**Software Requirements Specification**

*Мобільний додаток ExpensesTracker для фінансового моніторингу*

Розробники**:**

* Щербинський Дмитро
* Абд Ель Вахаб Юсіф
* Ніколенко Дмитро

1. **Інформація про проєкт**

**Вступ**

**Мета застосунку**

Створення зручного інструменту для ефективного управління особистими фінансами, який допоможе користувачам:

* Автоматизувати облік фінансових операцій
* Отримувати детальну аналітику витрат
* Підвищити фінансову грамотність та контроль над бюджетом
* Спростити процес моніторингу та аналізу персональних фінансів

**Основні проблеми, які вирішує застосунок**

* Складність ручного обліку фінансових операцій
* Брак актуальної та структурованої інформації про витрати
* Необхідність постійного моніторингу фінансових потоків
* Потреба в зручному інструменті для аналізу та планування бюджету

**Контекст проєкту**

Додаток розроблюється для допомоги користувачам у:

* Автоматичному імпорті банківських транзакцій
* Веденні ручного обліку фінансів
* Генерації детальної фінансової статистики
* Підвищенні фінансової грамотності

1. **Загальний опис системи**

**Архітектура системи**  
Додаток використовує клієнт-серверну архітектуру:

1. Клієнтська частина: Мобільний додаток (Android) на Kivy

2. Серверна частина:

* REST API для обробки запитів від клієнта
* WebHook-ендпоінти для отримання транзакцій від Monobank
* База даних для зберігання транзакцій та аналітики
* Сервер автентифікації

**Цільова аудиторія**

* Приватні особи віком 18-45 років
* Молоді професіонали
* Підприємці та фрілансери
* Користувачі Monobank

**Бізнес-цілі**

* Спрощення процесу фінансового моніторингу
* Надання зручного інструменту для аналізу витрат
* Підвищення фінансової дисципліни користувачів

1. **Функціональні вимоги**

**Аутентифікація**

* Авторизація через Monobank API або без інтеграції (введення пошти, паролю, імя/токену)
* Посилання на сайт АПІ монобанку в вікні реєстрації для зручного взяття токену
* Безпечне зберігання токену користувача
* Можливість виходу з облікового запису

**Транзакції**

* Імпорт транзакцій через MonobankAPI
* Автоматичне отримання транзакцій через WebHook
* Ручне додавання транзакцій
* Редагування та видалення транзакцій
* Підтримка різних типів транзакцій (витрати, доходи)
* Фільтрація транзакцій за різними характеристиками (дата, сума транзакції, пошук по категорії/опису, дохід/витрата)

**WebHook-інтеграція**:

* Серверна частина надає HTTPS-ендпоінт для отримання транзакцій від Monobank
* Валідація та обробка вхідних WebHook-запитів
* Шифрування даних при передачі між Monobank і сервером
* Механізм повідомлень клієнту про нові транзакції (push-сповіщення)

**Аналітика**

* Перегляд основної статистики (min, max, mean і т.д.)
* Генерація графіків розподілу витрат
* Статистика витрат за категоріями
* Динаміка витрат за часовими інтервалами
* Перегляд балансу рахунку
* Можливість обирати інтервал та категорії

**Серверна частина**

1. Функціонал сервера:

* Обробка WebHook від Monobank:
* Підтвердження джерела запитів (IP-фільтрація, сигнатури)
* Парсинг JSON-даних транзакцій
* Нормалізація даних для зберігання
* 2. Синхронізація з клієнтом:
* API для запиту історії транзакцій

3. Безпека:

* Rate limiting для API
* JWT-автентифікація
* Резервне копіювання даних

1. **Нефункціональні вимоги**

**Продуктивність**

* Час завантаження інтерфейсу не більше 2 секунд
* Швидка обробка транзакцій (до 500 записів)
* Оптимізовані запити до бази даних
* Обробка до 1000 WebHook-запитів/хвилину
* Затримка відповіді API < 500 мс

**Безпека**

* Шифрування токену Monobank та паролю
* Захист персональних фінансових даних
* Попередження дублювання транзакцій
* Безпечне локальне зберігання даних

**Надійність**

* Автоматичне відновлення при збоях API
* Кешування даних
* Механізм повторної синхронізації
* 99.9% uptime сервера
* Реплікація бази даних

**Зручність користування**

* Інтуїтивний адаптивний інтерфейс
* Підтримка темної та світлої теми
* Вбудовані підказки та навігація

**Сумісність**

* Підтримка Android 7.0 та вище
* Адаптація під різні роздільні здатності екрану
* Сумісність з останніми версіями Kivy

**Масштабованість**

* Docker-контейнеризація
* Автомасштабування при навантаженні при навантаженні

**5. Пріоритизація вимог (MoSCoW)**

**Must Have (Обов'язкові функції)**

* Аутентифікація через Monobank API
* Імпорт транзакцій через Monobank API
* Автоматичне отримання транзакцій через WebHook
* Ручне додавання транзакцій
* Базова аналітика витрат:
  + Графіки розподілу витрат
  + Статистика витрат за категоріями
  + Динаміка витрат за часовими інтервалами
* Перегляд балансу рахунку
* Редагування та видалення транзакцій
* Підтримка різних типів транзакцій (витрати, доходи)
* HTTPS WebHook-ендпоінт
* Механізм автентифікації запитів
* API для синхронізації з клієнтом

**Should Have (Важливі функції)**

* Фільтрація та сортування транзакцій
* Підтримка різних часових інтервалів аналітики
* Підтримка вибору аналітики по категоріям

**Could Have (Бажані функції)**

* Бюджет користувача
* Перегляд курсів валют
* Встановлення лімітів на витрати
* Моніторинг регулярних виплат
* Додавання, редагування та видалення власних категорій витрат
* Експорт фінансових звітів
* Налаштування персонального інтерфейсу

**Won't Have (Функції для майбутніх версій)**

* Інтеграція штучного інтелекту (ChatGPT-подібний фінансовий асистент)
* Прогнозування майбутніх витрат на основі машинного навчання
* Інтеграція з криптовалютними гаманцями
* Автоматичне інвестиційне планування
* Повноцінна система фінансових рекомендацій з використанням складних алгоритмів штучного інтелекту

**6. Обмеження та припущення**

* Залежність від стабільності Monobank API
* Необхідність підтримки актуальної версії бібліотек
* Потреба в регулярних оновленнях безпеки
* Використання безкоштовного хмарного рівня (наприклад, AWS Free Tier) для сервера
* Обмеження Monobank API на кількість запитів

**7. Технологічний стек**

**Клієнтська частина** (Мобільний додаток)

1. Мова програмування: Python

2. Фреймворк для UI: Kivy/KivyMD (для крос-платформового інтерфейсу)

3. Бібліотеки для роботи з API:

* requests (HTTP-запити до сервера)

4. Локальна база даних: PostgreSQL (кешування даних SQLite)

5. Безпека:

* cryptography (шифрування токенів)
* Secure Storage (Android Keystore для конфіденційних даних)

**Серверна частина** (Backend)

1. Мова програмування: Python

2. Фреймворк: Flask

3. База даних: PostgreSQL (основна база)

4. WebHook-обробник: HTTPS-endpoint (перевірка підписів Monobank)

5. Автентифікація:

* JWT (OAuth2 для безпеки)
* Rate Limiting (fastapi-limiter)

6. Хмарний хостинг:

* DigitalOcean
* Docker + Docker Compose (контейнеризація)

**Інтеграції**

Monobank API:

* Офіційний API для імпорту транзакцій
* WebHook для автоматичних оновлень

**Інструменти розробки**

Версіонування коду: Git + GitHub

**8. Висновки**

Даний SRS-документ визначає повний набір вимог до мобільного додатку ExpensesTracker, окреслює його функціональність, технічні характеристики та пріоритети розвитку.

**Software Requirements Specification**

**Mobile Application: ExpensesTracker for Financial Monitoring**

**Developers:**

* Dmytro Shcherbynskyi
* Abd El Wahab Yousif
* Dmytro Nikolenko

**1. Project Information**

**Introduction**

**Purpose of the Application**

Creating a convenient tool for effective personal finance management that will help users:

* Automate financial transaction accounting
* Obtain detailed expense analytics
* Increase financial literacy and budget control
* Simplify the process of monitoring and analyzing personal finances

**Key Problems Solved by the Application**

* Complexity of manual financial transaction accounting
* Lack of current and structured information about expenses
* Constant need to monitor financial flows
* Need for a convenient tool for budget analysis and planning

**Project Context**

The application is being developed to help users:

* Automatically import bank transactions
* Keep manual records of finances
* Generate detailed financial statistics
* Improve financial literacy

**2. General System Description**

**Target Audience**

* Individuals aged 18–45
* Young professionals
* Entrepreneurs and freelancers
* Monobank users

**Business Goals**

* Simplify the financial monitoring process
* Provide a convenient tool for expense analysis
* Improve users' financial discipline

**3. Functional Requirements**

**Authentication**

* Authorization via Monobank API or without integration (email, password, name/token input)
* Link to Monobank bot in the registration window for easy token retrieval
* Secure storage of user token
* Ability to log out of the account

**Transactions**

* Automatic transaction retrieval via Monobank API
* Manual transaction addition
* Editing and deleting transactions
* Support for different transaction types (expenses, income)
* Filtering transactions by various characteristics (date, amount, category/description search, income/expense)

**Analytics**

* Viewing basic statistics (min, max, mean, etc.)
* Generating expense distribution charts
* Expense statistics by categories
* Expense dynamics across time intervals
* Viewing account balance
* Ability to select intervals and categories

**Additional Features**

* Viewing current exchange rates

**4. Non-Functional Requirements**

**Performance**

* Interface load time no more than 2 seconds
* Fast transaction processing (up to 500 records)
* Optimized database queries

**Security**

* Encryption of Monobank token and password
* Protection of personal financial data
* Prevention of transaction duplication
* Secure local data storage

**Reliability**

* Automatic recovery from API failures
* Data caching
* Re-synchronization mechanism

**Usability**

* Intuitive adaptive interface
* Support for dark and light themes
* Built-in tips and navigation

**Compatibility**

* Support for Android 7.0 and higher
* Adaptation to various screen resolutions
* Compatibility with the latest Kivy versions

**5. Requirements Prioritization (MoSCoW)**

**Must Have**

* Authentication via Monobank API
* Automatic transaction retrieval
* Manual transaction addition
* Basic expense analytics:
  + Expense distribution charts
  + Expense statistics by categories
  + Expense dynamics across time intervals
* Viewing account balance
* Editing and deleting transactions
* Support for different transaction types (expenses, income)

**Should Have**

* Currency rate viewing
* Transaction filtering and sorting
* Support for different analytics time intervals
* Support for category-based analytics

**Could Have**

* User budget
* Setting expense limits
* Monitoring of regular payments
* Adding, editing, and deleting custom expense categories
* Exporting financial reports
* Personal interface customization
* Automatic transaction import via WebHookURL

**Won’t Have (for Future Versions)**

* Integration of artificial intelligence (ChatGPT-like financial assistant)
* Forecasting future expenses using machine learning
* Integration with cryptocurrency wallets
* Automatic investment planning
* Full-featured financial recommendation system using complex AI algorithms

**6. Limitations and Assumptions**

* Dependence on Monobank API stability
* Need to support up-to-date library versions
* Regular security updates required

**7. Technological Stack**

**Client Side (Mobile App)**

1. Programming Language: Python

2. UI Framework: Kivy / KivyMD (for cross-platform interface)

3. API Interaction Libraries:

* requests (for HTTP requests to the backend)

4. Local Database: PostgreSQL, SQLite (used for offline data caching)

5. Security:

* cryptography (for secure token encryption)
* Android Keystore (for storing sensitive data locally via Secure Storage)

**Server Side (Backend)**

**Programming Language:** Python

1. Framework: Flask (for REST API development)

2. Database: PostgreSQL (primary storage for transactions and analytics)

3. WebHook Handler: HTTPS endpoint for receiving Monobank WebHook updates (signature verification and IP filtering).

4. Authentication & Security:

* JWT (OAuth2 for secure user authentication)
* Rate Limiting (fastapi-limiter)

5. Cloud Hosting:

* DigitalOcean
* Docker + Docker Compose (containerization for scalability and deployment)

**Integrations**

Monobank API:

* Official Monobank API for importing transactions
* WebHook mechanism for automatic updates

**Development Tools**

Version Control: Git + GitHub

**8. Conclusions**

This SRS document defines the complete set of requirements for the ExpensesTracker mobile application, outlines its functionality, technical characteristics, and development priorities.