**DAM- Sprint 1**

**Nume proiect:** SpendWise **Grupa**: mSIA2

**Echipa**: Burcă Iulian-Ștefan, Gavril Ionela-Mălina, Pintecan Alexandru

**1. Tema proiect**:

Aplicație de gestionare a veniturilor și cheltuielilor personale pe categorii definite și generarea statisticilor pentru cheltuielile într-o anumită perioadă de timp selectată de utilizator.

**2. Roluri:**

* **Product Owner & QA:** Gavril Ionela-Mălina
* **Full Stack Developer & DevOps:** Burcă Iulian-Ștefan
* **Developer & Tech Lead & UI/UX Designer:** Pintecan Alexandru

**3. Backlog inițial**

**Funcționalități:**

* Crearea conturilor
* Editarea profilului
* Adăugarea veniturilor și cheltuielilor
* Adăugarea recurentă a anumitor venituri și cheltuieli (ex: salariu, facturi)
* Generarea statisticilor
* Editarea veniturilor și cheltuielilor
* Filtrarea cheltuielilor/veniturilor în istoricul de activitate (ex: doar cheltuielile dintr-o perioadă de timp pe mancare sau alte lucruri)
* Export statistici în anumite formate (ex: PDF, XLSX, CSV etc.)
* Setarea unor limite pe categorii (pentru a nu depăși anumite sume impuse într-o anumită perioadă de timp)
* Compararea cheltuielilor de la luna la luna (prin statistici)
* \*\*\* Imposibilitatea de a cheltui mai mulți bani decât veniturile din luna respectivă și economiile din lunile anterioare

**User stories**

***Feature: Gestionare cont***

Scenario: Creare cont

Given că sunt user

When apasă butonul de înrolare

And completează câmpurile obligatorii: nume, prenume, e-mail și parolă

And apasă butonul de salvare date

Then utilizatorul i se alocă un userID și BudgetAccountID unic în aplicație

Scenario: Editare profil

Given că userul este autentificat

When apasă butonul de editare profil

And actualizează unul dintre câmpuri

And apasă butonul de salvare

Then modificarea se actualizează în baza de date

***Feature: Gestionarea veniturilor și cheltuielilor***

Scenario: Adăugare venit sau cheltuială

Given că userul este autentificat

When apasă butonul de adăugare tranzacție

And completează suma, tipul (venit/cheltuială), selectează categoria și data efectuării

Then tranzacția apare în lista de tranzacții

Scenario: Editare venit sau cheltuială

Given că tranzacția este adăugată și userul este autentificat

When apasă butonul de editare tranzacție

And actualizează unul dintre câmpuri

And apasă butonul de salvare

Then modificarea se actualizează în baza de date

Scenario: Ștergere venit sau cheltuială

Given că tranzacția este adăugată și userul este autentificat

When apasă butonul de editare tranzacție

And apasă butonul de ștergere tranzacție

Then tranzacția se șterge din lista de tranzacții

Scenario: Filtrare tranzacții

Given că userul este autentificat

When apasă butonul de filtrare tranzacții

And completează cel puțin unul dintre câmpurile: perioadă, tip, sumă sau categorie

Then se afișează lista de tranzacții filtrate conform criteriilor

Scenario: Adăugare limită pe categorie

Given că userul este autentificat

When apasă butonul de adăugare limită

And selectează categoria și suma totală limitată pe categorie

Then când se depășește limita setată, userul primește un mail de avertizare

***Feature: Generare statictici***

Scenario: Comparare cheltuieli

Given că userul este autentificat

When apasă butonul de generare statictici

And selectează tipul: analiză lunară

And selectează lunile luate în calcul

Then se afisează sumarul lunilor selectate: cheltuieli totale, grupare pe categorii,

total cheltuieli pe categorii și procente lunare

Scenario: Export statictici

Given că userul este autentificat

When apasă butonul de generare statictici

And selectează tipul de statictică

And apasă butonul de export PDF

Then se descarcă statictica pe device-ul de pe care userul este autentificat în formatul selectat

**4. ADR 001 – Alegerea stack-ului tehnologic**

Pachete de tehnologii luate în calcul pentru dezvoltarea aplicației:

* Java+ SpringBoot+ PostgreSQL+ React
* Java+ SpringBoot+ PostgreSQL+ Angular
* .Net Core+ MongoDB+ Angular
* Python+ Flask+ SQL Server+ Vue.js

Pentru dezvoltarea aplicației s-a ales următoarea combinație: **Java+ SpringBoot+ PostgreSQL+ React** pentru că:

* Java este un limbaj de programare cunoscut de majoritatea membrilor din echipă;
* Java, SpringBoot, PostgreSQL și React reprezintă un ecosistem matur și stabil, care oferă o documentație riguroasă;
* SpringBoot este specializat în dezvoltarea aplicațiilor multistrat (MVC, microservicii, stratificată), codul rezultat fiind scalabil, ușor de testat,și modular;
* PostgreSQL este o bază de date open-source, robustă și scalabilă, potrivită pentru interogări complexe;
* React prezintă o integrare ușoară cu backend-ul, în timp ce un frontend dezvoltat cu Angular necesită un setup mai complex;
* React are o curbă de învățare mai lină decât Angular, având o flexibilitate mult mai mare decât Angular.

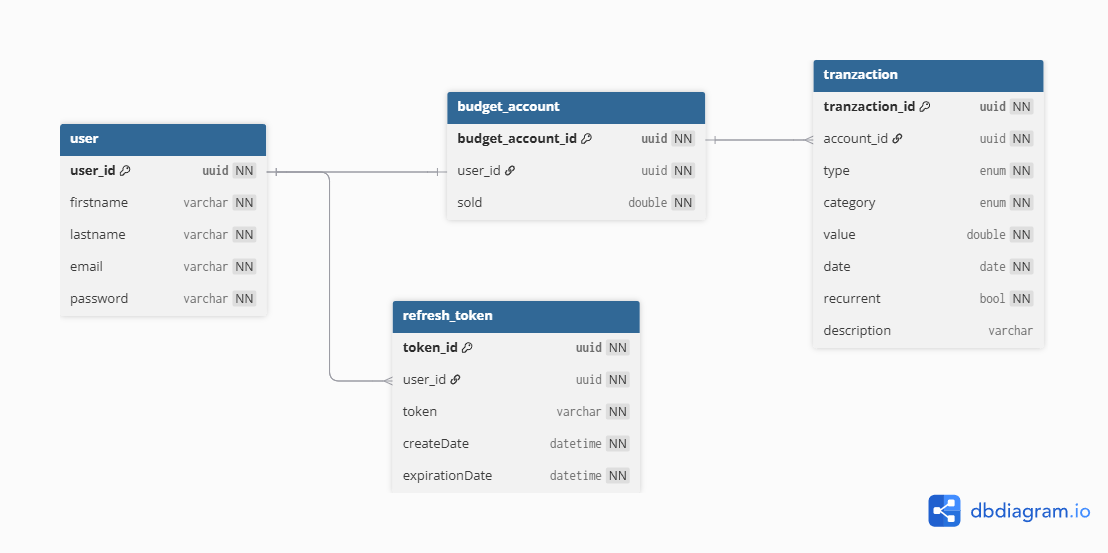
**ADR 002 – Stil arhitectural abordat**

Se disting mai multe arhitecturi posibile ale unei aplicații web, printre care amintim: Monolith, Client-Server, Modular Monolith, arhitectura bazată pe microservicii și arhitectura SPA (Single-Page Application). Având în vedere faptul că timpul este limitat, iar echipa este formată doar din 3 membri, s-a ales o arhitectura **Modular Monolith,** organizată pe mai multe straturi logice deoarece:

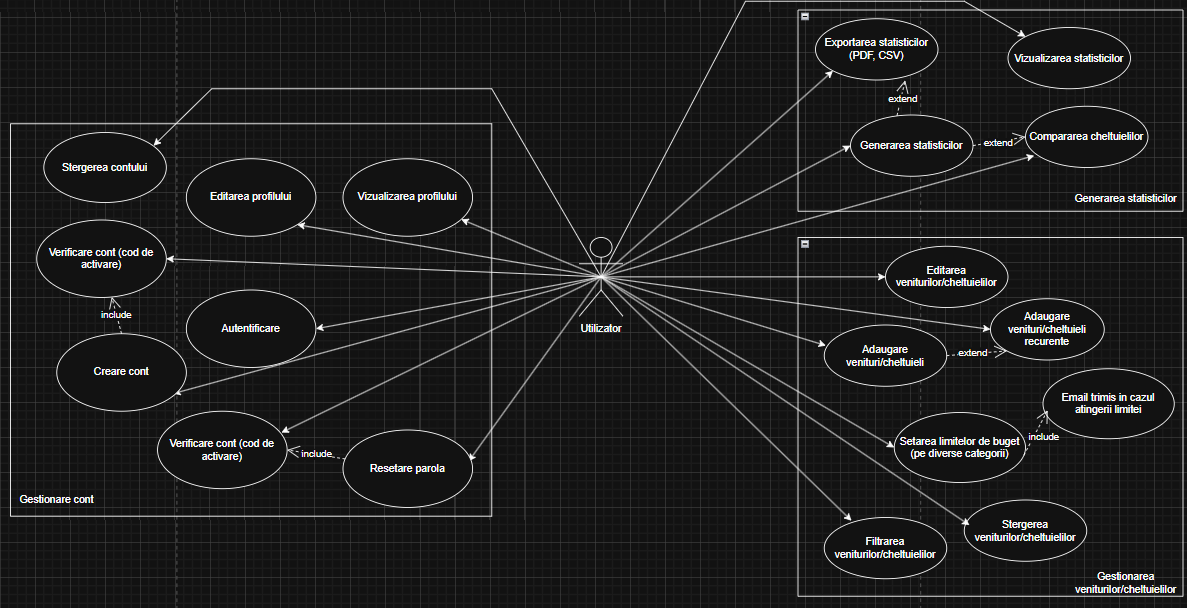
* oferă o separare clară pe straturi funcționale;
* oferă o scalabilitate bună a aplicației- permite extinderea ulterioară a aplicației ușor;
* cod organizat (modular), ușor de reutilizat- ce permite o testare ușoară;
* simplifică dezvoltarea, oferind o implementare rapidă.

**5. Diagrame:**

* **DB**



* **Use-case**



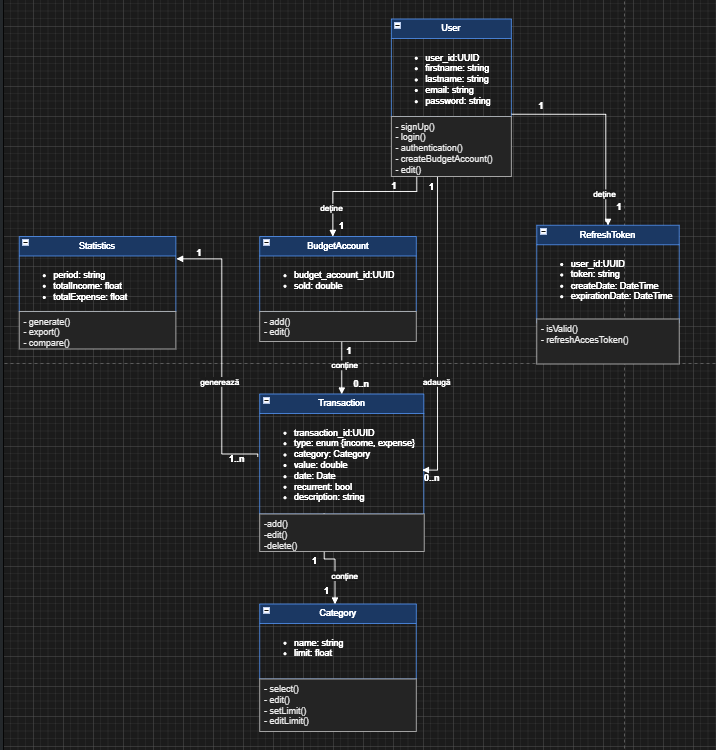
* **UML**

Actorii principali:

Utilizator - user: creează contul, adaugă venituri și cheltuieli și generează statistici.

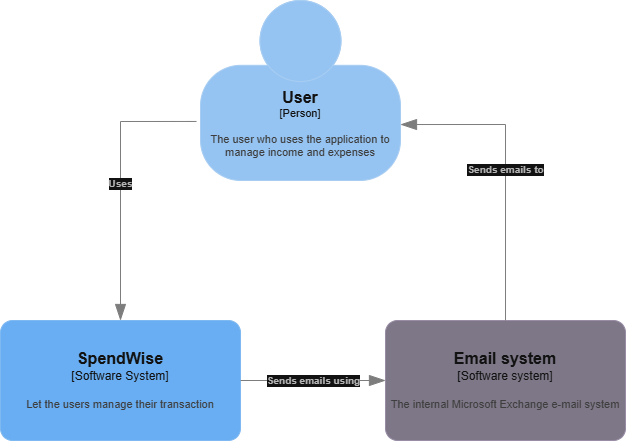
Sistem: Aplicație de gestionare a bugetului personal.

* Conține module pentru venituri, cheltuieli, filtrare pe categorii, limitare pe categorii de cheltuieli, generarea statisticilor: comparații, rapoarte periodice, ponderi etc. și exportul în diferite formate (CSV, XLSX, PDF).
* Folosește datele din baza de date PostgreSQL.



**- C4**

**- Nivelul 1**



**- Nivelul 2**

