Smart Admin Panel

# Tema Proiectului

Proiectul consta dintr-o aplicatie de tipul Dashboard/Admin Panel care poate fi folosita pentru adaugarea datelor intr-o baza de date reprezentand datele unei firme, precum si vizionarea datelor filtrate sub forma de rapoarte.

Pentru crearea aplicatiei am folosit biblioteca winForms din .NET, deoarece aceasta este tehnologia de dezvoltare aplicatii cu care sunt cel mai familiar.

Pentru crearea bazei de date am ales SQL Server, atat pentru ca sunt familiar cu T-SQL ca t si pentru ca este tehnologia cea mai potrivita pentru legatura cu aplicatii .NET

# Structura Bazei de Date

Baza de date este formata din 6 tabele, 5 proceduri stocate, 1 trigger si o functie auxiliara. Tabelele ce formeaza baza de date sunt:

* Angajati: contine date relevante despre angajati
* Departamente: contine date pentru fiecare departament din firma
* Plati: retine fiacare plata efectuata in cadrul firmei
* TipPlati: tabela folosita pentru a distinge diferitele tipuri de plata
* Valute: retine date despre diferitele valute folosite pentru efectuarea platilor
* Istoric\_angajari: tine un istoric cu toate angajarile. Este inserata o noua intrare de fiecare data cand se adauga un nou angajat

Structura bazei de date este descrisa de diagrama urmatorare:

Diagram

Description automatically generated

Constrangerile de integritate sunt :

Chei primare: Angajati(idAngajat), Departamente(idDepartament), Plati(IdPlata), Valute(idValuta) si tipPlati(idTipPlata)

Chei straine: Angajati(idDepartament)->Departamente, Plati(idAngajat)->Angajati,

Plati(idTipPlata)->TipPlati si Plati(idValuta)->Valute

Not Null: toate cheile primare, Angajati(nume,prenume,dataNasterii), Departamente(nume), Valute(numeValuta) si istoric\_angajari(NumeAngajat, dataAngajare)

Procedurile stocate sunt urmatorele:

* GetEmployeeData: primeste 2 dati si un id si face o interogare pentru angajatul cu id-ul dat ca parametru, pe langa date generice din tabela va afisa pentru angajat si toate formele sale de venit din perioada respectiva. Daca primeste -1 ca id va face interogarea pentru toti angajatii
* getInfoDepartamente: primeste 2 dati si face o interogare pe Departamente, selectand datele pentru fiecare departament, precum si costul acestuia reprezentat de totalitatea platilor efectuate in intervalul dat
* insertAngajat: adauga un angajat in tabela si, daca acesta are un departament nenul, actualizeaza numarul de angajati din departamentul respectiv
* insertPlata: adauga o plata in tabela Plati
* getAngajari Recente: selecteza cele mai recente 10 angajari efectuate din tabelul instoric\_angajare

Singura functie suplimentara din baza de date este getNumarManageri, care returneaza numarul de angajati dintr-un departament care sunt manageri

# Descrierea Aplicatiei

La baza aplicatiei sta forma Main\_panel, care reprezinta calea de acces catre celelalte forma. Aceasta este formata din urmatoarele componente:

* Profilul de admin cu imaginea de profil editabila in stanga sus
* Butonul ‘Raport Angajati’ care duce catre forma Dashboard
* Butonul ‘Statistici Departamente; care leaga panoul la forma Statistici\_Departamente
* Butonul ‘Adauga angajat/plata’ care deschide forma FormularInregistrareAngajat
* Un buton aditional ‘In caz de plictiseala’

Forma Main\_panel are urmatorul design

Graphical user interface

Description automatically generated

Pe langa butoane aceasta forma are si un grid ce arata istoricul celor mai recente angajari.

Forma Dashboard contine un raport al angajatilor bazat pe procedura GetEmployeeData, aceasta poate fi deschisa din Main\_panel. La rulare aceasta arata astfel

Graphical user interface, application

Description automatically generated

Aceasta forma este legata la forma FormularInregistrareAngajat, putandu-se accesa forme ulterioara prin righ-click si selectarea optiunii ‘adauga angajat’.

FormularInregistrareAngajat este o forma de tip submit, avand rolul de inserare a datelor in tabelele Angajati si Plati.

Aceasta arata astfel

Graphical user interface

Description automatically generated

La inserarea datelor se da refresh la toate campurile, anuntand utilizatorul ca adaugarea s-a facut cu succes.

Cel de al doilea raport este reprezentat de forma Statistici\_Deprtamente, care afiseaza un grid cu datele filtrate pe departamente, pe baza procedurii getInfoDepartamente, precum si un grafic cu veniturile pentru fiecare departament

La Deschiderea formei aceasta va arata astfel

Text

Description automatically generated

Am adaugat deasemenea o forma Boredom\_Button, care nu are functionalitate cu este doar un buton rosu ce poate fi apasat. Acesta arata astfel

Chart, bubble chart

Description automatically generated

Pe langa forme am creat 3 clase pentru retinerea informatilor aduse de proceduri.

InfoAngajat: folosit pentru raportul despre angajati. Acesta are structura:

* Id (int)
* isManager(string)
* varsta(int)
* nume(string)
* prenume(string)
* salarii(int)
* bonusuri(int)
* prime(int)

InfoDepartament: folosit pentru raportul pe departamente. Are structura:

* nume(string)
* nrManageri(int)
* nrAngajati(int)
* venit(double)
* costDepartament(double)

InfoIstoric: tine datele folosite in grid-ul din panoul main. Are structura:

* idAngajare(int)
* nume(string)
* DataAngajare(string)

Legarea aplicatiei la baza este asigurata de clasa DBAccess, care foloseste ORM-ul Dapper pentru maparea bazei de date la Clasele din aplicatie

# Surse

<https://youtu.be/Et2khGnrIqc>

[c# - Round shaped buttons - Stack Overflow](https://stackoverflow.com/questions/3708113/round-shaped-buttons)

[c# - How to move and resize a form without a border? - Stack Overflow](https://stackoverflow.com/questions/2575216/how-to-move-and-resize-a-form-without-a-border)

[c# - Make a borderless form movable? - Stack Overflow](https://stackoverflow.com/questions/1592876/make-a-borderless-form-movable)