Tema laborator TS .NET 2019

Vom simula (partial, in scop didactic) activitatea unui service auto. Pentru aceasta vom considera urmatoarele tipuri:

Client - caracteristici pentru un client

ClientId - intreg ce identifica in mod unic un client (generare automata)
Nume – string[15]
Prenume – string[15]
Adresa – string[50]
Localitate – string[10]
Judet – string[10]
Telefon – numeric[13]
Email – adresa de email, string[50]

Auto – caracteristici pentru un autoturism

AutoId – intreg ce identifica in mod unic auto (generare automata)
NumarAuto – string sub forma xx nnn yyy sau pentru cele cu numere provizorii xxnnnnnnnn
unde xx si yyy reprezinta litere [A-Z] iar nn reprezinta cifre.
Sasiu – tip descris mai jos
SerieSasiu - string[25]. Pozitiile 7 si 8 reprezinta CodSasiu. Are dimensiune variabila.
Ex. WAUZZZ**4F**Z6A111222

Sasiu

SasiuId – intreg ce identifica in mod unic un Sasiu
CodSasiu – string[2]. Ex “4F”
Denumire – string[25] Ex. Audi A6 2005-2010

Mecanic

MecanicId – intreg ce identifica in mod unic un mecanic
Nume – string[15]
Prenume – string[15]

Material

MaterialId - intreg ce identifica in mod unic un Material
Denumire – string[50]
Cantitate – decimal 10.2
Pret – decimal 10.2
DataAprovizionare – data calendaristica

Imagine – reprezinta imagini ale masinii in diferite momente de timp: la intrarea in service, executia unei operatii, descoperirea unui defect neidentificat la constatare

ImagineId - intreg ce identifica in mod unic o imagine
Comanda – tip Comanda
Titlu – string[15]
Descriere – string[256]
Data – data calendaristica
Foto – sir de bytes ce reprezinta imaginea

Operatie –

OperatieId - intreg ce identifica in mod unic o operatie
Denumire – string[256]. Ex. Schimbare placute frana fata.
TimpExecutie – decimal(6,2)

Comanda

ComandaID - intreg ce identifica in mod unic o comanda
Auto – tip Auto
Client – tip Client
Mecanic – tip Mecanic, colectie. Pot lucra mai multi mecanici la aceeasi masina.
Material – tip Material, colectie
Operatie – tip Operatie, colectie. Fiecare operatie va avea asociat mecanicul care o executa.
Imagine – tip Imagine, colectie.Optional. Anumite comenzi pot sa nu contina imagine.
StareComanda – {in asteptare, executata, refuzata la executie}.
DataSystem – reprezinta momentul introducerii comenzii in baza de date.
DataProgramare – data calendaristica, cand va incepe executia comenzii.
DataFinalizare – data calendaristica, cand se finalizeaza comanda.
KmBord – intreg 32 biti. Km din bord in momentul intrarii masinii in service.
Descriere – string[1024] : descriere defect plus cerinte client
ValoarePise – decimal(10.2) suma (Material.Cantitate * Material.Pret)

Fluxul de lucru poate fi urmatorul:

Un Client se prezinta la service cu masina pentru constatarea unor defecte sau intretinerea periodica (schimb ulei, filtre, etc.).
Se verifica daca clientul este in baza de date.

Daca clientul exista , atunci se afiseaza masinile pe care le are inregistrate si se selecteaza cea corecta. Daca masina nu exista se adauga. Se creaza o comanda in starea “in asteptare”. Se completeaza

DataProgramare, Descriere (ce trebuie executat la masina), eventuale foto, KmBord in cazul cand clientul lasa masina in service. Daca clientul nu lasa masina in service, atunci KmBord se vor completa la finalizarea comenzii. Operatiile ce trebuiesc efectuate si mecanicii corespunzatori pot fi adaugati pe parcursul executiei avand in vedere ca in ziua programarii comenzii unii mecanici pot fi indisponibili (diverse motive). Materialele se vor adauga la comanda in momentul cand mecanicul incepe executia lucrarilor la masina.

Daca clientul nu exista, atunci se adauga clientul, se adauga Auto si in continuare se procedeaza ca mai sus.

Daca clientul inregistrat intr-o comanda nu se prezinta la data programata, atunci poate fi reprogramat. Daca clientul renunta atunci comanda va fi trecuta in starea “refuzata la executie” si in campul Descriere se va specifica un motiv. Aceeasi stare o va primi comanda si in cazul cand mecanicii decid ca nu pot executa anumite cerinte ale clientului sau anumite operatii. Campul descriere va contine si acest motiv. Conditia e sa nu se fi executat nici o operatie si sa nu se fi consumat material pe masina respectiva. Daca acest ultim caz nu e indeplinit atunci comanda poate fi finalizata numai cu operatiile executate.

Pentru fiecare entitate trebuie sa scrieti metode pentru: afisare, cautare, creare, modificare, stergere.

Identificati relatiile corecte dintre aceste tipuri si tratatile corect. Veti crea metode pentru a valorifica aceste relatii si sa aveti posibilitatea sa afisati astfel de informatii in aplicatiile pe care le veti crea : Windows Form, WPF, WCF cu WPF, ASP.NET.

Datele vor fi salvate intr-o baza de date, numita AUTO.

Din cauza ca nu cer sa editati rapoarte intr-un anumit format (pdf, excel, etc.) aplicatiile pe care le veti crea vor avea elemente in interfata grafica ce vor descrie sugestiv relatiile dintre entitati. Exemplu: Pentru un client selectat voi afisa toate masinile lui. Pentru fiecare din aceste masini voi afisa toate comenzile in care apare acea masina. Pentru o comanda din aceasta lista voi afisa toate operatiile si materialele aferente ei. Va veti axa pe controale ce contin colectii (de exemplu, ListView, DataGridView, TreeView, ListBox, etc.).

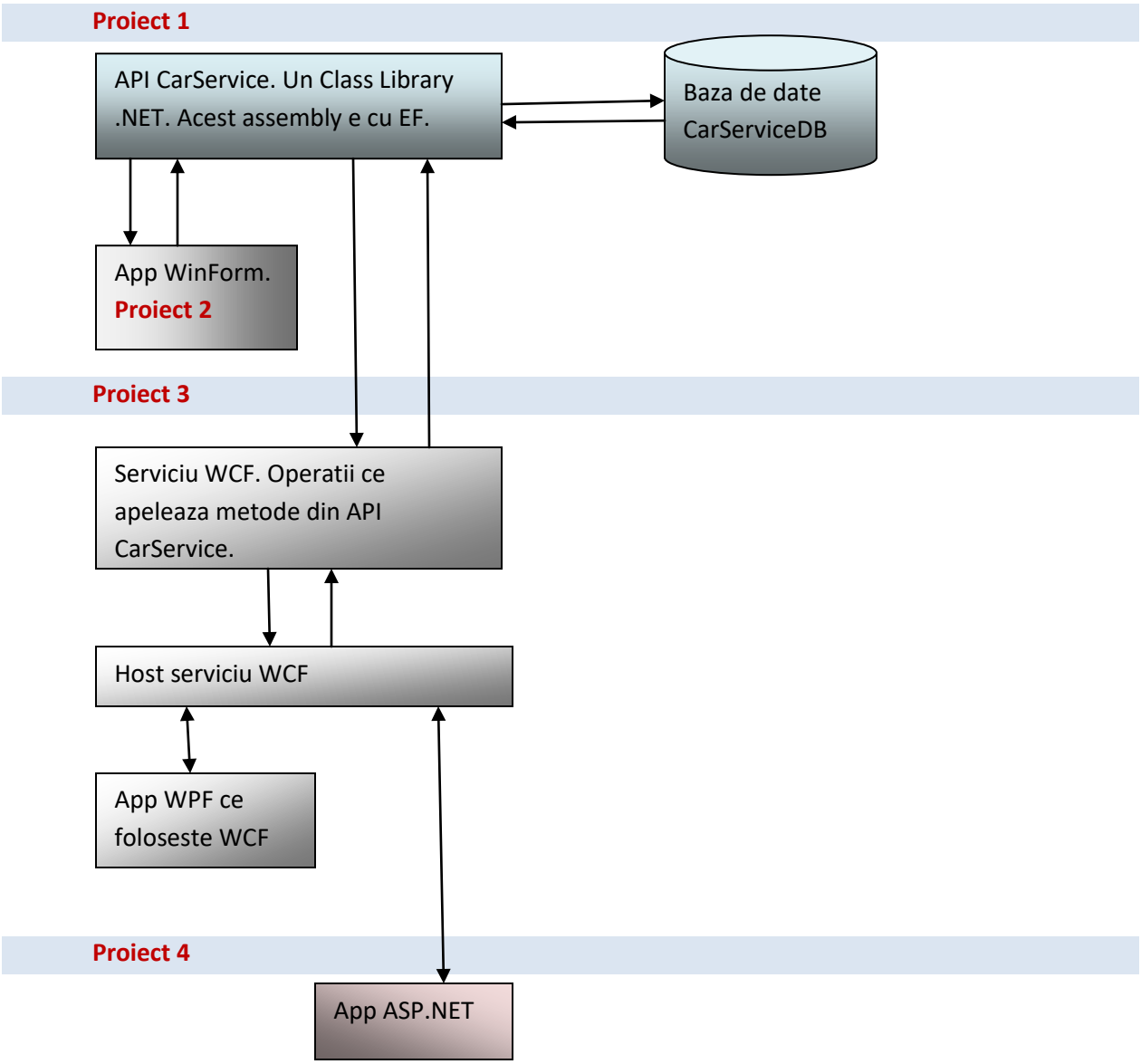
Pentru un material vreau sa stiu pe ce masini a fost folosit intr-o anumita perioada de timp, cantitativ si valoric.

Plecand din orice entitate din cadrul acestui graf de obiecte sa pot identifica fie clientul si masina, fie mecanicul, materialul, etc.

Proiectul 1

Veti realiza un assembly de tip Class Library .NET ce va expune un API. Metodele din acest API vor fi folosite in aplicatiile ce expun o interfata grafica. In final acest assembly e cu EF. Pentru inceput puteti folosi si alte obiecte pentru persistenta datelor. La EF veti folosi modelul de programare “Database First” sau “EF Model Designer First”. Nu veti folosi “Code First”.

Structura proiectelor este urmatoarea:



Proiectul 2 va folosi API CarService in cadrul unei aplicatii GUI cu Windows Form.

Proiectul 3 va crea un serviciu WCF folosind API din CarService, apoi va crea un host pentru serviciu si in final va exemplifica utilizarea serviciului in cadrul unei aplicatii GUI cu WPF.

Proiectul 4 va folosi serviciul WCF si va realiza o aplicatie web ASP.NET ce va expune functionalitatile din API CarService prin intermediul WCF.

Proiectele 2, 3 si 4 vor avea o interfata grafica asemanatoare, diferentele fiind date doar de tehnologia folosita.