A. Spital

1. Atunci când un pacient este internat in cadrul aplicației se face salvarea acestuia. Fiecare pacient are posibilitatea sa plătească extra pentru anumite facilitate precum: pat rabatabil, mic dejun inclus, papuci de camera, halat pentru interior. In cazul in care pacientul nu alege aceste facilitați extra, in cadrul aplicației sunt setate cu false. Să se dezvolte modulul care asigură crearea de obiecte de tipul pacient cu opțiuni extra.

2. În cadrul aplicației personalul spitalului este de mai multe tipuri. Acestea sunt salvate într-un enum { Brancardier, Asistent, Medic}. Să se implementeze modulul care pune la dispoziției crearea de obiecte din familia obiectelor PersonalSpital in funcție de tipul primit ca parametru.

3. În cadrul aplicației personalul spitalului este de mai multe tipuri: Brancardier, Asistent, Medic. Să se implementeze modulul care pune la dispoziției crearea de obiecte din familia obiectelor PersonalSpital. Modulul trebuie realizat, astfel încât pentru adăugări de noi tipuri de personal să nu fie necesare modificări în codul existent.

4. Spitalul are în dotare și un laborator în care mai mulți chimiști produc diferite rețete pentru medicamente. În momentul în care o rețetă este produsă trebuie ținut cont de cantitățile din soluțiile care sunt folosite. Dacă o rețetă este creată este recomandat să fie folosită pentru crearea viitoarelor medicamente fără a se trece prin procesul de creare al rețetei. Să se implementeze modulul care facilitează crearea de noi obiecte de rețete fără a fi nevoie de apelul constructorului.

5. Spitalul deține un mic magazin pentru medicamente și au o aplicație pentru cumpărarea d medicamente pe baza de rețetă. Spitalul închei un contrat cu o farmacie specializată și dorește să integreze sistemul informatic al farmaciei cu sistemul software existent în micul magazin de medicamente. Dezvoltatorii farmaciei trebuie să integreze aceste două aplicații, astfel încât aplicația farmaciei să poate folosi obiectele de tip Medicament din aplicația spitalului. Clasa Medicament din aplicația spitalului are metodele achizitioneazaMedicament() și prezintaReteta(). Metoda prezintaReteta() este apelată din achizitioneazaMedicament() pentru verificarea rețetei. Clasa Medicament din aplicația farmaciei are o singură metodă cumpărăMedicament() în care nu se face verificarea rețetei, deoarece farmacia o să vândă medicamente și fără rețetă.

6. Pentru internarea unui pacient în spital trebuie verificată gravitatea stării pacientului prin intermediul clasei Pacient, verificarea confirmării Medicului că trebuie internat, verificarea disponibilității unui pat în cameră prin intermediul clasei Salon care are lista cu paturile libere si ocupate. Spitalul dorește ca personalul spitalului să nu fi nevoit să facă aceste verificări separat ci să fie dezvoltat un modul care să permită această verificarea facilă a acestor lucruri.

7. Spitalul dorește să testeze punerea la dispoziție a rezultatelor prin intermediul platformei online ci nu doar printat. Există însă riscul să se revină la forma inițială de punere la dispoziția pacienților a rezultatelor. Se dorește adăugarea acestei noi funcționalități pentru sistemul software, care să permită revenirea la situația inițială.

8. Este dorita reprezentarea departamentelor spitalului in cadrul aplicației. Fiecare departament conține subdepartamente sau secții. Secțiile nu conțin subsecții. Sa se implementeze modulul care permite reprezentarea arborescenta a departamentelor si secțiilor spitalului.

9. Deoarece Spitalul este supraaglomerat se impune ca atunci când pacienții doresc internarea să fie internate doar personale care au asigurare de sănătate. Sa se realizeze un nivel intermediar care sa permită internarea doar acestor persoane.

10. Pentru fiecare internare trebuie să se rețină informații cu privire la pacientul internat precum: nume, număr de telefon, adresă, etc, precum și informațiile despre salonul unde este internat: număr salon, număr pat, număr zile spitalizare, etc. Astfel, dacă un pacient este internat de mai multe ori de-a lungul timpului, informațiile despre acesta sunt aceleași și se repetă, ocupând foarte multă memorie. Să se implementeze modulul de memorare al tuturor internărilor astfel încât să nu ocupe memorie foarte multă.

11. Este dorită implementarea modului de plată pentru pacienții care au fost internați în spital. Modul de plată îl decide persoana care plătește în momentul în care trebuie să facă plata. Plata se poate realiza cu cardul sau cash. Sa se implementeze modulul de plata.

12. Spitalul dorește să anunțe toți pacienții care au fost în spital ori de câte ori apare vreo urgență cu privire la viruși existenți în oraș. Astfel se dorește implementarea unui modul care atunci când apare o epidemie sau un virus nou să se trimită notificări tuturor persoanelor abonate la notificările spitalului.

13. Spitalul dorește implementarea unui modul pentru gestiunea pacienților. Un pacient poate să aibă una din următoarele stări: Internat, SubObservatie, Externat. Atunci când un pacient este adus în spital intră în starea Internat. Daca starea sa de sănătate este gravă, atunci este trecut în starea SubObservatie. Cand se vindeca si este trimis acasa, pacientul intra in starea Externat.

14. Internarea unui pacient se face după următorii pași: Se analizează dificultatea stării pacientului, Se verifica disponibilitatea in saloanele spitalului, Se emite fisa de internare. Sa se implementeze modului care realizează in aplicație internarea pacienților după acest pattern.

15. Managerul spitalului dorește sa grăbească procesul de la primiri urgente si astfel operatorul care primește pacienții la triaj va da comenzi de internare sau de tratare imediata pentru pacienții veniți si ii va așeza astfel la alte cozi separate. Comenzile sunt trimise către medici, însă operatorul poate primească alți pacienții mult mai rapid. Sa se implementeze modulul care permite trimiterea de comenzi către medici de la operatorul de primire.

B. Restaurant

1. Restaurantul servește mai multe tipuri de supe: supă de legume, supă de ciuperci, supă de vită, etc. Să se implementeze modulul care permite realizarea de obiecte din familia supelor. Modulul trebuie realizat, astfel încât pentru adăugări de noi tipuri de supă să nu fie necesare modificări în codul existent.

2. Atunci când un client face o rezervare poate alege una din următoarele opțiuni: așezare la geam, scaune ergonomice, decorarea mesei, muzica ambientala personalizata, gen muzica. În cazul în care clientul nu specifică vreun element dintre acestea, este setat pe false. Să se implementeze modulul care permite crearea de obiecte de tip rezervare cu aceste opțiuni extra.

3. Restaurantul servește mai multe tipuri de supe: supă de legume, supă de ciuperci, supă de vită, etc. Să se implementeze modulul care permite realizarea de obiecte din familia supelor. Tipurile de supă sunt reținute în cadrul unui enum.

4. Restaurantul dorește să implementeze un modul în cadrul aplicației, astfel încât dacă un client a mai fost la restaurant și revine pentru a fi realiza o rezervare aici să nu fie necesară reconstruirea unui cont respectivului client, deoarece prin construire clientului durează foarte mult.

5. Restaurantul achiziționează un nou soft nou pentru lucrul de la bar, însă acesta nu este compatibil cu softul de printare facturi folosit pentru produsele de la bucătărie. Vechiul soft era compatibil deoarece au fost realizate de aceeași echipa. Să se implementeze un nivel intermediar prin care noul soft să poată fi folosit cu softul existent, fără a se modifica codul din vreo aplicație.

6. În momentul în care un client vine la restaurant pentru o masă, recepționistul trebuie să verifice dacă are masă liberă, apoi să verifice dacă acea masă a fost debarasată de la plecarea ultimului client, de asemenea trebuie să verifice dacă au fost puse șervețele noi pe masă. Managerul restaurantului dorește realizarea unui modul care să simplifice munca recepționistului și să nu mai fie nevoit să verifice în toate locurile ci doar într-un singur loc.

7. Cu ocazia sărbătorilor de sfârșit de an managerul restaurantului dorește ca atunci când este printat o notă să se printeze și o felicitare de la Mulți ani pentru client. Se dorește adăugarea acestei noi funcționalități pentru clasa NotaDePlata la printare.

8. Este dorita reprezentarea meniului in cadrul aplicației. Meniul conține secțiuni (startere, băuturi, desert, etc.) fiecare secțiune poate conține subsecțiuni (sucuri, cafea, etc) sau item-uri (, apa plata, apa minerala, etc). Sa se realizeze modulul care permite reprezentarea arborescenta a meniului restaurantului.

9. Managerul restaurantului dorește ca atunci când cineva dorește să realizez e o rezervare sa fie permisă doar dacă aceasta este realizată pentru minim 4 persoane. În sens contrar rezervarea nu este realizata, iar persoanele sunt rugate să se prezinte la restaurant deoarece sunt suficiente locuri pentru mesele de doua persoane. Sa se realizeze un nivel intermediar care sa condiționeze realizarea rezervărilor de numărul de persoane.

10. Pentru fiecare rezervare trebuie să se rețină informații cu privire la clientul restaurantului precum: nume, număr de telefon, adresă de mail, etc, precum și informațiile despre masa rezervata: număr masa, număr persoane, ora rezervare, etc. Astfel, dacă un client realizează mai multe rezervări, la fiecare rezervare, informațiile despre client sunt aceleași și se repetă, ocupând foarte multă memorie. Să se implementeze modulul de memorare al rezervărilor astfel încât să nu ocupe memorie foarte multă.

11. Este dorită implementarea modului de plată pentru clienții restaurantului. Modul de plată îl decide clientul în momentul în care trebuie să facă plata. Plata se poate realiza cu cardul sau cash. Sa se implementeze modulul de plata al restaurantului.

12. Restaurantul dorește să anunțe clienții fideli ori de câte ori apar noi oferte. Astfel se dorește implementarea unui modul care atunci când se realizează o ofertă de preț sau se introduce un nou meniu să se trimită notificări tuturor clienților abonați la notificările restaurantului.

13. Restaurantul dorește implementarea unui modul pentru gestiunea modului de ocupare al meselor. O masă poate să aibă una din următoarele stări: Rezervata, Ocupata, Libera. Atunci când un client face o rezervare, masa trece in starea Rezervata. Când clientul ridica rezervarea masa trece in starea ocupata, iar când pleacă, masa trece in starea libera.

14. Ocuparea unei mese in restaurant se face după următorii pași: Se curata masa, Se așază șervetele, Se așază tacâmuri, sunt invitate persoanele sa se așeze la masa. Sa se implementeze modului care realizează in aplicație ocuparea meselor din restaurant.

15. Managerul restaurantului dorește ca rezervarea sau ocuparea meselor să se realizeze prin intermediul unui modul de comenzi conținut în clasa Operator. Sa se realizeze acest modul care permite trimiterea de comenzi către mese.

16. Restaurantul dorește să anunțe clienții fideli ori de câte ori apar noi oferte. Astfel se dorește implementarea unui modul sa notifice clienții restaurantului.
Problema este că restaurantul deține pentru anumiți clienți numărul de telefon, iar pentru alți clienți doar adresa de mail. Să se implementeze funcționalitatea de a trimite notificări clienților prin SMS, iar în cazul în care pentru anumiți clienți restaurantul nu are în baza de date numărul de telefon, să se trimită notificarea prin email. În cazul clienților pentru care nu există nici numărul de telefon, nici adresa de mail, se trimite managerului restaurantului o notificare cu numele clientului pentru care nu există date de contact.

C. Farmacie

1. Farmacia dorește ca toate medicamentele să facă parte dintr-o categorie. Categoriile sunt salvate într-un enum {Raceala, Durere, Body}. Fiecare medicament o sa aibă preț și denumire. Să se implementeze modul care va inițializa obiecte din familia medicamentelor.

2. Farmacia dorește ca toate medicamentele să facă parte dintr-una din categoriile: Raceala, Durere, Body. Fiecare medicament o sa aibă preț și denumire. Să se implementeze modul care va inițializa obiecte din familia medicamentelor. Modulul trebuie realizat, astfel încât pentru adăugări de noi tipuri de categorii să nu fie necesare modificări în codul existent.

3. Atunci când un client achiziționează o rețetă în aplicație se realizează un obiect de tipul factură. Un obiect de tipul factură conține și informații extra precum: numarPungi, daca acel client a cerut pungi pentru medicamentele achiziționate, plataCuCard, dacă este selectată o plată cu cardul ci nu cash, cardFidelitate, dacă este prezentat cardul de fidelitate al clientului, cotaTVA, dacă respectivul client este plătitor de TVA, în caz contrar o să fie trecut 0. Să se implementeze modulul care permite crearea de obiecte de tip factură cu aceste opțiuni extra.

4. Farmacia are în dotare și un laborator în care mai mulți chimiști produc diferite rețete pentru medicamente. În momentul în care o rețetă este produsă trebuie ținut cont de cantitățile din soluții sunt folosite. Dacă o rețetă este creată este recomandat să fie folosită pentru crearea viitoarelor medicamente fără a se trece prin procesul de creare al rețetei. Să se implementeze modulul care facilitează crearea de noi obiecte de rețete fără a fi nevoie de apelul constructorului.

5. Farmacia achiziționează un nou software pentru gestiunea stocurilor e medicamente din depozit. Însă această aplicație nu folosește aceleași clase ca și aplicația de vânzare folosită de farmacie. De exemplu aplicația de vânzare conține metoda setareMedicament(), care primește id-ul medicamentului și apoi verificareDisponibilitate() care primește numărul de medicamente dorite pentru medicamentul setat anterior și returnează true sau false. Noua aplicație de gestiune a stocurilor de medicamente in depozit are o singură metodă verificăStocPentruMedicament() care primește ID-ul medicamentului și numărul dorit și returnează true sau false. Să se implementeze modulul care permite lucrul celor două framework-uri fără a modifica codul existent.

6. Atunci când un client dorește să cumpere medicamente farmacistul trebuie să verifice în sistem rețeta clientului, trebuie să verifice disponibilitatea medicamentelor solicitate în depozitul farmaciei, și verificarea cardului de sănătate. Farmacistul trebuie sa verifice toate le trei lucruri in trei locuri diferite. Sa se implementeze un modul in cadrul aplicației care să permită farmacistului verificarea tuturor celor trei elemente într-un singur loc.

7. Cu ocazia sărbătorilor de sfârșit de an dirigintele farmaciei dorește ca atunci când este printat bonul de casă să se printeze și o felicitare de La Mulți Ani pentru client. Se dorește adăugarea acestei noi funcționalități pentru clasa BonDeCasa la printare.

8. Dirigintele farmaciei dorește organizarea medicamentelor într-o structura arborescenta pentru o căutare facila a acestora de către farmaciști. Astfel medicamentele vor fi organizate in cadrul aplicației pe secțiuni (Răceala, Durere, Antibiotice, etc) Fiecare secțiune conține subsecțiuni (Adulți, Copii, etc) sau medicamentele din acea categorie. Sa se realizeze modulul care permite reprezentarea arborescenta a medicamentelor in cadrul aplicației.

9. Deoarece stocul de medicamente este foarte mic se impune ca atunci când clienții doresc să achiziționeze medicamente să se realizeze doar pe baza de rețetă. Pentru clienții care nu au rețeta achiziționarea nu se va realiza. Sa se realizeze un nivel intermediar care sa permită cumpărarea de medicamente doar de către persoanele care au rețetă.

10. Pentru fiecare rețetă achiziționată de la farmacie trebuie să se rețină informații cu privire la client precum: nume, număr de asigurare, etc, precum și informațiile despre rețeta achiziționată precum: număr rețetă, suma de plata, număr medicamente, etc. Astfel, dacă un client achiziționează mai multe rețete de-a lungul timpului, informațiile despre acesta sunt aceleași și se repetă, ocupând foarte multă memorie. Să se implementeze modulul de memorare al tuturor achizițiilor de rețete, astfel încât să nu ocupe memorie foarte multă.

11. Este dorită implementarea modului de plată pentru clienți care achiziționează produse. Modul de plată îl decide clientul în momentul în care trebuie să facă plata. Plata se poate realiza cu cardul sau cash. Sa se implementeze modulul de plata.

12. Farmacia dorește să anunțe toți clienții abonați la notificările farmaciei cu privire la ofertele de preț pentru anumite medicamente. Astfel se dorește implementarea unui modul care atunci când apare o ofertă să se trimită notificări tuturor persoanelor abonate la notificările farmaciei.

13. Pentru fiecare rețetă se dorește gestiunea transformărilor atunci cand medicamentele sunt cumpărate. O rețetă poate sa fie într-una din următoarele stări: Emisa, Solicitata, Achizitionata. Starea Emisa este starea inițială a rețetei emise de medicul de familie. Starea Solicitata este starea in care ajunge atunci când clientul cere medicamentele la farmacie. Achizitionata este starea in are ajunge după ce medicamentele au fost cumpărate.

14. Achiziția de medicamente in Farmacie se face după o procedura bine stabilita care conține următorii pași: Se primește rețeta, se verifică în sistem stocul medicamentelor cerute, daca stocul este suficient se merge in depozit si se aduc medicamentele, daca stocul nu este suficient nu se face achiziția, se aduc medicamentele, se încasează banii, se scade din stoc si se emite bonul.

15. Deoarece se face o coada foarte mare la casa farmaciei, dirigintele acesteia dorește ca farmacistul care preia rețeta și sa trimită comenzi de aducere medicamente către ajutorul de farmacist. Ajutorul va prelua comenzile de aducere medicamente din depozit si la va aduce pe rând pe toate. In timp ce ajutorul de farmacist aduce medicamentele, farmacistul poate sa preia alte rețete. Sa se implementeze modulul de trimitere al comenzilor de către farmacist către ajutorul de farmacist.

D. Banca

1. Banca oferă clienților pachete pentru realizarea de credite de nevoie personale sau credite ipotecare. Să se implementeze modulul de construire de obiecte din familia creditelor.

2. Banca oferă servicii pentru persoane juridice și pentru persoane fizice. Să se implementeze modulul care construiește obiecte din familia de clienți ai băncii. Modulul trebuie realizat, astfel încât pentru adăugări de noi tipuri de clienți să nu fie necesare modificări în codul existent.

3. Banca pune la dispoziția clienților posibilitatea de creare de conturi pentru care dacă un client dorește poate să seteze să fie contul în care să primească salariul, să fie cu card atașat sau să aibă internet banking. În cazul în care clientul nu setează aceste informații, contul este creat fără aceste facilități.

4. Banca dorește să implementeze un modul în cadrul aplicației, astfel încât dacă un client deține deja un cont la bancă și dorește deschiderea unui nou cont să nu fie necesară reconstruirea datelor despre respectivul client, deoarece prin construire clientul trebuie să prezinte iar documentele necesare precum buletin și cardul de credit pentru plată.

5. Banca achiziționează un framework pentru oferirea de credite de leasing, însă clasele din acest framework nu sunt asemănătoare cu clasele utilizate de aplicația existentă. Se cere echipei IT să creeze un modul care să permită utilizarea claselor din framework-ul nou achiziționat, împreună cu clasele deja existente în aplicație.

6. Clienții care dețin cont bancar cu card atașat, pot realiza plăți online cu acel card, sau plăți la normale cu cardul. Banca decide să adauge o nouă funcționalitate cardurilor și anume plata ContactLess. Astfel se cere echipei IT să implementeze un modul care să adauge cardurilor această nouă funcționalitate.

7. Banca dorește simplificarea procesului de creare a unui cont. În momentul de față pentru crearea unui cont un operator de la bancă trebuie să verifice vârsta persoanei, să verifice dacă este urmărit de poliție sau dacă are creanțe la alte bănci. Să se realizeze un modul pentru simplificarea procesului pentru operator.

8. Banca ia hotărârea sa se realizeze credite doar în RON, deși în momentul de față oferă posibilitatea creării de conturi în orice monedă. Să se realizeze un nivel intermediar pentru clasa Credit, care să permită realizarea contului doar dacă se cere să fie în RON.

9. Să se implementeze structura ierarhică a agențiilor băncii știindu-se faptul că o sucursală deține mai multe agenții, iar o agenție deține mai multe filiale.

10. Pentru fiecare cont trebuie să se rețină informații cu privire la deținătorul său: nume, adresa, număr de telefon, adresă de mail, etc, precum și informațiile despre bancă: nume, sucursală, capital, etc. Pe lângă aceste informații un cont deține și informații precum număr cont, sumă etc. Astfel, dacă o persoană are mai multe conturi, la fiecare cont informațiile despre deținător sunt aceleași și se repetă, ocupând foarte multă memorie. Să se implementeze modulul de memorare al conturilor astfel încât să nu se ocupe memorie foarte multă.

11. Atunci când un client vine la banca, procesarea documentelor se realizează în funcție de tipul de client. Pentru clienții persoane fizice se solicită doar buletinul și adeverință de la munca, iar pentru clienții persoane juridice se solicita actele de înființare a firmei precum si dovada înregistrării la Registrul comerțului. Sa se implementeze modulul de verificare acte, știindu-se faptul că un client anunță când ajunge la ghișeu ce fel de client este. Deci stabilirea modului de procesare se stabilește la run-time.

12. Banca dorește să anunțe toți clienții abonați la notificări. Să se implementeze funcționalitatea de a trimite notificări clienților abonați. Acest proces se realizează pentru toți clienții băncii abonați la notificări.

13. Să se implementeze modulul de gestiune a bancomatelor. Bancomatul poate avea una din stările: areCard, nuAreCard, arePinIntrodus, nuAreBani. Atunci când o persoană dorește să retragă bani, bancomatul va trece prin aceste stări. Retragerea se poate face doar daca este in starea arePinIntrodus. Altfel nu se poate efectua retragere ci doar se poate trece într-una din celelalte stări. În cazul în care după retragere în bancomat nu mai sunt bani trece în starea nuAreBani și nu se mai pot face retrageri

14. Retragerile de bani de la bancomat se fac după următorii pași: se introduce cardul, se introduce pinul, se specifica suma solicitata, se retrag banii din bancomat și la final se retrage cardul. Sa se implementeze modulul care realizează acest algoritm.

15. Asupra unui cont bancar se dau comenzi de către operatorii băncii. Comenzile care se pot da sunt de constituire, retragere, depunere. Sa se realizeze modulul prin care se furnizează aceste opțiuni de trimitere a comenzilor către conturile bancare de către operatori. Modulul trebuie să permită și realizarea de *undo* pentru comenzile date.

16. Banca emite carduri care au mai multe conturi diferite. Se doreste ca un client sa isi poata seta ordinea in care vor fi folosite cele trei conturi. Astfel daca in contul curent nu o sa fie o suma suficienta de realizare a unei plati, plata respectiva va fi realizata din urmatorul cont setat de catre client. Daca nici in acesta nu este suma suficienta, se foloseste al treilea cont. Daca nici cel de al treilea cont nu are o suma suficient de mare, plata respectiva este refuzata.
Sa se implementeze modulul care permite clientului setarea ordinii in care vor fi folosite conturile pentru realizarea unei plati.

E. Sportiv

1. Se dorește implementarea unui modul care să inițializeze în cadrul aplicației obiecte din familia de obiecte Jucator. Categoriile de jucători existente sunt salvate într-un enum {Portar, Fundas, Atacant}. Să se implementeze acest modul care va crea obiecte din familia Jucator.

2. Se dorește implementarea unui modul care să inițializeze în cadrul aplicației obiecte din familia de obiecte Jucator. Categoriile de jucători existente sunt Portar, Fundas, Atacant. Să se implementeze acest modul care va crea obiecte din familia Jucator. Modulul trebuie realizat, astfel încât pentru adăugări de noi tipuri de jucători să nu fie necesare modificări în codul existent.

3. Atunci când un client face o rezervare poate alege una din următoarele opțiuni extra pentru locul său: mâncare inclusa, scaun ergonomic, bautura racoritoare inclusa, muzica ambientala personalizata, gen muzica. În cazul în care clientul nu specifică vreun element dintre acestea, este setat pe false. Să se implementeze modulul care permite crearea de obiecte de tip rezervare cu aceste opțiuni extra.

4. Firma de vânzare bilete dorește să implementeze un modul în cadrul aplicației, astfel încât dacă un client a mai cumpărat bilete și revine pentru a fi cumpăra un nou bilet să nu fie necesară reconstruirea unui cont respectivului client, deoarece prin construire clientului durează foarte mult. Să se implementeze modul care permite copierea datelor despre un client dintr-o instanță deja existentă.

5. Firma de vânzare bilete deține o aplicație pentru rezervarea de bilete și pentru vinderea de bilete la meciuri. Firma dorește să vândă bilete prin intermediul platformei de vânzare bilete: eBilet.ro, însă dezvoltatorii platformei spun că au interfețe pentru obiectele cu care lucrează acea platformă și trebuie creat un nivel intermediar pentru clasele existente în aplicația companiei, astfel încât, platforma să poată lucra cu obiecte furnizate de aplicația existentă.

6. Pentru verificarea unei persoane atunci când intră pe stadion un operator îi cere clientului buletinul ca să verifice dacă biletul este pe numele său, după care verifică daca acea persoană este căutată de poliție, în cadrul unei alte aplicații, apoi trebuie să verifice dacă persoana respectivă a avut antecedente pe alte stadioane. Managerul stadionului dorește să simplifice acest proces, iar operatorul să furnizeze într-un singur loc seria de buletin și codul de bilet, iar toate verificările să fie realizate acolo. Să se implementeze acest modul.

7. Compania dorește să adauge pe biletul printat un mesaj de susținere a echipei gazde. Compania dorește să adauge o nouă funcționalitate, astfel încât, la rezervarea biletului să fie printat acest mesaj doar atunci când joacă echipa locală. Pentru restul meciurilor biletele rămân la fel.

8. Pentru un meci de fotbal se ia decizia ca biletele sa fie vândute doar pentru persoanele care au minim 14 ani. Sa se implementeze un modul intermediar pentru clasa VanzareBilet prin care vânzarea de bilete sa fie condiționată de vârsta clientului.

9. Managerul stadionului dorește organizarea locurilor pe care stau spectatorii într-o structura arborescenta. Astfel locurile vor fi organizate in cadrul aplicației pe secțiuni (Tribuna, Peluza, etc) Fiecare secțiune conține subsecțiuni (Tribuna Nord, Tribuna Sud, Tribuna Copii, VIP, etc) sau direct locurile din acea secțiune. Sa se realizeze modulul care permite reprezentarea arborescenta a locurilor spectatorilor in cadrul aplicației.

10. Managerul stadionului dorește realizarea unui joc pentru meciurile care au loc pe stadionul sau. Doreste realizarea publicului din tribuna si doreste ca tribuna sa fie plina la fiecare meci. Pentru desenarea unei persoane in public este necesara salvarea informatiilor cu privire la dimensiunile acesteia: inaltime, latime, etc precum și informatii privind pozitia acestuia în tribuna sau culoarea tricoului purtat. Sa se realizeze modulul care realizeaza desenarea persoanelor in public cu un consum optim al memoriei.

11. Atunci când un client vine la stadion, verificarea corporala si a biletului se realizează în funcție de tipul de spectator. Pentru spectatorii de la tribuna VIP, verificarea este realizata doar pentru bilet, pentru spectatorii e la tribuna verificarea se face doar pentru bagajele pe care le au, iar pentru spectatorii de la peluza verificarea se face atât pentru bagaje cat si pentru hainele purtate. Stabilirea modului de procesare se stabilește atunci când clientul ajunge la poarta stadionului, adică la un-time.

12. Managerul unei Sali de sport dorește să anunțe toți clienții abonați la notificări atunci când este programat un nou meci de fotbal, de handbal sau de volei. Să se implementeze funcționalitatea de a trimite notificări clienților abonați. Acest proces se realizează pentru toți clienții abonați la notificări.

13. Se dorește simularea ocupării locurilor in tribuna de către spectatori. Un loc poate sa aibă una din următoarele stări: Ocupat, Liber, Rezervat. Daca este in starea rezervat nu poate fi rezervat de către alt spectator. Daca este ocupat nu poate fi nici rezervat nici ocupat de altcineva ci doar eliberat, trecând astfel în starea liber. Un loc liber poate fi si rezervat si ocupat.

14. Intrarea pe stadion si ocuparea locurilor se face pentru toata lumea la fel urmând pașii: Se așază la coada, Se prezinta biletul, Se face controlul corporal, Se intra in stadion, Se ocupa locul. Sa se realizeze modulul care simulează în aplicație procesul de intrare pe stadion și de ocupare a locurilor.

15. Pentru fiecare meci jucat pe stadion pentru vânzarea biletelor și pentru realizarea reclamelor se rețin următoarele informații: data meciului, echipele care au jucat, numărul de bilete vândute, numărul de spectatori care au participat la meci, numărul de sticle de apă vândute, numărul de jandarmi angajați, numărul de stewarzi, etc. Managerul stadionului dorește implementarea unui modul care să îi permită salvarea informațiilor pentru numărul de spectatorii care au participat la meci, numele echipelor care au jucat și data meciului. Își dorește să efectueze aceste salvări, astfel încât să îi permită la un moment revenirea la o salvare anterioară și să poată realiza anumite statistici.

F. Companie de transport in comun - STB.

1. Trebuie implementat un modul care sa creeze obiecte de tipul MijlocTransport: Autobuz, Tramvai, Troleibuz. Modulul trebuie sa ajute la crearea de obiecte de familia de clase MijlocTransport
Tipurile de transport sunt salvate intr-un enum{Autobuz, Tramvai, Troleibuz}.

2. Pentru crearea de obiecte de tip Autobuz sunt consumate foarte multe resurse. Din acest motiv trebuie propusa o varianta prin care daca exista deja un obiect creat, sa fie folosit acest obiect pentru viitoarele obiecte, fara a mai fi nevoie de crearea de la 0 a obiectelor de tip Autobuz.

Aceeasi regula se va aplica si pentru alte tipuri de mijloace de transport.

3. Pentru obiectele de tipul AutobuzLinie se doreste ca in momentul in care au fost create obiectele de acest tip sa nu mai poata fi modificate. De asemenea pentru crearea unui obiect de tipul AutobuzLinie trebuie precizat modelul de autobuz folosit, soferul care il va conduce, daca va avea opriri la capat de linie, daca deschide usile in fiecare statie fara ca pasagerii sa solicite acest lucru, textul afisat de ecranul derulator si alte elemente. Unele atribute sunt optionale avand o valoare prestabilita. Se doreste implementarea modulului care se va ocupa de initializarea acestor obiecte, nefiind necesara introducerea atributelor optionale, iar obiectul odata creat sa nu mai poata fi modificat.

4. Compania de transport in comun preia si infrastructura de Metrou a orasului si doreste ca biletele sau abonamentele achizitionate de catre clienti pentru transportul terestru sa poata fi folosite si pentru transportul subteran. Sisteme software ale celor doua moduri de transport sunt diferite si se doreste implementarea unui modul care sa permita utilizarea celor doua sisteme fara a le modifica. Modulul implementat trebuie sa se ocupe de validarea biletelor sau si a abonamentelor.

5. Autobuzele vin din fabrica cu 3 usi: usa din fata, usa de la mijloc si usa din spate. Pentru fiecare usa, soferul are cate un buton pentru punerea usilor in modul liber (sa poata fi deschise la solicitarea calatorilor) si un alt buton pentru deschiderea fortata a acestora. In total soferul are 6 butoane. Se doreste implementarea unui modul care sa simplifice procesul pentru sofer si sa aiba un buton pentru punerea tuturor usilor in modul liber si un alt buton pentru deschiderea fortata a tuturor usilor.

6. Pentru a gestiona garanția asigurată pentru fiecare autobuz disponibil în cadrul flotei companiei, se dorește implementrea unei soluții ce permite vizualizarea fiecărui autobuz în funcție de tipul acestuia si grupul pentru care este destinat. Structura definește o ierarhie între tipurile de autobuze

din flotă, grupate pe dimensiunea acestora (grupuri mici – 10 locuri, grupuri medii – 30 locuri, grupuri mari – 50 locuri). Fiecare autobuz este descris de producător, model si număr de locuri.

7. Compania doreste ca modulul de printare al biletelor sa permita cu ocazia anumitor zile nationale sa printeze pe bilet un mesaj de "La multi ani". Se doreste implementarea acestui modul care sa adauge functionalitatea de printare mesaj customizat. Exista posibilitatea ca aceasta functionalitate sa nu fie folosita, si din acest motiv se doreste sa fie optionala, fara modificarea codului existent.

8. Pentru autobuzele de noapte se doreste ca oprirea in statie sa se faca doar daca exista persoane in autobuz. In caz contrar autobuzul se retrage la autobaza fara sa mai realizeze opriri. Sa se implementeze modul care va permite oprirea in statie a autobuzelor doar daca exista calatori in autobuz.

9. Pentru fiecare AutobuzLinie sunt salvate in memorie, informatii precum: model autobuz, an fabricatie, numar locuri, numar linie,prima statie, ultima statie. Aceste informatii sunt salvate pentru fiecare AutobuzLinie din oras. Se doreste implementarea unui modul care sa asigure gestiunea in mod optim a memoriei, tinandu-se cont de faptul ca anumite informatii sunt redundante, deoarece pe o linie vor merge mai multe autobuze.

10. Un calator are posibilitatea sa plateasca cu cardul de calatorii, cardul bancar sau prin SMS. Trebuie implementat modulul dintr-un validator calatorie care sa permita plata calatoriei printr-una din cele trei metode. Calatorul va decide modul de plata atunci cand se urca in mijlocul de transport.

11. Se doreste implementarea unei aplicatii prin care utilizatorii se pot inregistra intr-o lista de calatori care sa fie anuntati atunci cand autobuzul pleaca de la capat de linie. Autobuzul cand pleaca in cursa de la capatul de linie trebuie sa anunte toti acesti calatori că s-a pus in miscare. Calatorii vor putea in acest mod sa stie cand trebuie sa astepte autobuzul in statie.

12. Trebuie implementat un modul care sa ii spuna calatorului ce mijloc de transport trebuie sa foloseasca in functie de distanta pe care o are de parcurs. Astfel, daca un calator are de parcurs o distanta mai mica de 3 km, este recomandat sa mearga cu Troleibuzul. Daca are o distanta cuprinsa intre 3 si 5 km i se recomanda sa foloseasca autobuzul, iar daca are o distanta cuprinsa intre 5 km si 10 km, i se recomanda sa foloseasca Tramvaiul. In cazul in care distanta este mai mare decat 10 km i se recomanda sa foloseasca Metroul.

Sa se implementeze acest modul in cadrul aplicatiei.

13. Pentru fiecare Autobuz se solicita salvarea acestora in memorie pentru posibilitatea de a reveni la o forma anterioara a obiectului respectiv. Se doreste implementarea unui modul care sa asigure aceasta salvare cu consum optim de memorie in conditiile in care este cunoscut faptul ca atributele care se vor modifica cu o frecventa mai mare sunt cele legate de soferul care il conduce, consumul mediu, etc. Celelalte atribute precum model, an fabricatie, numar locuri nu se vor modifca frecvent.

14. Un tramvai circula de fiecare data pe aceeasi linie, astfel el are de parcurs confom unui patern bine stabilit statiile de pe acea linie: Statia1, Statia2, Statia3, Statia4, Statia5, Statia6 atunci cand merge intr-un sens si invers atunci cand parcurge traseul in sens invers.

Sa se implementeze modului care asigura oprirea in statiile stabilite de catre patern pentru Tramvai.

15. Se doreste implementarea unui modul care sa gestioneze autobuzele din cadrul flotei in functie de starea in care se afla fiecare autobuz. Starile posibile sunt: InCursa, LaCapatDeLinie, LaReparat.

Un autobuz care se afla la capat de linie poate sa plece in cursa, insa un autobuz care este in cursa nu poate sa plece in cursa. Va putea pleca in cursa doar dupa ce ajunge la capat de linie. De asemneea un autobuz care este la reparat va putea s aplece in cursa doar dupa ce este reparat. Un autobuz care nu este in cursa nu are cum sa ajunga la capat de linie.

Modulul implementat trebuie sa tina cont de toate aceste conditii.

16. In cadrul unei autobaze este ceruta implementarea unui modul software prin care operatorul sa poata solicita plecarile pentru fiecare autobuz de la capat de linie si pe ce linie va merge (numarul liniei). Comenzile de plecare pe o anumita linie vor fi salvate intr-o colectie, iar in momentul in care autobuzul este disponibil acesta va prelua comanda de plecare in traseu.

Operatorul are posibilitatea sa stabileasca plecarile de la prima ora a zilei, astfel nu mai este necesara interventia acestuia pe timpul zilei.