# **Renovation Cost Tracker – Dokumentacja architektury wielowarstwowej**

## **1. Cel i zakres aplikacji**

REST‑owe API w FastAPI służące do rejestrowania i monitorowania kosztów remontu mieszkania. Aplikacja udostępnia trzy główne warstwy **Domain → Application → Infrastructure** (warstwa prezentacji = kontrolery FastAPI). Dane przechowywane są w PostgreSQL, a w testach mogą być uruchamiane na SQLite.

## **2. Wymagania funkcjonalne**

| **Id** | **Wymaganie** | **Priorytet** |
| --- | --- | --- |
| F‑1 | Rejestracja użytkownika (e‑mail, hasło) | Wysoki |
| F‑2 | Logowanie i otrzymanie tokenu JWT | Wysoki |
| F‑3 | Utworzenie projektu remontowego (nazwa, budżet) | Wysoki |
| F‑4 | Dodanie wydatku do projektu (kwota, kategoria, dostawca, data, opis, załącznik URL) | Wysoki |
| F‑5 | Edycja i usuwanie wydatku | Średni |
| F‑6 | Lista wydatków z filtrami (data, kategoria, dostawca) | Wysoki |
| F‑7 | Podsumowania projektu: łączny koszt, koszt wg kategorii, pozostały budżet | Wysoki |
| F‑8 | Eksport wydatków projektu do pliku CSV | Średni |
| F‑9 | Import wydatków z CSV (opcjonalnie) | Niski |

## 

## 

## 

## **3. Wymagania niefunkcjonalne**

* **Wydajność:** czas odpowiedzi API ≤ 200 ms dla zapytań do cache.
* **Bezpieczeństwo:** szyfrowanie TLS, JWT, haszowanie haseł bcrypt, zabezpieczenia OWASP Top 10.
* **Utrzymywalność:** pokrycie testów ≥ 85 %, kod zgodny z PEP 8 i SOLID.
* **Portowalność:** obraz docker‑compose (FastAPI + PostgreSQL) oraz możliwość podmiany bazy na SQLite w testach.
* **Skalowalność:** warstwy pozwalają na łatwe dodanie np. integracji z bank API.