SRS + SDD Habits - Gamified Habit Tracker

1. Descriere generala

Habits este un server REST API care permite gestionarea unui sistem de task-uri gamificat, orientat pe formarea si urmarirea obiceiurilor. Utilizatorii pot defini habits (obiceiuri), dailies (task-uri zilnice) si to-dos (task-uri unice). Aplicatia ofera operatii CRUD pentru entitatile principale si include functionalitati precum acordarea de XP si gold la completarea task-urilor, penalizarea HP pentru dailies ratate si gestionarea de recompense. Sistemul foloseste .NET 8, Entity Framework Core, SQLite si Docker Compose.

2. User Stories

• US1 - Gestionare utilizatori (CRUD): Ca user, vreau sa pot crea, edita si sterge contul meu pentru a-mi administra profilul in aplicatie.

• US2 - Gestionare task-uri (CRUD): Ca user, vreau sa pot crea, edita, lista si sterge task-uri de tip habit, daily si todo pentru a-mi organiza activitatile.

• US3 - Finalizare task si actualizare statistici: Ca user, vreau sa pot marca un task ca efectuat pentru a primi XP si gold si a-mi vedea progresul.

• US4 - Penalizare pentru dailies ratate: Ca user, vreau sa fiu penalizat la HP daca nu imi termin dailies pentru a fi motivat sa le fac.

• US5 - Gestionare recompense (CRUD): Ca user, vreau sa pot vedea si cumpara recompense folosind gold pentru a-mi putea rasplati progresul.

• US6 - Filtrare si organizare task-uri: Ca user, vreau sa imi pot filtra task-urile dupa tip si stare (active, complete, arhivate) pentru a vedea rapid ce am de facut.

• US7 - Vizualizare progres si statistici: Ca user, vreau sa pot vedea XP-ul, level-ul, HP-ul si gold-ul curent pentru a-mi urmari evolutia.

• US8 - Istoric completari: Ca user, vreau sa pot vedea un istoric al completarilor de task-uri pentru a intelege cum mi-au evoluat obiceiurile in timp.

3. Cerinte functionale

Aplicatia Habits trebuie sa permita operatii CRUD complete pentru entitatile principale: Users, TaskTypes, Tasks, Rewards, TaskCompletionHistory si UserRewardPurchases.

Sistemul trebuie sa permita gestionarea task-urilor: creare, actualizare, stergere, arhivare si listare, cu suport pentru tipuri diferite (habit, daily, todo) si dificultati (easy, medium, hard).

Aplicatia trebuie sa ofere un endpoint pentru finalizarea unui task, care sa calculeze XP si gold in functie de tipul si dificultatea task-ului si sa actualizeze statistica user-ului.

Pentru dailies ratate, sistemul trebuie sa poata aplica penalizari de HP (manual printr-un endpoint sau automat printr-un job programat, in functie de cerintele proiectului).

Aplicatia trebuie sa permita gestionarea recompenselor: creare, listare, modificare si dezactivare, precum si cumpararea lor de catre utilizatori folosind gold.

Sistemul trebuie sa expuna un endpoint pentru vizualizarea progresului utilizatorului (XP, Gold, Level, HP) si pentru listarea istoricului de completari.

Aplicatia trebuie sa ofere filtre pentru listarea task-urilor (dupa tip, stare si, optional, dificultate) pentru a simplifica organizarea.

4. Arhitectura

Controllers → Services → Entity Framework Core → SQLite

Aplicatia ruleaza prin Docker Compose, cu un serviciu pentru API (.NET 8 Web API) si un serviciu pentru baza de date (SQLite, cu volum atasat).

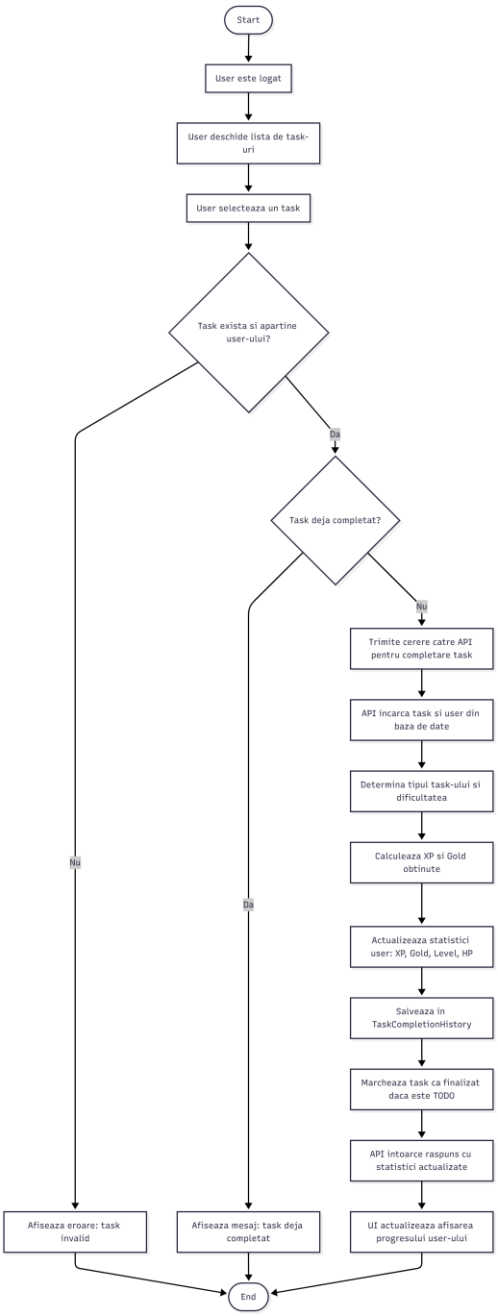

Figure Diagrama de activitate - Finalizare task

5. Flux simplificat - Finalizare task

1. User-ul trimite o cerere POST /tasks/{id}/complete catre API pentru a marca un task ca finalizat.

2. Controller-ul valideaza cererea, verifica daca task-ul exista si daca apartine user-ului.

3. Service-ul incarca task-ul si user-ul, determina tipul task-ului si dificultatea si calculeaza XP si gold de acordat, precum si eventuale modificari de HP.

4. Service-ul actualizeaza entitatea UserStats (XP, Gold, Level, HP), marcheaza task-ul ca finalizat daca este un todo si inregistreaza o intrare in TaskCompletionHistory.

5. API-ul salveaza modificarile in baza de date si intoarce raspuns catre client cu noile statistici ale user-ului si starea actualizata a task-ului.

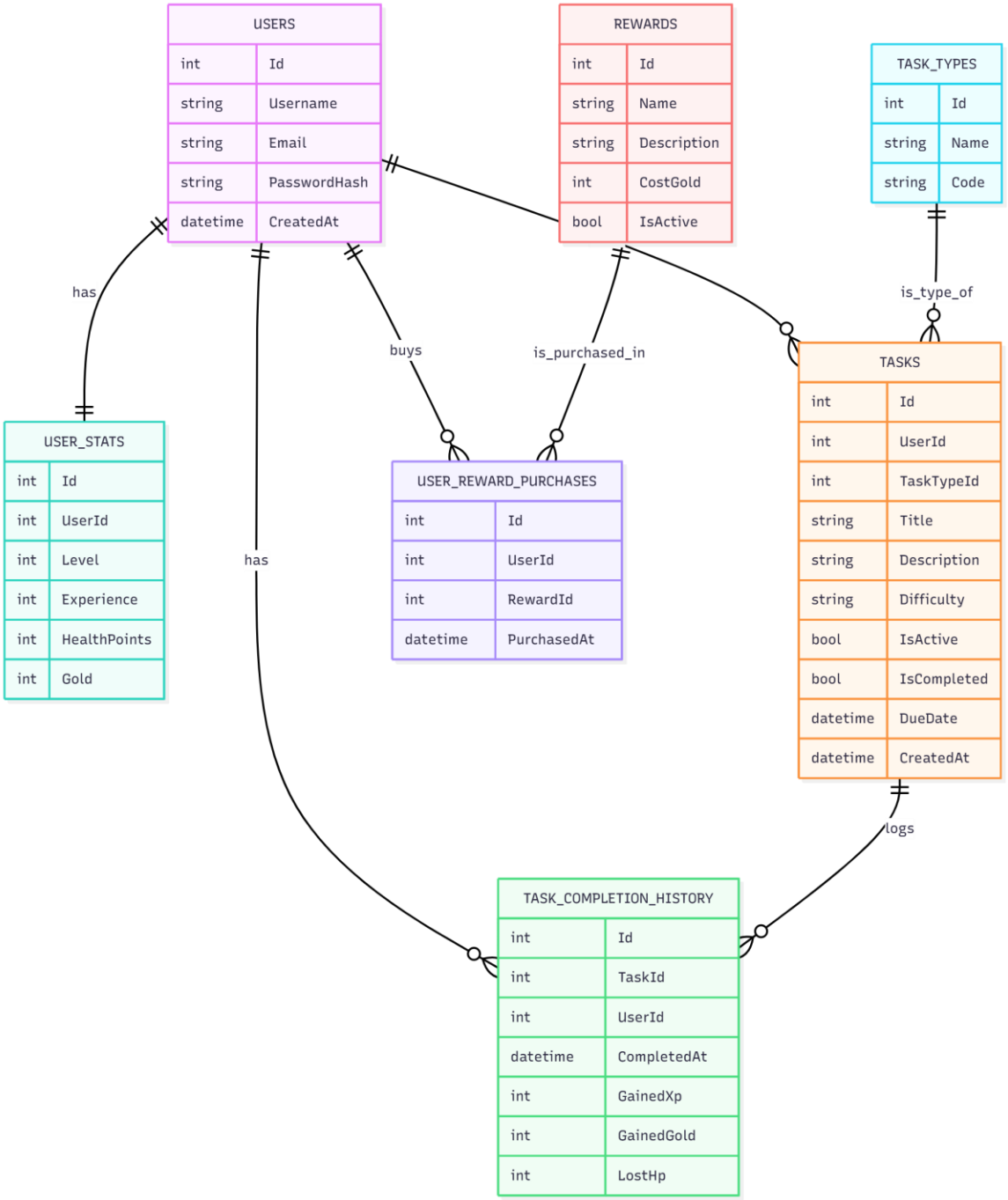

Figure Schema bazei de date (ERD)

6. Schema bazei de date (text simplificat)

User (1) → (1) UserStats

User (1) → (M) Tasks

TaskType (1) → (M) Tasks

Task (1) → (M) TaskCompletionHistory

User (1) → (M) TaskCompletionHistory

User (1) → (M) UserRewardPurchases

Reward (1) → (M) UserRewardPurchases

7. Tehnologii

- .NET 8 Web API

- C#

- Entity Framework Core

- SQLite

- Docker & Docker Compose