# Audit Implementare Registru de Casă și Registru de Bancă – Conformitate și Plan de Acțiune

## Cadrul Legislativ Actual în România

**Reglementări contabile și fiscale aplicabile:** Registrul de casă și evidența operațiunilor de bancă trebuie organizate conform Legii contabilității nr. 82/1991 și normelor în vigoare, precum și legislației specifice: OMFP 2634/2015 (care a înlocuit OMFP 3512/2008) privind documentele financiar-contabile, OMFP 2861/2009 (norme de inventariere), Legea 70/2015 (plafonarea tranzacțiilor în numerar), Codul fiscal și cerințele recente SAF-T (Declarația 406) impuse de ANAF. În esență, aceste acte cer: evidență zilnică a mișcărilor de numerar și stabilirea soldului de casă la sfârșitul fiecărei zile[[1]](https://infotva.manager.ro/articole/studii-de-caz/registru-de-casa-2023-registrul-se-intocmeste-centralizat-in-cazul-firmelor-cu-mai-multe-puncte-de-lucru-23081.html#:~:text=Nu%20exista%20nicio%20reglementare%20in,Registrul%20de%20casa%20serveste%20ca), folosirea formularelor tipizate cu conținut minim obligatoriu (denumirea unității, data, număr curent, număr document de casă, explicații, sume încasări/plăți, soldul precedent, semnături)[[2]](https://infotva.manager.ro/articole/infotva/registru-casa-23551.html#:~:text=Continutul%20minimal%20obligatoriu%20de%20informaii,al%20formularului%20este%20urmtorul), precum și respectarea plafoanelor legale pentru tranzacțiile cash și soldurile din casierie. Legea 70/2015 impune un plafon zilnic de **5.000 Lei** pentru plăți către furnizori (10.000 Lei pentru magazine tip cash&carry) și **10.000 Lei** per persoană (cumulat) pentru operațiuni de încasări/plăți între o firmă și altă entitate sau persoană fizică, interzicând fragmentarea tranzacțiilor peste aceste limite[[3]](https://www.avocatnet.ro/articol_68880/Ce-plafoane-de-incas%C4%83ri-%C8%99i-pl%C4%83%C8%9Bi-in-numerar-se-aplic%C4%83-in-2025.html#:~:text=Totodat%C4%83%2C%20opera%C8%9Biunile%20de%20%C3%AEncas%C4%83ri%20%C8%99i,c%C4%83tre%2F%20de%20la%20o%20persoan%C4%83). De asemenea, sumele de numerar deținute în casierie nu pot depăși **50.000 Lei la sfârșitul zilei** (respectiv 500.000 Lei pentru unități de tip hypermarket/cash&carry), excedentul trebuind depus la bancă în termen de 2 zile lucrătoare[[4]](https://www.avocatnet.ro/articol_68880/Ce-plafoane-de-incas%C4%83ri-%C8%99i-pl%C4%83%C8%9Bi-in-numerar-se-aplic%C4%83-in-2025.html#:~:text=Apoi%2C%20Legea%2070%2F2015%20impune%20limite,termen%20de%20dou%C4%83%20zile%20lucr%C4%83toare). Normele OMFP 2861/2009 prevăd obligativitatea inventarierii anuale a numerarului în ultima zi lucrătoare a exercițiului financiar, **confruntând soldul scriptic din registrul de casă cu monetarul (banii efectivi din casierie) și cu soldul din contabilitate**[[5]](https://infotva.manager.ro/articole/infotva/registru-casa-23551.html#:~:text=proprii%2C%20aprobate%20prin%20OMFP%20nr,2861%2F2009) – orice diferențe trebuind justificate, raportate și, dacă e cazul, imputate sau tratate fiscal corespunzător.

**Registrul de casă** este un document financiar-contabil oficial (cod 14-4-7, conform OMFP 2634/2015) ce **se întocmește zilnic**, la nivelul fiecărei casierii, pe baza documentelor justificative de încasare și plată, pentru evidența operativă a numerarului[[1]](https://infotva.manager.ro/articole/studii-de-caz/registru-de-casa-2023-registrul-se-intocmeste-centralizat-in-cazul-firmelor-cu-mai-multe-puncte-de-lucru-23081.html#:~:text=Nu%20exista%20nicio%20reglementare%20in,Registrul%20de%20casa%20serveste%20ca). Fiecare punct de încasare cu numerar trebuie să aibă propriul registru de casă; nu este permis un registru centralizat unic pentru toate casieriile unei firme[[6]](https://infotva.manager.ro/articole/studii-de-caz/registru-de-casa-2023-registrul-se-intocmeste-centralizat-in-cazul-firmelor-cu-mai-multe-puncte-de-lucru-23081.html#:~:text=Registrul%20de%20casa%20nu%20se,de%20la%20punctele%20de%20lucru). Registrul se completează de regulă în dublu exemplar (unul rămâne la casierie, unul la departamentul financiar-contabil), fie manual, fie informatizat, și la sfârșitul fiecărei zile se determină soldul de închidere al zilei, care devine soldul inițial pentru ziua următoare. Orice formular neutilizat integral în cursul zilei se barează, iar totalurile zilnice, precum și soldul final, se certifică prin semnătura casierului și a responsabilului financiar-contabil[[7]](https://infotva.manager.ro/articole/infotva/registru-casa-23551.html#:~:text=Soldul%20de%20casa%20al%20zilei,casa%20pentru%20ziua%20in%20curs)[[8]](https://infotva.manager.ro/articole/infotva/registru-casa-23551.html#:~:text=Se%20semneaza%20de%20catre%20casier,si%20a%20actelor%20justificative%20anexate). Registrul de casă servește totodată ca document de înregistrare în contabilitatea generală a operațiunilor de casă (debit/credit cont 531 “Casa” și conturi corespondente), fiind un jurnal auxiliar specializat al Registrului Jurnal general.

**Jurnalul de bancă** (evidența operațiunilor în conturi bancare) nu are un formular tipizat distinct impus de lege similar registrului de casă, însă practicile contabile cer ca toate mișcările din conturile 512 (bănci) să fie înregistrate cronologic, pe baza extraselor de cont și a documentelor bancare (ordin de plată, foaie de vărsământ etc.). Legea contabilității obligă înregistrarea operativă a tuturor operațiunilor de trezorerie, inclusiv prin conturi bancare, și reconcilierea periodică a acestora cu extrasele bancare. OMFP 2634/2015 include în lista documentelor financiar-contabile și **Extrasele de cont bancar** și impune ca soldurile din contabilitate ale conturilor de bancă să fie verificate cu extrasele (de obicei lunar). Deși nu există un “Registru de bancă” tipărit cu cerințe de formă precum registrul de casă, în practică se ține un jurnal auxiliar (sau rapoarte de extras) pentru conturile bancare, permițând urmărirea soldului curent al fiecărui cont și verificarea tuturor încasărilor/plăților prin bancă. SAF-T (Standard Audit File for Tax, implementat treptat din 2022) a introdus obligativitatea raportării electronice detaliate a tranzacțiilor financiare – fișierul SAF-T **include o secțiune “Payments (Plăți/Încasări)”** care reflectă toate plățile și încasările atât în numerar, cât și prin conturi, legate de parteneri[[9]](https://finlight.ro/docs/ghidul-saf-t-finlight/payments/#:~:text=Subsec%C8%9Biunea%20Payments%20,de%20tranzac%C8%9Bii%2C%20data%20tranzac%C8%9Biei%2C). Prin urmare, sistemele informatice contabile trebuie să poată furniza aceste date (inclusiv date despre parteneri, metode de plată, date tranzacție, referințe document) în format standardizat XML.

## Analiza Implementării Curente – Registrul de Casă

**Structură și funcționalități existente:** Aplicația are un modul dedicat Registrului de Casă, implementat conform normelor OMFP 2861/2009 și standardelor locale, după cum reiese din schema de date și codul sursă. În baza de date există tabelul cash\_registers pentru definirea casieriilor și cash\_transactions pentru operațiunile de numerar. Fiecare casierie (cash\_registers) are atribute precum: denumire și cod intern (ex. “Casa Centrală”, cod “CC”), tipul (principală, secundară, valută, casă de avansuri etc.), locația, moneda (ex. RON, EUR), persoana responsabilă (casierul) și soldul curent (currentBalance). De asemenea, sunt prevăzute câmpuri pentru **limite de numerar**: dailyLimit și maxTransactionAmount, care pot fi configurate în funcție de prevederile legale (de exemplu, dailyLimit – plafon zilnic de încasări/plăți per casierie, maxTransactionAmount – plafon per tranzacție cash). O casierie are și un status (active/closed/suspended), permițând închiderea unei case (ex. la epuizarea rolului sau la un punct de lucru închis).

Tabela cash\_transactions înregistrează fiecare operațiune de numerar, stocând informațiile cerute de legislație: - **Număr de document** (documentNumber compus dintr-o serie și un număr secvențial) – aplicația folosește serii separate, de exemplu “CH” pentru chitanțe (încasări) și “DP” pentru dispoziții de plată (plăți). Fiecare tranzacție primește automat un astfel de număr unic în serie, ceea ce asigură numerotarea consecutivă (momentan implementarea folosește un generator placeholder; va necesita finalizare – vezi Plan de acțiune). - **Data și ora tranzacției** (transactionDate) – este setată automat la momentul înregistrării operațiunii. Codul impune ca tranzacțiile de casă să fie înregistrate cu data curentă (utilizând new Date() fără posibilitatea de modificare manuală), respectând cerința ca operațiunile să fie consemnate în ziua în care au loc[[10]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L1270-L1278). Aplicația validează și previne introducerea datelor viitoare sau a unor date diferite de ziua curentă[[11]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L1276-L1284). - **Tipul tranzacției** (transactionType) – definit printr-un *enum* intern (cashTransactionTypeEnum): include valori precum cash\_receipt (chitanță – încasare), cash\_payment (dispoziție de plată – plată numerar), petty\_cash\_advance (avans spre decontare), petty\_cash\_settlement (decontare avans), cash\_count\_adjustment (regularizare la inventar de casă), cash\_transfer (transfer între casierii), bank\_deposit (depunere numerar la bancă), bank\_withdrawal (ridicare numerar de la bancă) etc.[[12]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/shared/schema/cash-register.schema.ts#L32-L40). Această clasificare acoperă toate situațiile uzuale (inclusiv avansuri de trezorerie, depuneri/retrageri legate de bancă și reglaje de casă). - **Scopul tranzacției** (transactionPurpose în schema și enum-ul asociat) – detaliază natura operațiunii: ex. customer\_payment (încasare de la client), supplier\_payment (plată furnizor), salary\_payment (plată salarii), expense\_payment (plată diverse cheltuieli), advance\_to\_employee (acordare avans numerar către angajat), advance\_settlement (rest de plată sau încasare la decont avans), bank\_deposit (depunere la bancă) sau cash\_withdrawal (ridicare de la bancă) etc.[[13]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/shared/schema/cash-register.schema.ts#L48-L56). Acești doi parametri (tip și scop) permit înregistrarea corectă și clasificarea contabilă a operațiunilor. - **Suma** (amount), cu detaliere de TVA dacă este cazul – schema prevede câmpuri vatAmount, vatRate, netAmount pentru situațiile când o tranzacție de casă implică TVA (ex. o vânzare cu bon fiscal). De regulă, încasările și plățile obișnuite sunt în sumă brută (TVA inclus dacă e cazul), iar în registru se trec valorile brute. Aplicația înregistrează totuși și netul și TVA aferent, facilitând rapoarte și verificări (ex. la generarea jurnalelor de TVA sau pentru SAF-T). - **Moneda și cursul** – câmpuri currency și exchangeRate. Registrul de casă poate opera și în valută; OMFP 2634 prevede formulare separate pentru Registrul de casă în valută (cod 14-4-7A), iar aplicația suportă definirea de casierii în valută. Sumele se pot înregistra în moneda respectivă, cu păstrarea cursului de schimb (câmpul exchangeRate) pentru calcularea echivalentului în RON în contabilitate. În implementarea actuală, tranzacțiile în valută sunt posibile, iar dacă exchangeRate != 1 se prevede calculul diferențelor de curs, însă acest aspect este marcat ca TODO în cod (momentan nu se generează automat linii de diferențe de curs la înregistrare, lucru de finalizat). - **Persoana implicată** – câmpurile personName, personId și personIdNumber stochează numele persoanei sau entității care achită/primește numerarul și, dacă e cazul, **CNP-ul sau seria CI** (personIdNumber). Legislația (Legea 70/2015) cere ca pentru plăți către persoane fizice peste un anumit plafon și pentru plata salariilor în numerar să se consemneze CNP-ul persoanei. Aplicația reflectă aceasta: în validarea unei tranzacții de casă se verifică dacă **pentru o plată către furnizor peste 5.000 Lei sau o plată de salariu**, a fost furnizat CNP/CI – altfel returnează eroare. De asemenea, se asociază opțional personId ca referință internă către tabela de parteneri (ex. un ID de client, furnizor sau angajat din alt modul) dacă tranzacția provine direct dintr-o factură sau din evidența angajaților. - **Referință document de bază** – câmpurile invoiceId și invoiceNumber permit legarea tranzacției de casă de o factură încasată sau plătită (din modulele de vânzări/cumpărări). Astfel, se asigură o trasabilitate: în registru apare numărul facturii la explicații, iar în contabilitate se poate face legătura între încasarea/plată și factura aferentă (cont 411/401). Există și contractNumber pentru referințe suplimentare, și un câmp descriptiv liber description în care se trece explicația tranzacției (ex: “Încasare factură 12345 / Client Popescu” sau “Plata avans deplasare Ion Ionescu” etc.). - **Informații despre document fiscal** – câmpuri isFiscalReceipt, fiscalReceiptNumber și fiscalReceiptData pentru cazurile în care registrul de casă primește date de la un aparat de marcat electronic fiscal (ex. vânzări retail în numerar cu bon fiscal). Aceste câmpuri permit evidența bonurilor fiscale integrate în registru (dacă softul este conectat la un POS). În lipsa unui POS, chitanțele tipizate acoperă încasările. - **Soldurile anterioare și ulterioare** – balanceBefore și balanceAfter – stocate pentru fiecare tranzacție. La salvarea unei tranzacții, aplicația calculează soldul casei înainte și după operațiune, actualizând și tabelul casieriei (cash\_registers.currentBalance). De exemplu, pentru o încasare: balanceBefore = sold anterior, balanceAfter = sold anterior + sumă[[14]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L342-L350)[[15]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L374-L382); pentru o plată: balanceAfter = sold anterior – sumă (cu verificarea că soldul nu devine negativ, altfel operațiunea este blocată)[[16]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L410-L418)[[17]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L412-L420). Acest mecanism asigură că în orice moment soldul casieriei este corect și permite listarea registrului de casă cu **sold inițial** la începutul zilei și **sold final** la sfârșit, conform cerințelor. Interfața utilizator arată aceste calcule: există un sumar pentru perioada selectată cu “Sold inițial”, “Total încasări”, “Total plăți” și “Sold final”[[18]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/client/src/modules/accounting/pages/cash-register/index.tsx#L618-L627)[[19]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/client/src/modules/accounting/pages/cash-register/index.tsx#L630-L638). - **Marcaj contabil și audit** – pentru fiecare tranzacție există isPosted (dacă a fost postată în contabilitate) și ledgerEntryId (referința către nota contabilă generată în jurnalul general). De asemenea, isCanceled și motive de anulare dacă e cazul (pentru a invalida tranzacții greșite fără a le șterge fizic, asigurând audit trail). Câmpurile de tip createdAt/By și updatedAt/By există atât pe tranzacții, cât și pe registru, pentru evidența utilizatorilor care au operat datele.

**Conformitate și acoperire a cerințelor legale:** Per ansamblu, implementarea curentă a registrului de casă este **detaliată și conformă** cu cerințele legislației românești la zi:

* **Înregistrare zilnică și documente justificative:** Aplicația obligă utilizatorul să înregistreze fiecare operațiune de numerar printr-un document numeric (chitanță/dispoziție), cu serie și număr unic, data și explicația. Acest lucru corespunde cerinței ca orice încasare în numerar să fie consemnată prin chitanță, iar orice plată prin dispoziție de plată, și apoi înscrise cronologic în registru[[1]](https://infotva.manager.ro/articole/studii-de-caz/registru-de-casa-2023-registrul-se-intocmeste-centralizat-in-cazul-firmelor-cu-mai-multe-puncte-de-lucru-23081.html#:~:text=Nu%20exista%20nicio%20reglementare%20in,Registrul%20de%20casa%20serveste%20ca). Prin generarea numerelor de document și nepermiterea duplicatelor, se respectă regula numerotării consecutive fără întreruperi sau suprapuneri. (Notă: generatorul de numere din cod va trebui finalizat pentru a lua ultimul număr folosit și a incrementa secvențial, inclusiv resetare pe an – vezi plan.)
* **Formatul registrului și elementele obligatorii:** Informațiile stocate acoperă toate coloanele cerute de formularul tipizat de Registru de casă[[2]](https://infotva.manager.ro/articole/infotva/registru-casa-23551.html#:~:text=Continutul%20minimal%20obligatoriu%20de%20informaii,al%20formularului%20este%20urmtorul). În particular: data (transactionDate), numărul documentului de casă (documentNumber cu serie și număr), explicația (description + personName/InvoiceNumber pentru detalii), sumele de **încasări** și **plăți** (aplicația le poate afișa cu plus/minus sau în coloane separate pe raport: în UI, plățile sunt marcate cu valoare negativă roșie, încasările cu verde, conform sumarelor[[20]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/client/src/modules/accounting/pages/cash-register/index.tsx#L640-L648)), soldul de început și de sfârșit de zi (deductibile din balanceBefore al primei tranzacții a zilei și balanceAfter al ultimei). Semnăturile casierului și contabilului pot fi adăugate pe raportul listat (deși nu se stochează digital semnături, documentul tipărit va avea spațiu pentru acestea).
* **Operațiuni speciale și limite legale:** Sistemul prevede explicit tipuri de tranzacții pentru depuneri și ridicări de numerar la bancă, transfer între două casierii, avansuri și deconturi – ceea ce permite reflectarea fidelă a tuturor situațiilor de numerar. Limitările impuse de Legea 70/2015 sunt și ele luate în considerare: câmpurile dailyLimit și maxTransactionAmount din tabelul casieriei pot fi setate (ex. 50.000 Lei limită de casierie la final de zi, 5.000 Lei per tranzacție cash), urmând ca aplicația să valideze la introducerea tranzacțiilor dacă se depășește limita (această verificare trebuie însă implementată efectiv – a se vedea secțiunea de îmbunătățiri). În codul de validare observăm că, de exemplu, pentru plățile mari către furnizori, dacă suma > 5.000 Lei se cere CNP-ul persoanei, ceea ce reflectă parțial disciplina impusă de plafon (deși legal tranzacția ar trebui fragmentată, solicitarea CNP asigură măcar evidența în caz de control). Recomandarea este ca aplicația să refuze sau atenționeze direct utilizatorul la introducerea unei plăți numerar care încalcă plafonul (ex. dacă încearcă să achite integral o factură peste 5.000 Lei prin casă).
* **Solduri și verificări:** Calculul și stocarea soldurilor înainte/după fiecare operațiune reprezintă un mare plus de conformitate – astfel registrul poate fi oricând scos la imprimantă cu soldul inițial al zilei și soldul final, exact cum cere practica contabilă. Aplicația previne soldul negativ (nu permite plăți care depășesc disponibilul – afișează eroare “Insufficient cash balance” dacă balanceAfter ar fi sub 0[[17]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L412-L420)[[21]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L414-L422)). Totodată, închiderea unei casierii (status “closed”) presupune stocarea unui closingBalance și data închiderii, permițând **arhivarea registrului** la momentul respectiv. Presupunem că aplicația nu mai permite adăugarea de tranzacții într-o casierie închisă (acest aspect trebuie confirmat/implementat, altfel se impune blocare în cod: la operațiuni noi să verifice că cashRegister.status este “active”).
* **Integrarea contabilă:** Fiecare tranzacție de casă poate fi marcată ca “postată” (isPosted) și legată de o înregistrare din jurnalul general (ledgerEntryId). În cod există un serviciu de contabilitate (JournalService) care generează în dublă partidă notele contabile aferente tranzacțiilor de casă, folosind conturi prestabilite. De exemplu, pentru cash\_receipt de la client se vor debita conturile de casă (5311) și credita contul client (4111), pentru cash\_payment către furnizor se debitează furnizorul (401) și se creditează casa etc. Implementarea actuală definește un obiect CASH\_ACCOUNTS cu conturile corespunzătoare (5311/5314 pentru casă în lei/valută, 5321 pentru casă de avansuri, conturi de contrapartidă 4111, 401, 425, 421, 4427/4426 pentru TVA, 581 pentru transfer intern, 6581/7588 pentru minus/plus de casă la inventar etc.)[[22]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L88-L97)[[23]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L114-L122). În modulul CashRegisterService, funcția internă care pregătește postarea (care ar putea fi numită *postTransaction* sau similar) ia o tranzacție și construiește linia debit/credit corespunzătoare în funcție de transactionType și transactionPurpose. Fragmentul de cod arată gestionarea tuturor cazurilor:
* Încasare de la client: debit 531 / credit 4111 (descriere “Payment received...”)[[24]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/backup/modules/accounting/services/bank-journal.service.ts#L151-L159)[[25]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/backup/modules/accounting/services/bank-journal.service.ts#L161-L169).
* Plată către furnizor: debit 401 / credit 531 (descriere “Payment to supplier...”)[[26]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/backup/modules/accounting/services/bank-journal.service.ts#L172-L180)[[27]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/backup/modules/accounting/services/bank-journal.service.ts#L181-L190).
* Comision bancar (bank\_fee): debit 627 / credit 512 (ieșire din cont bancă) – în registru de casă nu apare, dar dacă ar fi o cheltuială cu numerar tot prin conturi de cheltuieli ar trece.
* Avansuri salarii sau deconturi: cont 425 sau 542 etc., conform planului de conturi, și contul de casă.
* Regularizări la inventar (cash\_count\_adjustment): se folosește cont 6581 pentru lipsuri sau 7588 pentru excedente, în corespondentă cu casa[[28]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L1119-L1127)[[29]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L1129-L1137).
* Transferuri între case (cash\_transfer): folosesc contul 581 “Viramente interne” – o casă iese (credit 531 / debit 581) și cealaltă intră (debit 531 / credit 581)[[30]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L1169-L1177)[[31]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L1179-L1187). Toate aceste scenarii sunt acoperite logic. Sistemul, după generarea acestor linii, creează efectiv intrarea în jurnalul general (ledgerEntries + ledgerLines) apelând journalService.createLedgerEntry() cu tipul de jurnal “CASH”[[32]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L1197-L1205). Se transmite și referenceNumber (numărul documentului de casă) ca referință în jurnal, precum și descrierea compusă (“Payment received from X ref: Y” etc.). Astfel, tranzacțiile de casă se integrează în contabilitatea dublă, în Registrul Jurnal, permițând apoi includerea lor în balanță și raportări (inclusiv fișierul SAF-T, secțiunea de plăți, care necesită aceste detalii).

**Aspecte de îmbunătățit / Lipsuri identificate:** Deși implementarea este robustă și acoperă majoritatea cerințelor, am identificat câteva puncte ce necesită completare sau verificare pentru a atinge conformitatea 100% și a facilita utilizarea practică:

* **Numerotarea documentelor și gestionarea seriilor:** Funcția de generare a numărului de document (generateReceiptNumber) este în prezent o variantă placeholder care construiește un număr pe baza datei și unui index aleator[[33]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L1414-L1422)[[34]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L1419-L1424). Va trebui implementat un mecanism real, astfel încât chitanțele și dispozițiile să fie numerotate secvențial, fără goluri, pe serii distincte. Soluția poate implica fie o masă separată de “counters” per serie și an, fie interogarea ultimei tranzacții din serie pentru incrementare. De asemenea, codul folosește momentan prefixul “DI” (dispoziție încasare) pentru chitanțe în funcție (deși la inserție s-a folosit “CH”). Este necesară uniformizarea – de preferat utilizarea codurilor consacrate: “CH” pentru chitanță, “DP” pentru dispoziție plată. Un aspect legal: numerotarea ar trebui să reînceapă de la 1 la începutul fiecărui an, și ideal seria să includă anul (ex: CH2025/00001). Aplicarea acestor reguli trebuie clarificată și implementată.
* **Raportarea/Tipărirea Registrului de casă:** Aplicația oferă opțiuni de **Export (PDF, Excel, CSV)** în interfață[[35]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/client/src/modules/accounting/pages/cash-register/index.tsx#L526-L534), însă este necesar să se genereze un format tipărit al registrului de casă care să respecte structura formularului oficial. Acest raport zilnic ar trebui să includă antet (denumirea companiei, denumirea casieriei, data), listarea tranzacțiilor (ideal separate pe două coloane Încasări/Plăți sau cu semn distinct), cu număr curent de ordine pe zi, soldul reportat de ziua precedentă, total încasat, total plătit și soldul final, precum și rubrici pentru semnăturile casierului și contabilului[[36]](https://infotva.manager.ro/articole/infotva/registru-casa-23551.html#:~:text=,contabil)[[7]](https://infotva.manager.ro/articole/infotva/registru-casa-23551.html#:~:text=Soldul%20de%20casa%20al%20zilei,casa%20pentru%20ziua%20in%20curs). În prezent, datele există și pot fi formatate – pasul de generare PDF trebuie implementat pe backend (sau front-end) folosind librăria potrivită. **La final de zi**, utilizatorul ar trebui să poată tipări registrul și să îl arhiveze conform legii (ex. în biblioraft, sau stocat electronic cu semnătură digitală dacă se folosește în acest mod, deși legislația încă cere tipărire pentru control).
* **Închiderea zilnică a casieriei și blocarea modificărilor:** În modul ideal de lucru, după ce toate operațiunile unei zile au fost introduse, casierul “închide ziua” și tipărește registrul. După acest moment, nu ar trebui să mai fie posibilă modificarea sau ștergerea tranzacțiilor din ziua încheiată (orice corecție ulterioară trebuie făcută prin tranzacții distincte de stornare în ziua curentă). În implementarea actuală, nu există un concept explicit de “închidere zilnică” (doar închiderea definitivă a unei casierii). Este recomandată introducerea unui **flag de închidere pe zi** sau a posibilității de a marca tranzacțiile ca validate la sfârșitul zilei, poate printr-un buton “Finalizează ziua”. Acest lucru ar putea seta, de exemplu, un câmp closedAt la nivel de casierie cu data de până la care e închis registrul, și aplicația ar refuza adăugarea/ediția de tranzacții cu data <= closedAt. Alternativ, se poate realiza procedural: utilizatorul tipărește registrul, semnează, iar eventualele modificări ulterioare se descurajează prin politici interne. Totuși, dat fiind că aplicația are câmpul isPosted, o măsură minimă este: **după postarea contabilă a unei tranzacții (isPosted=true)**, să se blocheze editarea sau ștergerea acesteia în UI (posibil deja nepermițând editarea dacă a intrat în note contabile). În cod încă nu văd implementată explicit această restricție, deci trebuie adăugată: la încercarea de a edita/anula o tranzacție cu isPosted, sistemul să dea eroare și să ceară înregistrare de storn (o tranzacție opusă de anulare).
* **Validări suplimentare la introducere:** Ar fi util ca aplicația să folosească câmpurile dailyLimit și maxTransactionAmount configurate. De exemplu, maxTransactionAmount = 5.000 Lei ar trebui să determine ca o tranzacție de încasare/plată peste 5.000 să nu fie permisă (sau să afișeze avertisment “Depășește plafonul legal, fragmentați tranzacția!”). Similar, dailyLimit = 50.000 Lei ar putea fi utilizat la finalul zilei: când se adaugă ultima tranzacție ce duce soldul peste 50k, să atenționeze utilizatorul că se depășește plafonul zilnic în casierie și că excedentul trebuie depus la bancă[[4]](https://www.avocatnet.ro/articol_68880/Ce-plafoane-de-incas%C4%83ri-%C8%99i-pl%C4%83%C8%9Bi-in-numerar-se-aplic%C4%83-in-2025.html#:~:text=Apoi%2C%20Legea%2070%2F2015%20impune%20limite,termen%20de%20dou%C4%83%20zile%20lucr%C4%83toare). Eventual, aplicația poate sugera automat înregistrarea unei operațiuni de depunere la bancă pentru suma ce depășește plafonul.
* **Operațiuni de depunere/ridicare numerar la bancă – legătură cu modulul de bancă:** Aplicația are tipurile bank\_deposit și bank\_withdrawal în registrul de casă, care scad sau cresc numerarul. În prezent, înregistrarea unei depuneri la bancă se face ca o plată cash cu scop bank\_deposit (scade casa). Similar, ridicarea de numerar din bancă se face ca o încasare cash cu scop cash\_withdrawal (crește casa). Totuși, **pentru evidență completă**, aceste operațiuni ar trebui să se reflecte automat și în jurnalul de bancă (conturile 512). Ideal, când se înregistrează o depunere de numerar la bancă în registrul de casă, aplicația să genereze concomitent și o tranzacție în modulul bancă, de tip **încasare în cont (incoming\_payment)** în contul bancar ales, cu aceeași sumă și descriere (eventual marcându-le legate). În cod, funcția recordCashDepositToBank există, dar ea doar apelează intern reducerea din casierie[[37]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L492-L501); nu pare să creeze și intrarea în bank\_transactions. Propunem îmbunătățirea: la depunere/ridicare să se deschidă un dialog care cere selecția contului bancar și apoi sistemul să înregistreze *două* tranzacții: una în casierie, una în jurnalul de bancă, pentru ca cele două module să rămână sincronizate (alternativ, utilizatorul să fie instruit clar că trebuie să facă manual și operațiunea opusă în celălalt modul – însă automatizarea ar elimina erorile/omisiunile).
* **Tranzacții anulate (storno):** Aplicația are câmpuri de anulare (isCanceled, canceledAt, cancellationReason), semn că permite stornarea unei operațiuni de casă greșite. Ar fi de clarificat fluxul: dacă se anulează o tranzacție, soldurile și soldul curent se recalculază automat sau se creează o tranzacție inversă? Posibil se optează pentru marcarea ca anulată și ignorarea ei în rapoarte/sold (caz în care soldul trebuie refăcut). O abordare mai transparentă ar fi ca orice corecție să se facă prin introducerea unei tranzacții de sens contrar (ex: dacă ai înregistrat în plus 100 Lei, faci o plată fictivă de 100 Lei cu explicația “Stornare eroare”). Indiferent de soluție, trebuie testat că soldurile rămân corecte după anulări. De asemenea, pentru audit, registrul de casă tipărit ar trebui să includă eventual tranzacțiile anulate (barate sau cu mențiunea “anulat”) dacă acestea au fost inițial numerotate – ca să nu rămână “goluri” în numerotare. Acest detaliu ține de metodologie (OMFP 2634 menționează că documentele stornate se păstrează cu serie și se notează ca nevalabile).

## Analiza Implementării Curente – Registrul/Jurnalul de Bancă

**Structură și funcționalități existente:** Evidența operațiunilor bancare este gestionată prin tabelele bank\_accounts (conturi bancare ale companiei) și bank\_transactions (mișcări în conturi, similar extrasului de cont). Fiecare cont bancar are ID, companie, nume (ex. “Cont BCR”), număr de cont IBAN (accountNumber), banca și codul SWIFT (bankName, bankCode), moneda și soldul curent (currentBalance). Tabelul de tranzacții bancare stochează: - **Referința tranzacției** (referenceNumber) – corespunde de obicei numărului de document al băncii (nr. de extras, sau identificatorul operațiunii, ex. cod tranzacție SWIFT). Acesta este obligatoriu pentru a putea identifica tranzacția. - **Tipul tranzacției bancare** (transactionType, enum definit ca incoming\_payment – încasare în cont, outgoing\_payment – plată din cont, bank\_fee – comision, bank\_interest – dobândă, transfer\_between\_accounts – transfer între două conturi proprii, loan\_disbursement – primire credit, loan\_repayment – rambursare credit, foreign\_exchange – schimb valutar, other)[[38]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/shared/schema/bank-journal.schema.ts#L18-L26). Aceste categorii acoperă practic orice situație din extrasul bancar. - **Metoda de plată** (paymentMethod, enum cu valori: transfer bancar, direct debit, card, ordin permanent, online banking etc.) – util pentru SAF-T, care cere specificarea modului de plată[[9]](https://finlight.ro/docs/ghidul-saf-t-finlight/payments/#:~:text=Subsec%C8%9Biunea%20Payments%20,de%20tranzac%C8%9Bii%2C%20data%20tranzac%C8%9Biei%2C). De exemplu, dacă o plată este cu OP clasic va fi bank\_transfer, dacă e cu cardul firmei card\_payment, etc. - **Datele tranzacției:** data tranzacției (transactionDate) și data valorii (valueDate). De regulă, transactionDate e data operației în sistem, iar valueDate data decontării efective (util la dobânzi – de ex., dobânda poate avea data valorii diferită de data calculului). - **Suma și moneda** (amount, currency) cu curs (exchangeRate dacă tranzacția e în valută). Schema suportă conturi în valută: de exemplu cont 5124 (valută), caz în care se stochează și echivalentul. Dacă se înregistrează operațiuni de schimb valutar (foreign\_exchange), probabil se folosesc două tranzacții legate – una iese de pe un cont, alta intră pe alt cont, iar diferențele de curs se reflectă separat. - **Descrierea tranzacției** (description) – text liber, de exemplu “Încasare client X factura 1001” sau “Plata furnizor Y factura 123/2025”, “Comision bancar luna iunie” etc. Există și câmpuri pentru numele plătitorului și beneficiarului (payerName, payeeName), foarte utile pentru identificarea încasărilor/plăților de la/către terți, mai ales când IBAN-ul aparține unei persoane fizice sau când se procesează plăți salarii (lista salariaților). - **Legături către documente interne:** invoiceId și invoiceNumber permit, similar ca la casa, legarea unei tranzacții de bancă de o factură emisă/recepționată. Astfel, dacă se încasează o factură de la client prin bancă, se poate completa invoiceNumber pentru reconcilieri (chiar dacă nu e obligatoriu, ajută la raportarea în jurnalul de vânzări încasări sau la închiderea conturilor 411/401). - **Sold înainte și după** (balanceBefore, balanceAfter) – la fel ca la casă, se calculează soldul contului înainte și după acea tranzacție. Când se introduce o linie din extras, aplicația ar trebui să ia soldul curent al contului ca balanceBefore, să adauge sau scadă suma și să obțină balanceAfter, actualizând totodată în bank\_accounts.currentBalance. Fragmentul de cod sugerează această logică: la fiecare creare de tranzacție se face update la soldul contului bancar (similar cum se face la casă). Asigurând aceste date, se poate reconstitui oricând extrasul cu soldurile zilnice. - **Marcaj contabil:** Și tranzacțiile bancare au isPosted și ledgerEntryId, indicând integrarea cu modulul contabil. Tipul de notă folosit va fi probabil “BANK” (în JournalService LedgerEntryType există și BANK). Conturile corespondente sunt definite în BANK\_ACCOUNTS object (5121, 5124 pentru bănci; 627 pentru comisioane, 766/666 pentru dobânzi active/pasive, 519/162 pentru credite etc.)[[39]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/bank-journal.service.ts#L75-L83)[[40]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/bank-journal.service.ts#L88-L96).

**Conformitate și utilitate practică:** Având aceste elemente, modulul de bancă este aliniat cu cerințele contabilității financiare. Toate tranzacțiile care apar pe extrasele bancare pot fi introduse și identificate clar: - **Corectitudine contabilă:** Dacă la registru de casă accentul este pe formular și procedură, la jurnalul de bancă accentul e pe **completețea și corectitudinea datelor** pentru conturile contabile. Implementarea actuală asigură că fiecare mișcare are cont bancar specific (astfel se poate concilia contul 512x), are mențiuni despre contrapartidă (payer/payee sau factură) ceea ce ajută la închiderea conturilor de terți, și are încadrări pe tipuri pentru a fi contabilizată corect. De exemplu, o tranzacție de tip BANK\_FEE ar trebui să genereze nota: Debit 627 “Comisioane” / Credit 5121, iar enum-ul și conturile predefinite confirmă această logică. Similar, dobânzile primite: Debit 5121 / Credit 766 “Venituri din dobânzi” etc. - **Reconciliere extrase:** Stocând referenceNumber și valueDate, aplicația permite ulterior importul automat al extraselor (dacă se implementează) și potrivirea tranzacțiilor. Chiar dacă deocamdată utilizatorul trebuie probabil să introducă manual fiecare linie de extras, structura este pregătită pentru **integrare automată** (ex. import CSV/MT940 de la bancă). Recomandăm dezvoltarea unei funcții de import care să populeze bank\_transactions direct din fișierele băncii, minimizând erorile manuale. Ulterior, utilizatorul poate completa câmpurile invoiceNumber sau altele pentru reconcilieri contabile. - **Legătura cu plățile clienți/furnizori:** Odată ce o tranzacție de încasare bancară este introdusă cu referință la factura client, aplicația ar putea automat să marcheze acea factură ca plătită (în modulul de vânzări) sau să facă legătura inversă. Acest aspect ține de integrarea modulelor (vezi secțiunea următoare), dar evidențiem că modelul de date permite astfel de automatizări. - **Raportare și SAF-T:** Datele din bank\_transactions vor alimenta, alături de notele contabile, raportările oficiale. De exemplu, în fișierul SAF-T, secțiunea “Payments” va conține pentru fiecare plată în numerar sau bancă câmpuri precum: ID partener (se poate deduce din invoiceId/payerId), nume partener, date document sursă (factură) și metoda de plată (pentru plățile prin bancă va fi “BankTransfer” sau alt cod aferent, pentru plăți cash – “Cash”). Aplicația are deja enumerările pentru metoda de plată conforme cu nomenclatorul, deci exportul SAF-T va trebui doar să le mapeze. Un alt raport intern important este jurnalul de bancă (sau registrul de bancă) – aici ar fi utilă o listare, pe fiecare cont, a tuturor tranzacțiilor într-o perioadă, cu sold inițial și final. Acest raport ar semăna cu un extras de cont și ar servi atât la verificări contabile (ex. dacă soldul final din aplicație = soldul din extrasul băncii), cât și la eventuale controale (de exemplu, auditorul poate cere jurnalul tuturor operațiunilor prin bancă). Implementarea actuală nu menționează explicit un raport tipărit de jurnal bancă, dar opțiunea de **Export CSV/Excel** din UI pentru tranzacții bancare probabil există (similar modului de casă). Oricum, generarea unui PDF sumar pe lună sau pe perioadă ar fi de adăugat.

**Lipsuri și aspecte de rezolvat pe partea de bancă:**

* **Actualizarea soldului contului bancar:** Trebuie verificat în cod dacă la inserarea unei tranzacții bancare se actualizează automat bank\_accounts.currentBalance. Având în vedere cum e scris la casă (unde imediat după insert se dă update la soldul casei[[41]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L382-L390)[[42]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L448-L456)), este de așteptat că modulul de bancă are ceva similar. Dacă totuși nu e implementat, trebuie adăugat: la createDeposit (încasare) să crească soldul contului, la createPayment (plată) să scadă soldul, și la fel pentru comisioane sau dobânzi. Fără acest update, balanceBefore/After nu s-ar popula corect. Faptul că schema cere ca balanceBefore și balanceAfter să fie not null sugerează că logica există. (În backup sau cod, funcțiile createDeposit și createPayment ar trebui să facă: citire sold actual, calcul sold nou, inserare tranzacție cu balanceBefore/After, apoi update cont.)
* **Secvențe și referințe:** Nu există conceptul de serie și număr de document pentru tranzacțiile bancare, deoarece acestea sunt identificate de regulă de numărul de referință al băncii. Asta e corect. Totuși, dacă firma emite OP-uri din sistem, ar putea dori numerotarea ordinelor de plată. În absența unui astfel de modul, nu e o problemă – referenceNumber va fi completat cu ID-ul extrasului sau al tranzacției din extras. Asigurați-vă doar că utilizatorii folosesc consistent acest camp (ex. introduc “Ext#123 pagina1 poz.5” sau codul SWIFT al tranzacției) pentru a putea regăsi ulterior tranzacția.
* **Conectarea la general ledger:** Similar cu registrul de casă, tranzacțiile bancare necesită **postare contabilă**. Schema indică ledgerEntryId, iar JournalService suportă LedgerEntryType.BANK. În codul BankJournalService există o funcție createBankTransactionEntry care construiește liniile contabile în funcție de transactionType (ex. pentru OUTGOING\_PAYMENT: Debit furnizori 401 / Credit bancă 5121)[[26]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/backup/modules/accounting/services/bank-journal.service.ts#L172-L180)[[27]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/backup/modules/accounting/services/bank-journal.service.ts#L181-L190), apoi ar trebui să apeleze journalService.createLedgerEntry cu type=BANK și să întoarcă entry-ul. Rămâne de implementat legătura: după ce se inserează o tranzacție bancară în bank\_transactions, sistemul trebuie fie să lanseze imediat generarea notei contabile și să marcheze tranzacția isPosted=true, ledgerEntryId=<id> (preferabil), fie să ofere un mecanism de postare ulterioară (de exemplu, un buton “Postează tranzacțiile selectate” în UI, pentru a permite verificarea înainte de contabilizare). E de preferat postarea imediată pentru a evita discrepanțe. Dacă momentan nu se întâmplă automat, este un **gap de implementare** ce trebuie acoperit: altfel, tranzacțiile ar rămâne neînchise în contabilitate. Planul de acțiune include deci și finalizarea rutinei de postare automată la salvare.
* **Import și reconciliere semi-automată:** Un aspect funcțional – nu neapărat legal – este facilitarea concilierii bancare. Ar fi util ca aplicația să permită marcarea tranzacțiilor ca **conciliate** sau verificarea diferențelor între extrasul bancar importat și notele contabile. De exemplu, dacă se implementează un import de extras, sistemul ar putea încerca potrivirea automată a plăților cu facturile (după sumă și eventual nr. factură din descriere) și completarea câmpurilor invoiceId acolo unde găsește corespondent, facilitând contabilului munca. Aceasta e o sugestie pe termen lung pentru eficiență.
* **Tranzacții între conturi proprii:** Categoria transfer\_between\_accounts indică transferuri între două conturi ale aceleiași firme. Implementarea ar trebui să creeze două intrări: una de ieșire din contul sursă și una de intrare în contul destinație, eventual legate printr-o referință comună. Momentan, dacă utilizatorul folosește transfer\_between\_accounts, nu e clar dacă aplicația generează automat ambele părți. Probabil nu (similar ca la casă unde transferul între case apelează două funcții separate). Se poate adopta soluția folosită la casă: o funcție care, când se selectează cont sursă și cont destinație, apelează intern createPayment pe contul sursă și createDeposit pe contul destinație cu aceleași detalii. Acest lucru ar garanta că transferul intern nu lasă “găuri” (altfel, riscăm ca un utilizator să introducă ieșirea dintr-un cont și să uite să o introducă pe cealaltă parte).
* **Dobânzi și comisioane:** Asigurarea că aceste tranzacții sunt ușor de operat. Pentru comisioane, modulul prevede tipul BANK\_FEE și contul 627 implicit; pentru dobânzi primite – BANK\_INTEREST și cont 766/5187. Aplicația ar trebui să permită adăugarea eventual a mai multor linii de comision (ex. dacă banca percepe un comision cumulat pentru mai multe operațiuni, contabilul poate dori să îl detalieze, dar se poate și sumar). Oricum, legal, aceste elemente trebuie înregistrate lunar cel târziu, deci existența categoriilor e conformă.

## Integrarea cu Alte Module (Contabilitate, Vânzări/Cumpărări, SAF-T)

Implementarea celor două registre (casă și bancă) nu funcționează izolat, ci se leagă de restul sistemului ERP și de cerințele de raportare fiscală, asigurând un flux coerent al datelor financiare:

* **Legătura cu contabilitatea generală:** După cum s-a menționat, ambele module generează note contabile prin JournalService. În planul de conturi românesc, contul de casă 5311 și conturile de bancă 5121/5124 sunt conturi de **activ** care își găsesc reflectarea în balanță (soldurile de la final de lună ale acestor conturi trebuie să corespundă cu monetarul din casierie, respectiv cu soldurile din extrasele bancare). Aplicația, cu condiția ca toate tranzacțiile să fie postate, va alimenta tabela account\_balances pentru conturile 531/512, permițând obținerea rapidă a unui **balanțier** sau a balansului de verificare. Integrarea pare să fie gândită corect – trebuie doar confirmat că fluxul de postare se execută. Un aspect de verificat: **numerotarea notelor contabile**. În Registrul Jurnal general, toate operațiunile (inclusiv cele provenite din registrul de casă și jurnalul bancă) trebuie ordonate cronologic și numerotate (de obicei, ERP-ul generează un număr de jurnal pentru fiecare notă, ex. JV#100, JV#101 etc.). Ar trebui ca ledgerEntries.referenceNumber sau ledgerEntries.type să codifice proveniența (posibil se folosește tipul de jurnal: CASH, BANK, SALES, PURCHASE etc. – în cod există journalTypes dacă se dorește evidență separată). Asigurându-se că aceste intrări ajung în registrul jurnal general tipărit, se respectă OMFP 2634/2015 privind registrul jurnal. Totodată, **cartea mare** (contabilitatea pe conturi) este automat asigurată prin tabela ledger\_lines și account\_balances.
* **Integrarea cu modulul de Vânzări și Cumpărări (Facturi):** O facilitate foarte utilă este conectarea plăților la facturi. Deja există câmpuri invoiceId și invoiceNumber la ambele tipuri de tranzacții. Recomandăm ca atunci când utilizatorul încasează o factură din modulul de vânzări, să poată crea direct o tranzacție de casă sau bancă precompletată cu suma și referința facturii. Invers, dacă se adaugă manual un cash\_transaction cu invoiceNumber, aplicația ar trebui să marcheze factura respectivă ca “încasată” (total sau parțial, după sumă). Similar pentru plăți furnizori: o dispoziție de plată sau un outgoing\_payment legat de o factură de cumpărare poate **închide** acea factură în evidența furnizorilor. Această integrare asigură consistența între jurnalele de vânzări/cumpărări și registrul de casă/bancă. Legal, jurnalele de TVA (de vânzări și cumpărări) conțin coloane cu data încasării/plății pentru facturile cu TVA la încasare – deci iată un exemplu unde modulul de casă/bancă trebuie să transmită date: aplicația a implementat TVA la încasare (cont 4428) în jurnalul de vânzări, și probabil actualizează TVA exigibil când marcați factura ca încasată. Asigurând legătura directă prin invoiceId, se poate face transferul automat al TVA la momentul plății. Implementarea evidențiată în documentația internă (commit complet Jurnal Vânzări) arată că sistemul a fost pregătit pentru TVA la încasare[[43]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/SALES_JOURNAL_IMPLEMENTATION_COMPLETE.md#L114-L122)[[44]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/SALES_JOURNAL_IMPLEMENTATION_COMPLETE.md#L116-L124); partea de casă/bancă va declanșa acel mecanism.
* **Modul Plăți/Aprobare plăți:** Dacă ERP-ul include un modul de Trezorerie sau Plăți (pentru generarea ordinelor de plată, gestionarea avansurilor, planificarea plăților scadente), registrul de casă și jurnalul de bancă ar trebui conectate. De exemplu, dacă se aprobă plata unui furnizor, sistemul ar trebui să permită alegerea modalității: „din casierie” sau „prin bancă (OP)”. Alegând „din casierie”, ar genera o dispoziție de plată în registrul de casă (și ar marca factura plătită). Alegând „prin bancă”, ar putea fie crea o tranzacție bancară (dacă plata s-a făcut și se cunoaște extrasul), fie cel puțin un placeholder până la confirmarea din extras. Momentan, nu avem detalii despre un astfel de modul, dar dat fiind că există PaymentMethod enum și tipuri de tranzacții, sistemul e conceput să capteze informațiile dacă utilizatorul le introduce. Planul pe viitor poate fi integrarea mai strânsă: la marcarea unei facturi ca plătite prin OP, să apară direct în jurnalul de bancă ca „în curs de decontare” până la primirea confirmării (aceasta e o funcție mai complexă, dar utilă firmelor mari).
* **Rapoarte SAF-T și declarații fiscale:** O dată ce tranzacțiile de casă și bancă sunt înregistrate și postate, aplicația trebuie să fie capabilă să genereze **Declarația 406 SAF-T**. Pentru secțiunea de **Plăți (Payments)**, se va prelua fiecare tranzacție de încasare/plată aferentă unui partener: de exemplu, o încasare numerar de la client va apărea ca Payment cu <PaymentMethod>Cash</PaymentMethod>, <Customer> cu datele clientului, <Invoice> referința facturii (dacă există), <Amount> suma etc. O plată bancară la furnizor va fi un Payment cu <PaymentMethod>Transfer, <Supplier> etc. Aplicația are toate aceste date, doar trebuie mapate la XML. De asemenea, SAF-T cere și raportarea soldurilor finale din casierii și conturi (care se regăsesc în secțiunea de solduri contabile din fișier, derivate din balanța conturilor 531/512). **Normele ANAF** (precum Ghidul contribuabilului SAF-T) specifică și necesitatea raportării numerarului în casierie la final de an, dar asta se realizează oricum prin inventariere și înscrierea rezultatului în situații financiare. Pe linie de declarații curente: nu există o declarație periodică care să preia direct registrul de casă (poate doar dacă ANAF cere la control). Însă e demn de menționat **Declaratia 394**: aceasta cere raportarea operațiunilor cu parteneri (clienți/furnizori) inclusiv tipul încasării (numerar sau nu) la livrări către populație peste un anumit plafon. Dacă aplicația categorizează corect plățile (are flag să știe dacă o factură client a fost achitată cash), atunci se pot extrage acele informații pentru 394 (de exemplu, total facturi achitate numerar către persoane fizice > 10k, care se declară distinct). Astfel, evidența implementată ajută și la conformitatea declarativă.
* **Securitatea și auditul intern:** Merită menționat că atât registrul de casă, cât și tranzacțiile bancare, fiind zone sensibile (gestiune de bani), trebuie supuse controlului intern: drepturi de acces (nu oricine poate înregistra sau modifica; de obicei doar casierul și supervizorul), jurnalizare a modificărilor (aplicația are createdBy/updatedBy, ceea ce e foarte bine – permite să se știe cine a operat fiecare tranzacție). De asemenea, periodic (lunar, anual), se recomandă listarea acestor registre și arhivarea lor. Implementarea actuală, cu posibilitatea de export și cu stocarea datelor de audit, oferă suport tehnic pentru aceste proceduri; rămâne ca fluxurile de lucru să fie definite în companie și eventual reflectate în aplicație (ex: un status al tranzacției – “înregistrat”, “verificat” – dacă se implementează un workflow de aprobare).

În concluzie, modulele de Registru de Casă și Jurnal de Bancă din aplicația GeniusERP sunt construite în acord cu standardele contabile românești și acoperă majoritatea cerințelor legale. Urmează prezentarea unui **plan de acțiune detaliat** pentru a aduce implementarea la un nivel complet de conformitate și eficiență, acoperind punctele nevralgice identificate.

## Plan de Acțiune – Consolidare și Finalizare Implementare Registru de Casă și Bancă

Pentru a atinge o implementare completă, aliniată 100% la legislația în vigoare și la bunele practici, propunem următorii pași detaliați:

1. **Implementarea numerotării secvențiale a documentelor de casă:**
2. Creați un mecanism de stocare a ultimului număr folosit pe fiecare serie și an (de ex. un tabel document\_counters cu chei: companie, casierie, serie, an, lastNumber). La inserția unei noi tranzacții, înlocuiți logica provizorie din generateReceiptNumber cu: citire counter curent, incrementare și formatare număr (zerofill pe 5-6 cifre). Asigurați separarea seriilor **CH** (chitanțe) și **DP** (dispoziții plată), conform uzanțelor. Includeți anul în formatul numărului (ex: CH/2025/000123). Actualizați seria “DI” din cod la “CH” pentru consistență.
3. **Reset anual:** La prima tranzacție din noul an, resetați contorul (puteți detecta schimbarea de an în query sau printr-un job la 31 dec).
4. **Evitarea golurilor:** dacă se anulează o tranzacție deja numerotată, nu renumerotați restul – lăsați numărul neutilizat marcat ca “anulat” (tranzacția există cu flag de anulare). Documentați în manualul de utilizare că seriile de registru de casă sunt gestionate automat și nu pot fi modificate manual, pentru a preveni abuzuri.
5. **Validări suplimentare privind plafoanele de numerar (Legea 70/2015):**
6. Implementați în CashRegisterService.recordCashPayment și recordCashReceipt verificări utilizând maxTransactionAmount. Dacă data.amount depășește acest plafon (ex. > 5.000 Lei) și transactionPurpose este “supplier\_payment” sau “customer\_payment”, **afișați o eroare** de tip “Tranzacție peste plafonul legal – suma maximă cash permisă este 5000 Lei. Fragmentați plata sau folosiți bancă.”. Acest lucru va impiedica înregistrarea accidentală a unor sume neconforme. (Pentru situații precum restituire avans > 5.000 către administrator – legea permite 10.000 Lei către PF – puteți ajusta condiția în funcție de tipul de partener dacă se cunoaște).
7. La finalizarea fiecărei tranzacții, verificați dacă prin acea operațiune soldul casei la sfârșitul zilei ar depăși dailyLimit (dacă e setat, ex. 50.000). Puteți calcula total încasări cash ale zilei curente din DB + sold inițial. Dacă depășește, lansați un **warning** către utilizator (nu neapărat blocant): “Atenție: Soldul casieriei depășește plafonul de 50.000 Lei. Conform Legii 70/2015, numerarul excedentar trebuie depus la bancă în 2 zile[[4]](https://www.avocatnet.ro/articol_68880/Ce-plafoane-de-incas%C4%83ri-%C8%99i-pl%C4%83%C8%9Bi-in-numerar-se-aplic%C4%83-in-2025.html#:~:text=Apoi%2C%20Legea%2070%2F2015%20impune%20limite,termen%20de%20dou%C4%83%20zile%20lucr%C4%83toare).”. Oferiți opțiunea de a deschide un dialog de **Depunere la bancă** precompletat cu suma ce depășește plafonul. Astfel, aplicația nu doar semnalează problema, ci și ajută la rezolvarea ei.
8. Mențineți deja implementată validația de CNP pentru plăți mari și extindeți-o dacă e cazul: includeți și situația încasărilor de la persoane fizice peste 10.000 Lei (dacă apar; legea interzice fragmentarea tranzacțiilor peste 10.000 între persoane fizice[[45]](https://www.avocatnet.ro/articol_68880/Ce-plafoane-de-incas%C4%83ri-%C8%99i-pl%C4%83%C8%9Bi-in-numerar-se-aplic%C4%83-in-2025.html#:~:text=servicii%2C%20precum%20%C8%99i%20cele%20reprezent%C3%A2nd,cum%20mai%20apare%20%C3%AEn%20lege), dar dacă totuși se încasează, cereți CNP). Practic, orice personName de tip persoană fizică și sumă > 10.000 -> obligatoriu CNP. Această măsură ține evidența necesară pentru eventuale raportări fiscale.
9. **Funcționalitate de “Închidere zilnică” a registrului de casă:**
10. Adăugați în UI un buton **“Închide ziua”** pe ecranul Registrului de casă (activ doar dacă există tranzacții în ziua curentă neînchise). La apăsare, solicitați confirmare și apoi:
    * Calculați soldul final al zilei respective (deja îl aveți ca balanceAfter al ultimei tranzacții a zilei).
    * Marcați undeva în DB că ziua respectivă e închisă. Variante: creați un tabel nou cash\_register\_closures cu (cashRegisterId, date, closingBalance, closedBy) sau, mai simplu, adăugați un lastClosedDate la cash\_registers. Puteți seta lastClosedDate = <ziua> și eventual closingBalance = <sold> (deși closingBalance există pentru închiderea completă a casieriei, poate fi refolosit zilnic dacă nu creați tabel dedicat).
    * Generați raportul **Registru de casă pentru ziua respectivă** în PDF și salvați-l sau oferiți-l pentru descărcare. Marcarea zilei ca închisă ar putea fi condiționată de generarea cu succes a PDF (astfel te asiguri că utilizatorul a scos raportul).
11. Odată ziua închisă, aplicația ar trebui să **blocheze orice modificare ulterioară** a tranzacțiilor din acea zi. Implementați această regulă în layerul de servicii: la updateCashTransaction sau ștergere (dacă există), verificați transactionDate vs cashRegister.lastClosedDate. Dacă data tranzacției <= lastClosedDate, returnați eroare “Registrul de casă pentru această zi este închis, nu mai puteți modifica tranzacțiile. Operațiunile corective se vor face în ziua curentă.”.
12. Dacă un utilizator insistă să corecteze o eroare din trecut, instruiți-l (prin mesaj sau procedură) să facă o tranzacție de ajustare astăzi (ex: dacă acum 2 zile a înregistrat greșit +100 lei, azi face o plată “Stornare eroare 2 zile în urmă” de 100 lei). Acest mod păstrează intact registrul închis și evidențiază corecția separat, fiind conform cu cerințele de audit (nu rescriem istoria financiară, ci o corectăm prin înregistrări ulterioare).
13. **Postarea automată a tranzacțiilor în contabilitate și blocarea la editare după postare:**
14. Integrați apelul către funcțiile de generare a notelor contabile direct în fluxul de înregistrare a tranzacțiilor. Concret, în recordCashReceipt și recordCashPayment, imediat după inserarea în cash\_transactions și actualizarea soldului, **apelați rutina de creare a notei contabile**. În cod există deja părți din această rutină (createCashTransactionEntry sau similar, care construiește ledgerLines și apelează journalService.createLedgerEntry) – folosiți-le. Asigurați-vă că transmiteți toate informațiile necesare (inclusiv receiptNumber/documentNumber, personName etc. pentru descrieri).
15. La întoarcerea cu succes a notei contabile (veți primi un ID de ledgerEntry), faceți update pe tranzacție: isPosted = true, ledgerEntryId = <ID>, postedAt = now(). În caz de eroare la generarea notei (ex. dezechilibru – teoretic nu ar trebui, dar prindeți excepțiile), puteți fie să marcați tranzacția ca nepostată și să lăsați pentru reparare, fie să faceți rollback total al operațiunii. Mai simplu: puteți încerca postarea **în cadrul aceleiași tranzacții de bază de date** (dacă utilizați tranzacții SQL) astfel încât fie ambele (inserția și nota) reușesc, fie niciuna.
16. Pentru tranzacțiile bancare, faceți același lucru în createDeposit, createPayment etc.: după ce calculați soldul și inserați în bank\_transactions, apelați createBankTransactionEntry (din BankJournalService) pentru a genera nota contabilă, apoi marcați isPosted=true.
17. După implementarea acestor postări imediate, **restricționați editarea/ștergerea** tranzacțiilor postate: în metodele de update/delete verificați flag-ul. Dacă isPosted e true, blocați operația cu mesaj: “Tranzacție deja contabilizată – nu poate fi modificată. Dacă e necesară o corecție, efectuați o tranzacție de stornare.”. Această măsură previne discrepanțe între registru și contabilitate și este alinată principiilor de integritate (similar cum o factură emisă nu mai poate fi editată după ce a fost declarată în jurnalul de TVA).
18. **Finalizarea modulului de import și conciliere bancă (opțional, dar recomandat):**
19. Elaborați un importator de extrase bancare: permiteți utilizatorului să încarce un fișier MT940, CAMT.053 sau măcar CSV exportat din e-banking. Parseați fișierul și completați o listă de tranzacții propuse. Pentru fiecare, încercați maparea: dacă la descriere detectați “FACT 123” și există factura 123 neîncasată, precompletați invoiceId și payerName. Oferiți UI de validare unde utilizatorul poate edita aceste propuneri (să aleagă la ce client să înregistreze o plată cu descriere generică, etc.). La confirmare, creați intrările în bank\_transactions și postați-le.
20. Marcați tranzacțiile importate ca **nerevizuite** inițial (isPosted ar putea fi false până confirmă user, sau un flag separat “reconciled”). După ce utilizatorul le verifică (poate vedea diferențele de câțiva bani pe curs, etc.), să poată apăsa “Conciliere” și atunci să se posteze contabil cu eventuale ajustări (ex: dacă există diferențe de curs, adăugați automat linii pe 665/765).
21. Această funcționalitate ține de eficiență, nu e impusă legal, dar contribuie la acuratețe și la alinierea cu cerința OMFP 2634/2015 de a verifica extrasele cu contabilitatea.
22. **Raportare și documente tipărite:**
23. **Registru de Casă tipărit:** După cum s-a descris la pasul 3, implementați generarea PDF la închiderea zilei. Asigurați formatul cu coloanele clasice: Nr. crt, Document (Chitanța/Dispoziție nr. X), Explicație (conține denumire partener și motivul plății/încasării), Încasări (suma în coloană dedicată dacă e pozitivă), Plăți (suma la plăți dacă e negativă), eventual TVA separat dacă doriți (nu e neapărat pe registru de casă tipărit, dar intern având net/TVA se pot controla), Sold reportat (pe prima linie a zilei – corespunde soldului final zilei anterioare) și Sold de închidere. La final: total încasări, total plăți și sold final, plus linii de semnătură “Casier:” și “Compartiment financiar-contabil:”.
    * Includeți în antet și numele casieriei (dacă firma are mai multe case). Este posibil ca legal să trebuiască ținut un registru separat pe fiecare casă – aplicația deja separă tranzacțiile pe cashRegisterId, deci tipăriți individual. Dacă userul selectează “Toate” casieriile pe UI, generarea ar trebui ori să fie gri (nu permiți, ceri să aleagă o casă), ori să scoți rapoarte distincte pe fiecare. Conform normelor, **nu se cumulează mai multe casierii într-un singur registru**[[6]](https://infotva.manager.ro/articole/studii-de-caz/registru-de-casa-2023-registrul-se-intocmeste-centralizat-in-cazul-firmelor-cu-mai-multe-puncte-de-lucru-23081.html#:~:text=Registrul%20de%20casa%20nu%20se,de%20la%20punctele%20de%20lucru).
    * Arhivare: indicați utilizatorului să tipărească și să păstreze aceste registre. Alternativ, dacă se acceptă arhivare electronică, asigurați că PDF-urile nu pot fi alterate ulterior (eventual semnătură digitală sau măcar checksum).
24. **Jurnal de Bancă / Extras de cont tipărit:** Oferiți un raport per cont bancar, per interval (de regulă lunar): similar extrasului, cu coloane Dată, Descriere, Încasări, Plăți, Sold (sau sold coloană separată pe fiecare rând cum fac băncile). Totalizați doar eventual comisioane/dobânzi dacă e cazul. Semnături nu sunt necesare aici conform legii, dar un auditor intern ar putea dori viză.
    * În plus, se poate genera **situația soldurilor pe conturi de bancă** la final de lună: un tabel ce listează fiecare cont cu soldul inițial, rulaj debitor, rulaj creditor, sold final (aceste date sunt oricum în balanță, dar uneori se atașează la dosarul balanței și extrasele de cont cu aceste cifre). Având account\_balances, puteți realiza ușor acest report.
25. **Jurnalul operațiunilor de casierie în valută:** Dacă se folosesc casierii în valută, generați registrul separat pentru acestea, conform formularului 14-4-7A (care e similar, doar că precizați valuta și eventual cursul folosit pentru echivalent – deși nu se cere neapărat în registru fizic, dar contabilitatea trebuie oricum să țină cont de echivalent în lei pentru înregistrarea în conturi 5314). Asigurați că în rapoarte, sumele apar atât în valută cât și totalul în lei, dacă e cerut de politicile firmei. În general, registrul de casă în valută se ține pe moneda respectivă, iar în conturile contabile se înregistrează în lei la cursul zilei, diferențele de curs mergând la venituri/cheltuieli – aplicația va trebui să aloce acele diferențe (de ex. la închiderea avansurilor sau la restituirea lor). Puneți acest subiect în atenție la implementarea completă a diferențelor de curs.
26. **Tratamentul diferențelor de curs valutar la operațiuni cash și bancă:**
27. Completați logica lăsată în TODO în cod: dacă currency != 'RON' la o tranzacție de casă sau bancă, calculați diferența dintre exchangeRate folosit la acea operațiune și cursul BNR al zilei (sau cursul de la înregistrarea facturii, în cazul plății unei facturi valută). Dacă diferența e semnificativă, generați automat linii contabile pe 665/765. De exemplu, dacă s-a încasat 100 EUR numerar pentru o factură la curs 4.9, dar cursul în ziua plății e 5.0, atunci se va fi înregistrat în contabilitate: Debit 5314 = 490 RON, Credit 411 = 490 RON la emitere, dar la încasare ar trebui să se facă Credit 411 = 490, Debit 5314 = 500 și diferența 10 RON Debit 411/Credit 765? Aici e de discutat, dar mecanismul general: la plată/incasare se realizează diferența de curs între valoarea în lei la curs istoric vs curs curent. Aplicația poate păstra cursul facturii în invoice (dacă e IFRS, altfel metoda PCU IFRS vs Ordinul 1802 diferă puțin la evidența diferențelor). Simplificat: generați diferența ca *cheltuială/venit din diferențe de curs* astfel încât contul de terț să se închidă exact. Acest subpas este destul de complex și poate necesita consultarea unui contabil pentru corectitudine, dar este necesar pentru conformitatea completă la raportări financiare (mai ales la bancă, unde soldurile valutare pot genera diferențe la fiecare extragere de curs).
28. Dacă implementarea acestor diferențe depășește scope-ul imediat, asigurați măcar că **utilizatorul poate interveni**: adică dacă soldul în valută nu se potrivește pe lei, să poată adăuga manual o tranzacție de tip foreign\_exchange în modulul bancă, care să ajusteze conturile (ex: reevaluare lunară). Multe ERP-uri lasă reevaluarea cursului pe final de lună manual (sau modul separat). Deci ca minim, modulul să nu împiedice asta.
29. **Integrare SAF-T și alte rapoarte fiscale:**
30. După ce fluxurile de postare și colectare date sunt stabile, dezvoltați generatorul de fișier SAF-T D406. Pentru secțiunile **GeneralLedgerEntries** (jurnal) deja aveți datele postate. Pentru **Payments**, parcurgeți tranzacțiile cash și bancă postate din perioada selectată și construiți nodurile XML necesare. Exemplu de mapping:
    * <Payment>  
      – <PaymentID> = poate fi cash\_transaction.documentNumber sau bank\_transaction.referenceNumber  
      – <PaymentDate> = data (transactionDate)  
      – <PaymentAmount> = suma (pozitivă)  
      – <Currency> = valuta  
      – <PaymentMethod> = metodă (folosiți direct valorile din enum, mapând la codurile SAF-T: de ex. “cash” pentru Cash, “bankTransfer” pentru transfer etc.)  
      – <SourceDocuments> = referințe la facturi dacă există (acest nod e opțional dacă e asociat clar unei facturi, altfel lăsați generic)  
      – <Description> = ce aveți în description/personName.  
      – <CustomerID> sau <SupplierID> = dacă personId referă la un client/furnizor intern, includeți-l (SAF-T cere ID-ul din lista de parteneri raportată în secțiunea dedicată). Dacă partenerul nu e în listă (ex. persoană fizică neafiliată), se poate totuși trece numele și CNP într-un câmp de descriere – consultăm normele SAF-T pentru asta (de obicei doar clienții și furnizorii cu cod fiscal se declară, dar să fim consecvenți).
31. Testați fișierul generat folosind validatorul ANAF și corectați eventuale erori. Documentați în aplicație cum se generează D406 și ce condiții trebuie să respecte utilizatorul (de ex. să aibă completate CUI la parteneri, codurile de țară etc.).
32. **Alte declarații:** Verificați dacă trebuie ajustat ceva pentru D394. D394 cere să raportezi facturile către populație și dacă au fost achitate cu card sau numerar. Cum aplicația clasifică metodele de plată, se poate extrage: toate facturile către persoane fizice cu încasare cash => totalul acestor încasări. Asta e însă extra-cerință, poate fi abordată ulterior în modulul de rapoarte fiscale.
33. **Optimizare UI și flux de lucru casierie:**
34. Asigurați că interfața Registru de casă este ușor de folosit de către un casier non-tehnic. De exemplu: la adăugarea unei încasări, dacă se selectează transactionPurpose = customer\_payment, ar fi eficient ca aplicația să permită alegerea directă a facturii neîncasate dintr-o listă (filtrată pe client). Astfel se autopopulează personName și amount. Similar, la plată furnizor, să alegi factura de plătit. Aceste îmbunătățiri reduc erorile de introducere și cresc viteza.
35. Pentru plăți de salarii (dacă se procesează prin casierie), integrați cu modulul de salarii astfel încât la generarea statului de plată, pentru cei cu plata cash, să se creeze dispoziții de plată individuale sau cumulative. Legea permite plata salariilor în numerar în limita plafonului (salar de obicei < plafon, deci e ok), dar impune CNP pe chitanță – ceea ce deja aplicația cere prin personIdNumber la salary\_payment. Totuși, dacă se plătesc mai mulți angajați, registrul va avea o tranzacție separată pentru fiecare (ceea ce e corect, pentru că fiecare semnează de primire). Trebuie deci pregătit modulul de salarii să emită acele dispoziții (sau casierul să le introducă manual pe fiecare angajat – mai anevoios). E un punct de integrare inter-modulară pentru viitor.
36. **Avansuri și deconturi (Casa de avansuri):** Implementarea include tipuri speciale petty\_cash\_advance și petty\_cash\_settlement, și de asemenea tip casierie “petty\_cash” (casă de avansuri delegat). Verificați că fluxul e închegat: când se acordă un avans (petty\_cash\_advance) unui angajat, creați și o entitate de tip decont intern eventual. La închiderea decontului (petty\_cash\_settlement), tranzacția poate fi fie o plată (dacă angajatul returnează bani necheltuiți) – deci o încasare în casierie centrală a restului, fie o plată suplimentară (dacă angajatul a cheltuit din banii personali peste avans). Asigurați că aceste scenarii pot fi acoperite (posibil implică modul de deconturi). Din perspectivă registru de casă, e important că se pot înregistra ambele tipuri de operațiuni și că sunt contabilizate pe cont 542 “Avansuri de trezorerie” – cont pe care îl puteți adăuga în maparea CASH\_ACCOUNTS (dacă lipsește).
37. **Suspendarea unei casierii:** Statusul “suspended” pentru cashRegister ar putea fi utilizat când un casier e plecat temporar sau casieria e în revizie. Clarificați comportamentul: probabil similar cu closed (nu se pot adăuga tranzacții), dar fără intenția de a o închide definitiv (nu se calculează closingBalance). Dacă nu e necesar, puteți ignora acest status sau implementa la nevoie (ex. nu permite tranzacții noi dacă e suspendată, dar permite redeschidere ulterioară cu același sold).
38. **Documentare și training:**
    * Actualizați **manualul de utilizare** al aplicației pentru modulul de casierie și bancă, incluzând noile funcționalități: cum se închide ziua, cum se tipărește registrul, ce fac diferitele tipuri de tranzacții, ce erori pot apărea (ex. plafon depășit) și cum se rezolvă (prin depunere la bancă, etc.). O bună înțelegere de către utilizatori asigură respectarea procedurilor legale.
    * Furnizați **ghiduri interne** pentru personalul contabil, arătând cum pot verifica oricând soldul scriptic vs monetar (ex. să bată soldul din aplicație cu banii din casierie zilnic), cum să facă un inventar de casă (aplicația poate genera rapid totalul de bancnote și monede dacă se adaugă modul de monetar – un “nice to have”), cum să folosească rapoartele de bancă pentru conciliere cu extrasele.
    * Încheiați implementarea cu un **audit final intern**: desemnați un contabil sau auditor să parcurgă noul modul, să simuleze o lună de operațiuni, și să verifice dacă toate cerințele sunt acoperite: Registrul de casă zilnic corespunde formularului oficial și poate fi prezentat oricând la un control ANAF, Jurnalul de bancă reflectă fidel extrasele, soldurile contabile coincid cu cele fizice, iar restricțiile legale (plafonare, documente justificative) sunt respectate. În baza feedback-ului, faceți ultimele ajustări.

Prin realizarea pașilor de mai sus, aplicația va avea un **Registru de Casă** și un **Registru/Jurnal de Bancă** complet implementate, conform tuturor standardelor românești actuale și gata să treacă cu brio atât testul practic al utilizatorilor casieri/contabili, cât și eventualele controale sau audituri. Această dezvoltare riguroasă va asigura nu doar conformitatea legală, dar și eficiența operațională și integritatea datelor financiare din sistemul GeniusERP.

**Surse:**

* Legea nr. 70/2015 – plafoane operațiuni de numerar[[3]](https://www.avocatnet.ro/articol_68880/Ce-plafoane-de-incas%C4%83ri-%C8%99i-pl%C4%83%C8%9Bi-in-numerar-se-aplic%C4%83-in-2025.html#:~:text=Totodat%C4%83%2C%20opera%C8%9Biunile%20de%20%C3%AEncas%C4%83ri%20%C8%99i,c%C4%83tre%2F%20de%20la%20o%20persoan%C4%83)[[4]](https://www.avocatnet.ro/articol_68880/Ce-plafoane-de-incas%C4%83ri-%C8%99i-pl%C4%83%C8%9Bi-in-numerar-se-aplic%C4%83-in-2025.html#:~:text=Apoi%2C%20Legea%2070%2F2015%20impune%20limite,termen%20de%20dou%C4%83%20zile%20lucr%C4%83toare).
* OMFP 2634/2015 – Norme specifice utilizare documente financiar-contabile (Registru de casă)[[1]](https://infotva.manager.ro/articole/studii-de-caz/registru-de-casa-2023-registrul-se-intocmeste-centralizat-in-cazul-firmelor-cu-mai-multe-puncte-de-lucru-23081.html#:~:text=Nu%20exista%20nicio%20reglementare%20in,Registrul%20de%20casa%20serveste%20ca)[[2]](https://infotva.manager.ro/articole/infotva/registru-casa-23551.html#:~:text=Continutul%20minimal%20obligatoriu%20de%20informaii,al%20formularului%20este%20urmtorul).
* OMFP 2861/2009 – Inventarierea patrimoniului (verificarea cash-ului la sfârșit de an)[[5]](https://infotva.manager.ro/articole/infotva/registru-casa-23551.html#:~:text=proprii%2C%20aprobate%20prin%20OMFP%20nr,2861%2F2009).
* Articole InfoTVA/Avocatnet – exemple de completare registru de casă și obligații legale.
* Codul fiscal, art. 114(2)i – tratament fiscal diferențe de casierie (venituri din alte surse)[[46]](https://infotva.manager.ro/articole/infotva/registru-casa-23551.html#:~:text=In%20acest%20moment%2C%20deoarece%20nu,reglati%20soldul%20de%20casa%20astfel). (Notă: diferențele de plus la casă pot fi considerate venit impozabil dacă nu se pot atribui unei surse, conform interpretării fiscale).
* Documentație internă GeniusERP (schema bazei de date și codul sursă) – implementare module casă și bancă.

[[1]](https://infotva.manager.ro/articole/studii-de-caz/registru-de-casa-2023-registrul-se-intocmeste-centralizat-in-cazul-firmelor-cu-mai-multe-puncte-de-lucru-23081.html#:~:text=Nu%20exista%20nicio%20reglementare%20in,Registrul%20de%20casa%20serveste%20ca) [[6]](https://infotva.manager.ro/articole/studii-de-caz/registru-de-casa-2023-registrul-se-intocmeste-centralizat-in-cazul-firmelor-cu-mai-multe-puncte-de-lucru-23081.html#:~:text=Registrul%20de%20casa%20nu%20se,de%20la%20punctele%20de%20lucru) Registru de casa 2023: Registrul se intocmeste centralizat in cazul firmelor cu mai multe puncte de lucru?

<https://infotva.manager.ro/articole/studii-de-caz/registru-de-casa-2023-registrul-se-intocmeste-centralizat-in-cazul-firmelor-cu-mai-multe-puncte-de-lucru-23081.html>

[[2]](https://infotva.manager.ro/articole/infotva/registru-casa-23551.html#:~:text=Continutul%20minimal%20obligatoriu%20de%20informaii,al%20formularului%20este%20urmtorul) [[5]](https://infotva.manager.ro/articole/infotva/registru-casa-23551.html#:~:text=proprii%2C%20aprobate%20prin%20OMFP%20nr,2861%2F2009) [[7]](https://infotva.manager.ro/articole/infotva/registru-casa-23551.html#:~:text=Soldul%20de%20casa%20al%20zilei,casa%20pentru%20ziua%20in%20curs) [[8]](https://infotva.manager.ro/articole/infotva/registru-casa-23551.html#:~:text=Se%20semneaza%20de%20catre%20casier,si%20a%20actelor%20justificative%20anexate) [[36]](https://infotva.manager.ro/articole/infotva/registru-casa-23551.html#:~:text=,contabil) [[46]](https://infotva.manager.ro/articole/infotva/registru-casa-23551.html#:~:text=In%20acest%20moment%2C%20deoarece%20nu,reglati%20soldul%20de%20casa%20astfel) Registrul de Casa: ce Este si cum se Completeaza

<https://infotva.manager.ro/articole/infotva/registru-casa-23551.html>

[[3]](https://www.avocatnet.ro/articol_68880/Ce-plafoane-de-incas%C4%83ri-%C8%99i-pl%C4%83%C8%9Bi-in-numerar-se-aplic%C4%83-in-2025.html#:~:text=Totodat%C4%83%2C%20opera%C8%9Biunile%20de%20%C3%AEncas%C4%83ri%20%C8%99i,c%C4%83tre%2F%20de%20la%20o%20persoan%C4%83) [[4]](https://www.avocatnet.ro/articol_68880/Ce-plafoane-de-incas%C4%83ri-%C8%99i-pl%C4%83%C8%9Bi-in-numerar-se-aplic%C4%83-in-2025.html#:~:text=Apoi%2C%20Legea%2070%2F2015%20impune%20limite,termen%20de%20dou%C4%83%20zile%20lucr%C4%83toare) [[45]](https://www.avocatnet.ro/articol_68880/Ce-plafoane-de-incas%C4%83ri-%C8%99i-pl%C4%83%C8%9Bi-in-numerar-se-aplic%C4%83-in-2025.html#:~:text=servicii%2C%20precum%20%C8%99i%20cele%20reprezent%C3%A2nd,cum%20mai%20apare%20%C3%AEn%20lege) Ce plafoane de încasări și plăți în numerar se aplică în 2025

<https://www.avocatnet.ro/articol_68880/Ce-plafoane-de-incas%C4%83ri-%C8%99i-pl%C4%83%C8%9Bi-in-numerar-se-aplic%C4%83-in-2025.html>

[[9]](https://finlight.ro/docs/ghidul-saf-t-finlight/payments/#:~:text=Subsec%C8%9Biunea%20Payments%20,de%20tranzac%C8%9Bii%2C%20data%20tranzac%C8%9Biei%2C) Payments (Plăți) | Ghidul SAF-T Finlight

<https://finlight.ro/docs/ghidul-saf-t-finlight/payments/>

[[10]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L1270-L1278) [[11]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L1276-L1284) [[14]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L342-L350) [[15]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L374-L382) [[16]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L410-L418) [[17]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L412-L420) [[21]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L414-L422) [[22]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L88-L97) [[23]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L114-L122) [[28]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L1119-L1127) [[29]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L1129-L1137) [[30]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L1169-L1177) [[31]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L1179-L1187) [[32]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L1197-L1205) [[33]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L1414-L1422) [[34]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L1419-L1424) [[37]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L492-L501) [[41]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L382-L390) [[42]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts#L448-L456) cash-register.service.ts

<https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/cash-register.service.ts>

[[12]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/shared/schema/cash-register.schema.ts#L32-L40) [[13]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/shared/schema/cash-register.schema.ts#L48-L56) cash-register.schema.ts

<https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/shared/schema/cash-register.schema.ts>

[[18]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/client/src/modules/accounting/pages/cash-register/index.tsx#L618-L627) [[19]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/client/src/modules/accounting/pages/cash-register/index.tsx#L630-L638) [[20]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/client/src/modules/accounting/pages/cash-register/index.tsx#L640-L648) [[35]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/client/src/modules/accounting/pages/cash-register/index.tsx#L526-L534) index.tsx

<https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/client/src/modules/accounting/pages/cash-register/index.tsx>

[[24]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/backup/modules/accounting/services/bank-journal.service.ts#L151-L159) [[25]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/backup/modules/accounting/services/bank-journal.service.ts#L161-L169) [[26]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/backup/modules/accounting/services/bank-journal.service.ts#L172-L180) [[27]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/backup/modules/accounting/services/bank-journal.service.ts#L181-L190) bank-journal.service.ts

<https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/backup/modules/accounting/services/bank-journal.service.ts>

[[38]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/shared/schema/bank-journal.schema.ts#L18-L26) bank-journal.schema.ts

<https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/shared/schema/bank-journal.schema.ts>

[[39]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/bank-journal.service.ts#L75-L83) [[40]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/bank-journal.service.ts#L88-L96) bank-journal.service.ts

<https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/server/modules/accounting/services/bank-journal.service.ts>

[[43]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/SALES_JOURNAL_IMPLEMENTATION_COMPLETE.md#L114-L122) [[44]](https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/SALES_JOURNAL_IMPLEMENTATION_COMPLETE.md#L116-L124) SALES\_JOURNAL\_IMPLEMENTATION\_COMPLETE.md

<https://github.com/neacisu/GeniusERP_V_5_0_1/blob/b60ff0f7c0a3a37a5c3517a3c872dd2f9252c51d/SALES_JOURNAL_IMPLEMENTATION_COMPLETE.md>