МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет імені Петра Могили
Факультет комп'ютерних наук

Кафедра інженерії програмного забезпечення

ФАЛІНСЬКА ЄЛИЗАВЕТА СЕРГІЇВНА

УДК 004.4

МОБІЛЬНИЙ ОРГАНАЙЗЕР ІЗ ФУНКЦІЄЮ ОБЛІКУ ФІНАНСІВ

Автореферат кваліфікаційної роботи на здобуття ступеня вищої освіти «Бакалавр»

Спеціальність 121 «Інженерія програмного забезпечення»

Освітня кваліфікація
«Бакалавр з інженерії програмного забезпечення»

Миколаїв – 2021

Кваліфікаційною роботою є рукопис.

Робота виконана в Чорноморському національному університеті імені

Петра Могили Міністерства освіти і науки України на кафедрі інженерії програмного забезпечення.

Керівник: Старший викладач кафедри ІПЗ
Дворецька Світлана Володимирівна

Рецензент: Канд. фіз.-мат. наук, доцент
Кафедри ІІС
Кулаковська Інесса Василівна

Захист відбудеться «25» червня 2021 р. о 9 год. на засіданні екзаменаційної комісії (ауд. 2-309) у Чорноморському національному університетіімені Петра Могили за адресою: вул. 68 Десантників, 10, Миколаїв, 54003.

З кваліфікаційною роботою можна ознайомитися в бібліотеці Чорноморського національного університету імені Петра Могили за адресою: вул. 68 Десантників, 10, Миколаїв, 54003.

Автореферат представлений «26» травня 2021р.

Секретар
екзаменаційної
комісії,
Викладач кафедри ІПЗ І.О. Кандиба

ЗАГАЛЬНА ХАРАКТЕРИСТИКА РОБОТИ

Актуальність та науково-практичне значення теми: Актуальність обраної теми полягає в тому, що темп життя сучасної людини дуже швидкий та кожен день достатньо навантажений, через що неможливо тримати все необхідне у своїй пам’яті, щоб не пропустити важливі та заплановані події у своєму розкладі.. В наш час існує безліч мобільних застосунків для будь–якої сфери життя. Вони можуть бути для розваг та відпочинку, навчання та саморозвитку, або для спрощення деяких повсякденних аспектів життя. Кожному сьогодні хочеться мати якомога більше функцій у своєму телефоні, так як смартфон став невід’ємною частиною сучасної людини і дуже зручно мати все що необхідно у повсякденному житті у своїй кишені. Таким чином з’явився попит на мобільні застосунки, які дозволяють планувати справи або відслідковувати власні фінансові операції. З існуючих аналогів на даний момент, на жаль, немає таких, які б поєднували обидві функції, що не є досить зручним, бо необхідно мати в своєму смартфоні декілька застосунків для ведення записів пов’язаних з повсякденним життям. Саме тому розробка мобільного застосунку, який вирішує перелічені проблеми та є легким у використанні, є актуальною.

Мета роботи: підвищення зручності процесу планування часових та фінансових витрат шляхом розробки мобільного органайзеру із функцією обліку фінансів.

Для досягнення визначеної мети необхідно вирішити такі завдання:

- проведення порівняльного аналізу існуючих аналогів мобільних застосунків;
- визначення функцій системи;
- вибір стеку технологій для мобільного застосунку;
- створення бази даних для застосунку;
- реалізація програмного коду мобільного застосунку;

– опис користувацького інтерфейсу.

Об’єкт дослідження: процеси планування часу та фінансових витрат, та підходи щодо їх автоматизації.

Предмет дослідження: технології та інструменти розробки мобільних застосунків для планування вільного часу та фінансових витрат.

Обсяг і структура КРБ: КРБ викладена на 60 сторінок, вона містить 3 розділи, 51 ілюстрацій, 1 таблицю, 16 джерел в переліку посилань.

ОСНОВНИЙ ЗМІСТ ДИПЛОМНОЇ РОБОТИ

У **вступі** аргументовано актуальність обраної теми, визначено мету даної роботи, поставлені завдання для досягнення означеної мети та сформульовані об’єкт і предмет.

У першому розділі – «**Аналіз існуючих рішень та технологій розробки**» розглянуто декілька аналогів мобільних застосунків до розроблювального застосунку, а саме: «IsoTimer», «TickTick», «1Money» та «Monefy». Дані застосунки відрізняються та мають різні підходи для розв’язання проблеми. Під час аналізу раніше згаданих застосунків розглянуто їх функціонали та приклади інтерфейсу для розробки власних ідей задля поліпшення функціоналу та користувацького досвіду розроблювального мобільного застосунку, а також виділено їх недоліків для усунення подібних в розроблюваному застосунку. Наведені аналоги мають багато функцій, які можуть навіть відштовхувати користувачів та не має такого аналогу, який би вміщував у собі всі необхідні функції як органайзера так і застосунку для обліку фінансів, що підкреслює актуальність розробки відповідного застосунку.

Для реалізації застосунку обрано такі технології, як Ionic, Angular та Firebase, та розглянуто їх функціональні можливості. Кожна з розглянутих технологій використовує найсучасніші методи розробки програмного забезпечення.

Також визначені та складені вимоги до програмного забезпечення, яким має відповідати вже створений у кінці мобільний застосунок.

У підрозділі 1.1 «Аналіз аналогів» розглянуто декілька аналогів застосунків для виявлення вдалих та поганих рішень. Мобільний застосунок «IsoTimer» має проблеми із синхронізацією даних, після заміни приладу та повторного встановлення застосунку користувачі не мають доступу до своїх збережених подій та задач, які були вже в історії та заплановані на майбутнє. Також не є дуже зручним з погляду користувацького досвіду, тобто він є складним в використовувані та налаштуванні для звичайного користувача. Ще одним недоліком є те, що безкоштовна версія цього застосунку не має всіх запропонованих функцій та для використання застосунку в повному обсязі необхідно купувати абонемент.

Другий застосунок для планування справ під назвою «TickTick» також має деякі недоліки, наприклад кольори тем, на які користувачі скаржаться, за їх нечитабельність текстів та незручність іконок. Головним недоліком є те, що безкоштовна версія дуже урізана в функціоналі, а в платній версії є проблеми з надмірною оплатою і також проблеми з роботою деяких функцій, наприклад із збереженням задач з персональним налаштуванням користувачів.

Наступний застосунок для контролю власного бюджету, який називається «1Money», одним з головних недоліків якого є те, що для користування цим застосунком необхідно сплачувати щомісячну підписку. Також цей застосунок є складним для інтуїтивного розуміння інтерфейсу, що відштовхує потенційних користувачів. Ще одним виявленим недоліком є те, що при помилковому створені запису витрат неможливо видалити або виправити введені дані. До того ж треба звернути увагу, що є багато скарг від користувачів, а від розробників надалі не має ніякого розвитку застосунку та покращень.

Останній застосунок під назвою «Monefy» розглядаючи недоліки якого, хочеться, по-перше, виділити те, що як і у багатьох застосунків у

безкоштовній версії дуже обмежений функціонал, для більше-менш зручнішого користування застосунком необхідно купувати платну версію. До того ж, щоб користуватись цим застосунком на декількох пристроях (розробник заявляв про можливість користування застосунком всією сім'єю) необхідно купувати стільки ж версій застосунку. Наступним важливим пунктом треба зазначити, що є проблеми із синхронізацією та резервними копіями у користуванні в сімейній версії та при переході на новий пристрій. Ще одним недоліком є виявлені користувачами невірні підрахунки та появи спонтанних витрат. Також маємо відмітити, що цей застосунок є складним та не зручним у користувацькому використанні. Наостанок необхідно відмітити, що, враховуючи всі перелічені недоліки, розробники не розвивають свій проект та вносять мало змін для виправлення виявлених помилок.

У підрозділі 1.2 «Вибір стека технологій» виконаний аналіз існуючих технологій для вирішення поставленої задачі. Виконано порівняння TypeScript та JavaScript та виділено переваги використання TypeScript перед JavaScript:

- TypeScript завжди вказує на помилки компіляції лише під час розробки. Через це під час виконання шанс отримати помилки дуже менший, тоді як JavaScript є інтерпретованою мовою;
- TypeScript має особливість, в тому, що вона строго-типізована та перевіряє правильність типу під час компіляції. Це недоступно в JavaScript;
- TypeScript - це не що інше, як JavaScript та деякі додаткові функції, тобто функції ES6. Можливо, він не підтримується у вашому цільовому браузері, але компілятор TypeScript може також скомпілювати файли .ts у ES3, ES4 та ES5.

Виконано порівняння гібридного та нативного застосунків, а також виділено переваги використання гібридного застосунку перед нативним:

- замість того, щоб створювати дві програми, ви створюєте одну програму і просто трохи її налаштовуєте, щоб вона працювала на обох платформах
- швидше в розробці
- як і у власних програмах, гібридні програми дозволяють зберегти таку ж можливість доступу до функцій пристрою.

Виконано порівняння React Native та Ionic та виділено переваги використання Ionic перед React Native:

- оскільки Ionic працює з веб-технологіями (HTML, CSS та JavaScript), він добре розуміється розробником, який не має досвіду з нативними застосунками;
- за допомогою Ionic можна створювати чудові мобільні застосунки, отримуючи доступ до власних функцій, таких як GPS, карти чи аудіо;
- за допомогою Ionic ви по суті створюєте веб-програми, які можуть стати мобільними додатками та завантажені у Google Play та Apple App Stores;
- має хорошу документацію, оскільки це компанія, що спеціалізується на створенні інструментів, які допомагають компаніям та розробникам створювати мобільні додатки.

Після аналізу існуючих технологій обрано наступні технології: Ionic, Angular та Firebase для зберігання даних, а саме Firebase Auth та Cloud Firestore.

У підрозділі 1.3 «Вимоги до програмного забезпечення» виділені основні вимоги до програмного застосунку:

1. Повинна бути можливість авторизації та реєстрації.
2. Всі вхідні дані повинні зберігатись в хмарному сховищі, до яких користувач буде мати доступ з будь–якого мобільного пристрою, на якому встановлений мобільний застосунок та після проходження автентифікації.

3. Вхідними даними для автентифікації повинні бути електронна пошта та пароль користувача.
4. Повинна бути можливість переглядати історію витрат та доходів користувача за місяць.
5. Вихідними даними для перегляду історії історію витрат та доходів користувача повинні бути сутності витрат та доходу, їх категорія, назва, сума та дата, які прив’язані до даного користувача.
6. Повинна бути можливість додавання витрат та доходу.
7. Вхідними даними для додавання витрат та доходу повинні бути категорія, назва, сума та дата.
8. Повинна бути можливість перегляду суми доходів, витрат та баланс за місяць.
9. Повинна бути можливість додавання завдань на певний день, тиждень та місяць.
10. Вхідними даними для додавання завдань має бути текст та дата.
11. Повинна бути можливість додавання подій.
12. Повинна бути можливість перегляду статистики доходів та витрат.
13. Повинна бути можливість налаштування потрібних для користувача категорій для доходів та витрат з існуючих.
14. Повинна бути можливість виходу з мобільного застосунку.

У другому розділі – «**Архітектура, моделювання та проєктування програмного забезпечення**» розроблено та описано діаграми використання, класів, послідовностей, діяльностей, станів, компонентів та розгортання, які створенні для кращого розуміння структури розроблюваного програмного застосунку. Детально описані UML–діаграми, цілі, і основні види діаграм. Розглянуто більш детально кожна з діаграм яка використовувалася в дослідженні, а так само проаналізовані коротко діаграми які належать даним проєктом.

Завдяки діаграмам можна краще зрозуміти функції та структуру розроблюваного застосунку.

У підрозділі 2.1 «Діаграма використання» створена діаграма прецедентів, яка описує взаємодію користувача з мобільним додатком, а саме яку функціональність він може використовувати при натисканні на ті чи інші значення і як ці значення між собою пов'язані.

У підрозділі 2.2 «Діаграма класів» створена діаграма класів описує основні процеси всередині класів проєкту, які розбиті за певними критеріями, таке як додаванням подій, грошових коштів, перегляд балансу тощо.

У підрозділі 2.3 «Діаграма станів» створена діаграма станів, в якій представлений статичний стан програми з усіма вихідними функціями, з якими може взаємодіяти користувач і всі наступні процеси випливаючи з дій користувача. Для більш детального розуміння станів системи при певних діяхстворено діаграму стану додавання завдання більш детально, так як там є ряд додаткових завдань, які можна показати. так само і багато інших станів можуть бути декомпозувати, але в схемі їх занадто багато, з цього було прийнято рішення деталізувати лише дві з них.

У підрозділі 2.4 «Діаграма діяльності» створено дві діаграми діяльності. Діаграма додавання завдання, яка описує який вибір піде при використанні функції додавання завдання і його результат. Також діаграма перегляду статистики, яка так само покаже яку статистику можна переглядати і які активності при цьому відбуваються.

У підрозділі 2.5 «Діаграма розгортання» створена діаграма розгортання являє собою набір приладів і опцій а так само характеристики які необхідні для функціонування програми, а так само взаємодія цих приладів з основною системою проєкту.

У підрозділі 2.6 «Діаграма компонентів» створена діаграма компонентів проєкту покаже які компоненти необхідно організувати, їх структуру та типи а так само виконавчий файл який їх організовує.

У підрозділі 2.7 «Діаграма послідовності» побудовані діаграми послідовності основних елементів системи, а саме додавання завдання і перегляд статистики з уточненням що перегляд буде стосуватися витрати.

У третьому розділі – «**Програмна реалізація та опис користувацького інтерфейсу застосунку**» розглянуто з яких класів та методів складається програмна реалізація, та за що відповідає кожен з них. Також було забезпечено базу даних та виконання різних операції з даними, які в ній зберігаються. Була підключена автентифікація користувачів за допомогою Firebase.

Також описано керівництво користувача, яке містить в собі пояснення до всіх необхідних функцій застосунку та полегшує та пришвидшує ознайомлення з даним застосунком. Сам застосунок створений з інтуїтивним інтерфейсом, що забезпечує приємний користувацький досвід.

У підрозділі 3.1 «Реалізація складових застосунку» були виділені необхідні сторінки для розроблюваного застосунку, а саме:

- авторизація;
- реєстрація;
- сторінки для перегляду завдань та подій на день, тиждень та місяць;
- сторінка для додавання події на певний період;
- сторінка для додавання завдань на певній період;
- категорії витрат та доходу;
- сторінка для додавання та редагування витрат та доходу;
- сторінка для перегляду інформації про додані витрати та доходи;
- перегляд статистики витрат та доходів.

Також розглянуто структуру кожного класу застосунку, за що відповідає кожний створений метод та за відображення яких складових відповідає певний клас.

У підрозділі 3.2 «Взаємодія із БД firebase» описано спосіб підключення створеного проєкту до firebase, налаштування автентифікації користувачів, щоб користувач мав доступ до свої даних після введення адреси електронної пошти і паролю, та реалізовано код для роботи з базою даних. Описано структуру класу для роботи з firebase та за що відповідає кожен з методів даного класу.

У підрозділі 3.3 «Опис користувацького інтерфейсу» описано інтерфейс користувача, що допомагає швидше ознайомити та краще зрозуміти як користуватися всіма функціями розробленого застосунку.

ВИСНОВКИ

Під час виконання кваліфікаційної роботи розглянуті декілька аналогів мобільних застосунків до розроблювального застосунку, а саме: «IsoTimer», «TickTick», «1Money» та «Monefy». Дані застосунки відрізняються та мають різні підходи для розв’язання проблеми. Під час аналізу раніше згаданих застосунків розглянуті їх функціонали та приклади інтерфейсу для розробки власних ідей задля поліпшення функціоналу та користувацького досвіду розроблювального мобільного застосунку.

Також переглянуті технології Ionic, Angular та Firebase, які використовувались для створення мобільного застосунку, та їх функціональні можливості. Кожна з розглянутих технологій використовує найсучасніші методи розробки програмного забезпечення. Для зберігання різних вхідних даних створена та підключена база даних , яка дає змогу на виконання різних операції з даними, які в ній зберігаються. Для збереження даних за кожним користувачем підключена автентифікація користувачів за допомогою Firebase.

Детально описані UML-діаграми, цілі, і основні види діаграм. Розглянуто більш детально кожна з діаграм яка використовувалася в дослідженні, а так само були проаналізовані коротко діаграми які належать до даного проєкту.

В результаті створений мобільний застосунок, який вирішує поставлені задачі та відповідає усім складеним вимогам, та описано інтерфейс користувача, що допомагає швидше ознайомити та краще зрозуміти як користуватися всіма функціями розробленого застосунку.