# Studio e Analisi della Funzione — Finhub Credit Bank

Questo documento descrive in modo operativo la funzione del sistema sviluppato: ambito, attori, requisiti, casi d’uso, flussi, regole e criteri di accettazione. È allineato ai file del progetto (modelli, viste e schema OpenAPI).

## 1. Scopo e ambito

Realizzare un servizio di banking digitale API‑based che consenta all’utente di: registrarsi e autenticarsi, gestire conti, consultare movimenti, eseguire trasferimenti (bonifici) e impostare obiettivi di risparmio. Il sistema espone API REST consumabili dal front‑end web e da client terzi.

## 2. Attori e stakeholder

• Utente (Cliente): registra/attiva l’account, accede, gestisce conti e transazioni, usa la rubrica, crea e alimenta goals.  
• Amministratore (solo gestione tecnica e demo): crea superuser, può inizializzare saldi via admin.  
• Servizi esterni: email per attivazione; (facoltativo) integrazioni future per KYC/pagamenti.

## 3. Glossario minimo

• Conto (BankAccount): risorsa con IBAN, saldo, valuta, intestata a un utente.  
• Transazione (Transaction): movimento contabile; importo positivo=entrata, negativo=uscita.  
• Bonifico/Trasferimento: operazione che addebita un conto e accredita un IBAN (interno/esterno).  
• Goal (GoalsSaving): obiettivo di risparmio con importo target e stato di avanzamento.  
• JWT: token usato per autenticare le richieste alle API protette.

## 4. Funzioni del sistema (panoramica)

F1. Onboarding: registrazione (email+password) e attivazione via email.  
F2. Login/Logout JWT e verifica/refresh token.  
F3. Gestione conti: elenco e dettaglio conti dell’utente (IBAN, saldo, valuta).  
F4. Transazioni: elenco/storico movimenti con ordinamento per data.  
F5. Trasferimenti: esecuzione bonifico verso IBAN interno/esterno con controlli.  
F6. Rubrica: creazione/lista/eliminazione beneficiari (nome, IBAN, email, città).  
F7. Estratti conto: consultazione movimenti per statement mensile.  
F8. Goals: creazione, modifica, cancellazione e alimentazione obiettivi di risparmio.

## 5. Requisiti funzionali (selezione)

F‑01 Registrazione utente con email e password; invio link di attivazione.  
F‑02 Attivazione account con token.  
F‑03 Login con rilascio coppia JWT (access/refresh) e verifiche token.  
F‑04 Visualizzare conti dell’utente autenticato.  
F‑05 Elencare transazioni di un conto (paginazione/ordinamento per data).  
F‑06 Eseguire un trasferimento: from\_account\_id, to\_iban, amount, description.  
F‑07 Gestire rubrica contatti (create/list/delete, unico IBAN per utente).  
F‑08 Consultare movimenti per estratto conto (mese/anno).  
F‑09 Creare/aggiornare/eliminare un goal di risparmio.  
F‑10 Aggiungere un versamento/prelievo al goal e aggiornare importo attuale.

## 6. Requisiti non funzionali

NF‑01 Sicurezza: password hashate (AbstractBaseUser), JWT per le API protette.  
NF‑02 Consistenza: aggiornamento del saldo solo all’interno di transazioni DB.  
NF‑03 Auditabilità: transazioni append‑only; storni come contro‑movimenti.  
NF‑04 Replicabilità: avvio con Docker; configurazione tramite .env.  
NF‑05 Chiarezza API: nomenclatura REST, codici HTTP coerenti, messaggi d’errore in JSON.

## 7. Casi d’uso principali

UC‑01 Registrazione e attivazione: l’utente invia email/password; riceve un link; attiva l’account.  
UC‑02 Login JWT: l’utente ottiene access/refresh token; opzionale refresh/verify.  
UC‑03 Visualizza conti: l’utente vede i propri conti (IBAN, saldo, valuta, nome).  
UC‑04 Lista transazioni: l’utente consulta lo storico, ordinato per data.  
UC‑05 Bonifico: l’utente invia un trasferimento verso un IBAN; il sistema valida e registra i movimenti.  
UC‑06 Rubrica contatti: l’utente salva un beneficiario e lo riutilizza nei bonifici.  
UC‑07 Estratto conto: l’utente consulta i movimenti dello statement mensile.  
UC‑08 Goals: l’utente crea un obiettivo, versa importi e monitora la percentuale di completamento.

## 8. Flussi principali (testo operativo)

• Bonifico (POST /api/transfer/): 1) L’utente invia from\_account\_id, to\_iban, amount (>0), description; 2) verifica proprietà del conto e saldo sufficiente; 3) se IBAN interno: doppio movimento (−amount/+amount); se esterno: registra uscita; 4) aggiornamento saldi in transazione DB; 5) esito 201/400/403/404.  
• Goals: creazione con importo\_target; versamento (DEPOSIT) che incrementa importo\_attuale; percentuale e importo\_rimanente calcolati dal modello.

## 9. Dati e vincoli essenziali

• UUID come PK; email univoca (User).  
• IBAN univoco (Accounts.iban e BankAccount.iban).  
• Transaction.amount firmato; ordering per data/created\_at.  
• Contact unique\_together(user, iban); GoalsSaving unique\_together(bank\_account, nome).

## 10. Mappatura Use Case → API

UC‑01/02: POST /auth/jwt/create/, /auth/jwt/refresh/, /auth/jwt/verify/.  
UC‑03: GET /api/accounts/, GET /api/accounts/me/.  
UC‑04: GET /api/transactions/, GET /api/transactions/{id}/.  
UC‑05: POST /api/transfer/.  
UC‑06: GET/POST /api/accounts/contacts/, DELETE /api/accounts/contacts/{id}/.  
UC‑07: GET /api/estratti-conto/{estratto\_id}/movimenti/.  
UC‑08: GET/POST /api/goals-saving/, GET/PATCH/DELETE /api/goals-saving/{id}/.

## 11. Regole di validazione e business

• Bonifico: amount > 0; saldo sufficiente; IBAN formato valido; conto mittente dell’utente.  
• Transazione: importo ≠ 0; movimenti append‑only.  
• Rubrica: IBAN unico per utente.  
• Goals: importo\_target > 0; importo\_attuale ≥ 0; update via movimenti goal.

## 12. Gestione errori e codici HTTP

400 input non valido; 401 token assente/scaduto; 403 risorsa di altro utente; 404 inesistente; 409 conflitto su external\_id (opzionale per idempotenza).

## 13. Sicurezza

Password hashate; JWT per API; permessi object‑level; dati carta solo hash (pan\_hash/cvv\_hash).

## 14. Assunzioni e vincoli

Operatività in EUR; IBAN europei; niente KYC avanzato o pagamenti istantanei; focus didattico su API e consistenza.

## 15. Rischi e mitigazioni

Doppie registrazioni: usare external\_id UNIQUE; sicurezza carte: evitare pan\_real/cvv\_real o cifrarli; coerenza saldi: transazioni DB e test di errore.

## 16. Criteri di accettazione

CA‑01 Dopo attivazione il login restituisce JWT validi; CA‑02 /api/accounts/ mostra solo conti dell’utente; CA‑03 bonifico con saldo insufficiente → 400; CA‑04 transazioni ordinate per data; CA‑05 goal aggiornabile con versamenti.

## 17. Piano di test funzionale

T‑01 Registrazione→attivazione→login; T‑02 verifica conti; T‑03 bonifico interno OK/KO; T‑04 rubrica create/list/delete; T‑05 goals create→versa; T‑06 estratto conto per mese/anno.