpan(new
ForegroundColorSpan(getResources().getColor(R.color.main_color)), 0,
plus.length(), 0);
            builder.append(plusBalanceColor);
            builder.append('\n');
            SpannableString minusBalanceColor= new SpannableString(minus);
            minusBalanceColor.setSpan(new
ForegroundColorSpan(getResources().getColor(R.color.main_color_red)), 0,
minus.length(), 0);
            builder.append(minusBalanceColor);
            chart.setData(data);
            chart.setCenterText(builder);
```

```
            chart.setRotationEnabled(false);
            // Вокруг круга
            chart.setTransparentCircleAlpha(20);
            chart.setTransparentCircleColor(Color.BLACK);
            chart.setTransparentCircleRadius(75f);
            chart.setDrawEntryLabels(false);
            chart.setHoleRadius(60f);
            chart.setDescription(null);
            chart.setDrawHoleEnabled(true);
            chart.setHoleColor(Color.WHITE);
            chart.setOnTouchListener(new View.OnTouchListener() {
                @Override
                public boolean onTouch(View v, MotionEvent event) {
                    switch (event.getAction()) {
                        case MotionEvent.ACTION_DOWN:
                            if((Math.pow(event.getX() -
chart.getCenter().getX(),2)) + (Math.pow(event.getY() -
chart.getCenter().getY(),2)) <= Math.pow((chart.getRadius() *
(chart.getHoleRadius() / 100)),2)) {
                                break;
                            }
                            else {
                                Highlight high =
chart.getHighlightByTouchPoint(event.getX(), event.getY());
                                if(high == null)
                                    break;
                                chart.highlightValue(high);

chart.setCenterText(str.get((int)high.getX())+"\n"+high.getY());

chart.setHoleColor(dataSet.getColor((int)high.getX()));

chart.setCenterTextColor(getResources().getColor(R.color.main_color_white));
                                break;
                            }
                        case MotionEvent.ACTION_UP:
                            chart.highlightValues(null);
                            DecimalFormatSymbols decimalSymbols =
DecimalFormatSymbols.getInstance();
                            decimalSymbols.setDecimalSeparator('.');
                            DecimalFormat df = new
DecimalFormat("0.00",decimalSymbols);
                            String plus = df.format(plusBalance);
```

```
                            String minus = df.format(minusBalance);
                            SpannableStringBuilder builder = new
SpannableStringBuilder();
                            SpannableString plusBalanceColor= new
SpannableString(plus);
                            plusBalanceColor.setSpan(new
ForegroundColorSpan(getResources().getColor(R.color.main_color)), 0,
plus.length(), 0);
                            builder.append(plusBalanceColor);
                            builder.append('\n');
                            SpannableString minusBalanceColor= new
SpannableString(minus);
                            minusBalanceColor.setSpan(new
ForegroundColorSpan(getResources().getColor(R.color.main_color_red)), 0,
minus.length(), 0);
                            builder.append(minusBalanceColor);
                            chart.setCenterText(builder);
                            chart.setHoleColor(Color.WHITE);
                            v.performClick();
                            break;
                        default:
                            break;
                    }
                    return true;
                }
            });
            Legend legend = chart.getLegend();
            legend.setEnabled(false);
            chart.invalidate();
        }
        public void initCategoryBtn(View view){
            ImageButton[] buttons = new ImageButton[12];
            gestureDetector = new GestureDetector(new SingleTapConfirm());
            buttons[0] = view.findViewById(R.id.btnCat1);
            buttons[1] = view.findViewById(R.id.btnCat2);
            buttons[2] = view.findViewById(R.id.btnCat3);
            buttons[3] = view.findViewById(R.id.btnCat4);
            buttons[4] = view.findViewById(R.id.btnCat5);
            buttons[5] = view.findViewById(R.id.btnCat6);
            buttons[6] = view.findViewById(R.id.btnCat7);
            buttons[7] = view.findViewById(R.id.btnCat8);
            buttons[8] = view.findViewById(R.id.btnCat9);
            buttons[9] = view.findViewById(R.id.btnCat12);
```

```
            buttons[10] = view.findViewById(R.id.btnCat11);
            buttons[11] = view.findViewById(R.id.btnCat10);
            DbHelper dbHelper;
            Cursor cursor;
            dbHelper = new DbHelper(view.getContext());
            String image = "image_bely_plyus";
            cursor = dbHelper.getReadableDatabase().rawQuery("SELECT * FROM "
+ CategoryTable.TABLE +
                    " WHERE " + CategoryTable.COLUMN.PLUSORMINUS+ " = '" +-
1+"' AND "+
                    CategoryTable.COLUMN.IMAGE+" != '"+image+"'"  ,null);
            int i = 0;

            while (cursor.moveToNext()) {
                buttons[i].setOnTouchListener(listenerTouch);
                int id =
getActivity().getResources().getIdentifier(cursor.getString(cursor.getColumnI
ndex(CategoryTable.COLUMN.IMAGE)), "drawable",
getActivity().getPackageName());
                buttons[i].setImageResource(id);
                String srt =
cursor.getString(cursor.getColumnIndex(CategoryTable.COLUMN.COLOR));
                int qwe = Color.parseColor(srt);
                Drawable mDrawable  = buttons[i].getBackground();
                mDrawable.setColorFilter(new
PorterDuffColorFilter(qwe,PorterDuff.Mode.SRC_ATOP));
                buttons[i].setBackground(mDrawable);

buttons[i].setTag(cursor.getString(cursor.getColumnIndex(CategoryTable.COLUMN
.NAME)));
                i++;
            }
            cursor.close();
            while (i<12) {
                    buttons[i].setOnTouchListener(listenerTouch);
                    id_button_plus.add(buttons[i].getId());
                    id_cat_plus.add(-2);
                    buttons[i].setScaleType(ImageView.ScaleType.FIT_CENTER);

buttons[i].setBackground(getResources().getDrawable(R.drawable.drawable_plus_
btn_color_main));
                    buttons[i].setTag("");
                    i++;
```

```
            }
            cursor.close();
        }
        View.OnTouchListener listenerTouch = new View.OnTouchListener() {
            @Override
            public boolean onTouch(View v, MotionEvent event) {
                if (gestureDetector.onTouchEvent(event)) {
                        ImageButton b = (ImageButton)v;
                        for (int i = 0; i < id_button_plus.size(); i++){
                            if(id_button_plus.get(i) == b.getId()){
                                Intent myIntent = new Intent(getActivity(),
update_one_cat_activity.class);

myIntent.putExtra("id_cat",id_cat_plus.get(i).toString());
                                startActivity(myIntent);
                                return true;
                            }
                        }
                        String  tag_btn = "-";
                        String cat_name = b.getTag().toString();
                        Intent myIntent = new Intent(getActivity(),
AddTransactionActivity.class);
                        myIntent.putExtra("plusorminus", tag_btn);
                        myIntent.putExtra("categoryName",cat_name);
                        startActivity(myIntent);
                }
                switch (event.getAction()) {
                    case MotionEvent.ACTION_DOWN:
                        ImageButton clickedButton = (ImageButton) v;
                        String buttonText =
clickedButton.getTag().toString();
                        for (int i = 0; i < str.size(); i++){
                            if(str.get(i).equals(buttonText)){
                                chart.highlightValue(i, 0, false);

chart.setCenterText(str.get(i)+"\n"+num.get(i)*-1);
                                chart.setHoleColor(colorCat.get(i));

chart.setCenterTextColor(getResources().getColor(R.color.main_color_white));
                                chart.invalidate();
                            }
                        }
                        break;
```

```
                    case MotionEvent.ACTION_UP:
                        chart.highlightValues(null);
                        DecimalFormatSymbols decimalSymbols =
DecimalFormatSymbols.getInstance();decimalSymbols.setDecimalSeparator('.');
                                DecimalFormat df = new
DecimalFormat("0.00",decimalSymbols);
                                String plus = df.format(plusBalance);
                                String minus = df.format(minusBalance);
                                SpannableStringBuilder builder = new
SpannableStringBuilder();
                                SpannableString plusBalanceColor= new
SpannableString(plus);
                                plusBalanceColor.setSpan(new
ForegroundColorSpan(getResources().getColor(R.color.main_color)), 0,
plus.length(), 0);
                                builder.append(plusBalanceColor);
                                builder.append('\n');
                                SpannableString minusBalanceColor= new
SpannableString(minus);
                                minusBalanceColor.setSpan(new
ForegroundColorSpan(getResources().getColor(R.color.main_color_red)), 0,
minus.length(), 0);
                                builder.append(minusBalanceColor);
                                chart.setCenterText(builder);
                                chart.setHoleColor(Color.WHITE);
                                  v.performClick();
                                break;
                            default:
                                break;
                        }
                        return true;
                    }
        };
        private class SingleTapConfirm extends
GestureDetector.SimpleOnGestureListener {

            @Override
            public boolean onSingleTapUp(MotionEvent event) {
                return true;
            }
        }
    }
```

Додаток Б.

Презентація

Міністерство освіти і науки України
Східноукраїнський національний університет ім. В.Даля

Комплексна тема. Мобільний додаток обліку фінансових операцій: стаціонарний модуль

Студент групи КІ-15д
Рибальченко В.Р.

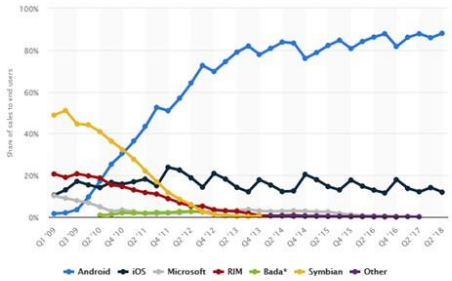
Керівник проекту:
Скарга-Бандурова І.С.

Мета роботи

Розробка мобільного додатку для контролю грошей за допомогою ведення своїх витрат і доходів кожного дня.

Чому Android?

За статистичними даними всесвітньої частки ринку провідних операційних систем смартфонів з точки зору продажів кінцевим користувачам за перші два квартали 2018 року ОС Android займає 88% ринку.



Існуючі рішення

Monefy

Money Lover

1Money

CoinKeeper

Вимоги до функціоналу

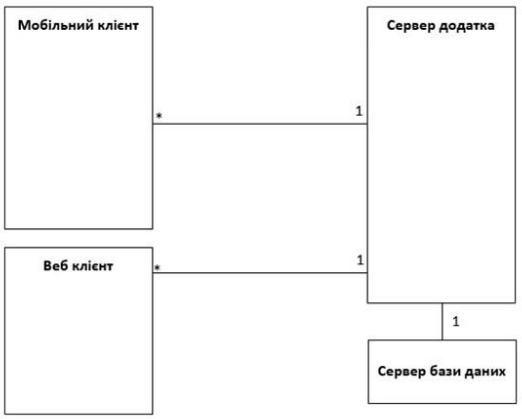
- Можливість вести облік фінансових операцій, проглядати, редагувати да заповнювати історію грошових транзакцій.
- Можливість створення та редагування особистих категорій, аналіз графіків витрат, прибутків.
- Мінімізація займаної постійної пам'яті на пристрої, високий рівень плавності роботи.
- Синхронізація зі стаціонарним модулем за допомогою облікового запису.

Структура системи

Сервер додатку – відповідає за прийом інформації для подальшої її обробки і передачі на сервер бази даних системи.

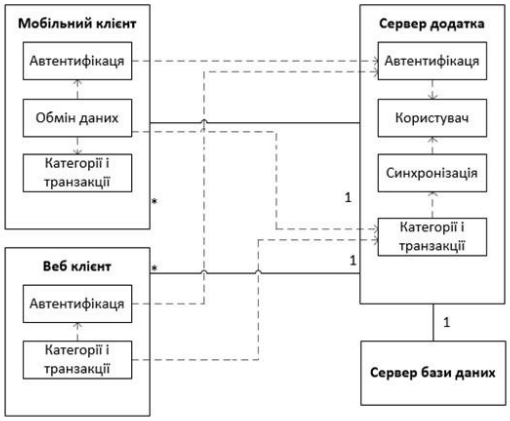
Мобільний клієнт – має можливість працювати у двох режимах. Режимі онлайн і в режимі офлайн.

Веб-клієнт – має той же функціонал, що і мольний клієнт крім можливості роботи у офлайн режими.



Підсистеми

- Підсистема автентифікації перевіряє справжність користувача
- Підсистема користувача – Керування даними користувача
- Підсистема синхронізації - синхронізація даних на сервері бази даних
- Підсистема обміну даних – синхронізація даних мобільного додатку з сервером
- Підсистема Категорії і транзакцій - Керування категорія и транзакціями



Проміжні результати

Фрагмент програмного коду відстеження натискання на діаграму

```
if ((Math.pow(event.getX() - chart.getCenter().getX(), 2)) +
        (Math.pow(event.getY() - chart.getCenter().getY(), 2)) <=
        Math.pow((chart.getRadius() * (chart.getHoleRadius() / 100)), 2)) {
    break;
} else {
    Highlight high = chart.getHighlightByTouchPoint(event.getX(), event.getY());
    if (high == null)
        break;
    chart.highlightValue(high);
    chart.setCenterText(str.get((int) high.getX()) + "\n" + high.getY());
    chart.setHoleColor(dataSet.getColor((int) high.getX()));
    chart.setCenterTextColor(getResources().getColor(R.color.main_color_white));
    break;
}
```

Перспективи розвитку

- Сканер чеків який дозволить набагато швидше конкурентних аналогів додавати нову інформацію

- Покращена система надання статистичної інформації

- Дороблена система синхронізації між клієнтами системи

- Система прогнозування фінансової ситуації користувача