## Descriere Problema

Problema 24 constă în dezvoltarea unei aplicații pentru organizarea de conferințe științifice, care va permite accesul a trei tipuri de utilizatori: participanți la conferință, organizatori ai conferinței și administratori. Participanții pot înscrie la conferință și vizualiza programul conferinței pe secțiuni fără autentificare, dar după autentificare, ei pot accesa volumul conferinței. Organizatorii pot efectua operații CRUD pentru persistența participanților și a lucrărilor prezentate, precum și filtrarea listei de participanți după secțiunea la care participă. Administratorii pot efectua operații CRUD pentru informațiile legate de utilizatorii aplicației care necesită autentificare și pot vizualiza lista tuturor utilizatorilor care necesită autentificare.

## Technologie aleasa

Pentru dezvoltarea proiectului de organizare a conferințelor științifice, am ales să folosim tehnologiile Windows Presentation Foundation (WPF) împreună cu limbajul de programare C#. WPF este un framework de dezvoltare de aplicații desktop pentru platforma Windows, iar C# este un limbaj de programare puternic și flexibil, care se integrează perfect cu WPF.

Prin utilizarea WPF, putem crea interfețe grafice moderne și atrăgătoare, care să ofere o experiență utilizator de înaltă calitate. WPF oferă un set bogat de controale și funcționalități pentru a construi interfețe complexe și interactive, precum și o arhitectură de legare de date flexibilă care facilitează manipularea și prezentarea datelor.

Prin intermediul limbajului C#, putem implementa logica de afaceri a aplicației într-un mod eficient și ușor de înțeles. C# oferă o sintaxă clară și concisă, suport pentru programare orientată pe obiecte și o gamă largă de funcționalități și biblioteci standard, ceea ce face să fie un limbaj potrivit pentru dezvoltarea aplicațiilor desktop.

Împreună, WPF și C# oferă un mediu de dezvoltare puternic și flexibil, care ne permite să construim o aplicație robustă, scalabilă și ușor de întreținut pentru organizarea conferințelor științifice. Această combinație de tehnologii ne oferă control deplin asupra aspectului și comportamentului aplicației noastre, și ne permite să răspundem eficient la cerințele și schimbările de business în timpul procesului de dezvoltare.

UseCase

In figura alaturata sunt prezentate patru tipuri principale de actori și responsabilitățile sau acțiunile asociate fiecăruia în sistem.

Imaginea ilustrează o diagramă de cazuri de utilizare pentru un sistem de organizare a conferințelor științifice, evidențiind patru categorii principale de actori și interacțiunile lor cu sistemul.

Primul actor, denumit simplu „Actor”, se referă la participanții la conferință. Ei au posibilitatea de a vizualiza programul conferințelor pentru a se informa despre sesiuni și evenimente și pot de asemenea să se înscrie la conferințe, o acțiune ce presupune probabil furnizarea de detalii personale și selecția evenimentelor de interes.

Următorul actor este „Utilizatorul”, care reprezintă persoanele înregistrate și autentificate în sistem. Responsabilitatea principală a acestui utilizator este accesarea volumelor de conferință, ce pot include diverse materiale și publicații asociate evenimentului.

Organizatorul joacă un rol cheie în gestionarea și coordonarea aspectelor administrative ale conferinței. Acesta are capacitatea de a căuta participanți după anumite criterii, cum ar fi secțiunea de interes, de a gestiona informațiile participanților și de a coordona lucrările prezentate la eveniment.

În sfârșit, există rolul de Administrator, care are responsabilități extinse în cadrul sistemului. Administratorul supraveghează conturile de utilizatori, gestionând activități precum crearea, modificarea și ștergerea acestora, și are posibilitatea de a vizualiza liste complete de utilizatori și detalii asociate.

Toți acești actori interacționează cu sistemul printr-un proces de „Login”, care le permite să acceseze funcționalitățile sistemului adecvate rolurilor și responsabilităților lor. O imagine care conține text, diagramă, linie, captură de ecran

Descriere generată automat

Diagrama UML

Imaginea alaturata este o diagramă UML care descrie arhitectura unui sistem de tip Repositor” pentru un model de organizare a conferințelor.

Clasa centrală este Repository, care conține metodele de bază pentru operațiile cu baza de date, cum ar fi deschiderea și închiderea conexiunilor, și executarea interogărilor. Aceasta servește drept clasă de bază pentru celelalte repository-uri specifice, care moștenesc aceste metode.Există patru repository-uri specifice în sistem:

PrezentareRepository Gerează datele legate de prezentări în sistem. Are metode specifice pentru adăugarea, obținerea, actualizarea și ștergerea prezentărilor. Aceasta interacționează cu clasa Prezentare, care stochează detaliile unei prezentări, cum ar fi id, titlu, autor, descriere, data, ora, secțiunea și identificatorul conferinței la care aparține. O imagine care conține text, captură de ecran, Paralel, software

Descriere generată automat

UtilizatorRepository: Se ocupă cu manipularea datelor utilizatorilor. Acesta include metode pentru adăugarea, obținerea și actualizarea utilizatorilor. Clasa „Utilizator” reflectă un utilizator individual cu detalii precum id, nume, email, parolă, tipul de utilizator, telefon și alte informații personale.

ConferintaRepository: Administrează datele despre conferințe. Aceasta conține funcționalități pentru a adăuga, obține, actualiza și șterge conferințe. Clasa „Conferința” reține informații despre o conferință individuală, inclusiv id, titlu, locație și data.

ParticipantRepository: Este responsabil pentru gestionarea participanților la conferințe. Include metode pentru a obține informații despre participanți, adăugarea și actualizarea acestora. Clasa „Participant” extinde clasa „Utilizator”, adăugând câmpuri suplimentare specifice participanților, cum ar fi id-ul și informațiile despre prezentările la care participă. O imagine care conține text, meniu, captură de ecran, document

Descriere generată automat

Imaginea prezentată ilustrează diferitele clase Presenter. Fiecare clasă Presenter gestionează interacțiunea dintre interfața utilizatorului și modelul de date, servind ca intermediar între interfața grafică și logica de afaceri.

Clasa MainPresenter este conectată la interfața principală a utilizatorului și include metode pentru schimbarea paginilor, afișarea diferitelor pagini precum cea de login, home, admin, organizator și participant, reflectând funcționalități diverse pentru navigarea în cadrul aplicației.

OrganizatorPresenter gestionează funcționalitățile legate de rolul de organizator în aplicație, având metode pentru lucrul cu prezentările, participanții și filtrarea informațiilor după anumite criterii, precum secțiunea evenimentului. De asemenea, aceasta clasă poate adăuga, șterge sau actualiza prezentările și participanții.

HomePresenter se ocupă de ecranul de start al aplicației, gestionând prezentările și participanții.

UtilizatorPresenter pare să se ocupe de funcționalitățile specifice utilizatorilor, cum ar fi încărcarea și configurarea conferințelor.

AdminPresenter se ocupă de funcționalități de administrare, cum ar fi crearea, ștergerea și actualizarea datelor utilizatorilor.

LoginPresenter gestionează logica de autentificare a utilizatorilor, inclusiv validarea datelor de logare și determinarea tipului de utilizator.

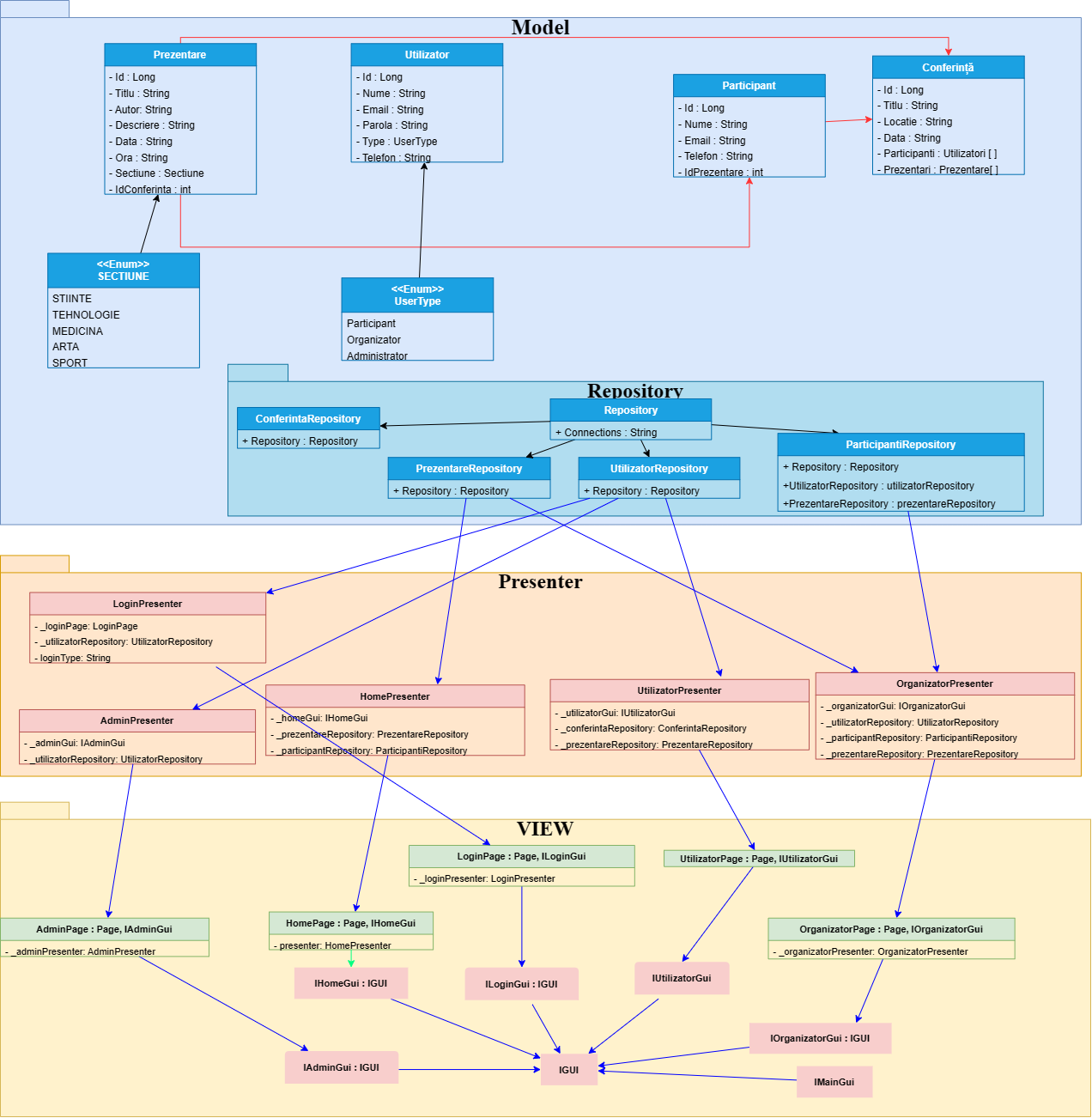


Diagrama UML prezentată este structurată pe mai multe niveluri care corespund cu arhitectura Model-View-Presenter ,detaliind modul în care diferitele componente ale aplicației colaborează pentru gestionarea informațiilor despre conferințe.

La nivelul de Model, găsim clasele de entități centrale ale sistemului. Clasa „Prezentare” stochează informații despre prezentările conferinței, în timp ce „Utilizator” deține detalii despre persoanele care interacționează cu sistemul. „Participant” este o extensie a „Utilizator”, sugerând că moștenește atributele clasei „Utilizator” și adaugă altele specifice participanților la conferință. Clasa „Conferință” este prevăzută să rețină informațiile esențiale ale unui eveniment, precum locația și data desfășurării.

La nivelul Repository, clasele „ConferintaRepository”, „PrezentareRepository”, „UtilizatorRepository” și „ParticipantRepository” extind clasa de bază „Repository”. Acestea sunt responsabile pentru manipularea persistentă a datelor: crearea, citirea, actualizarea și ștergerea informațiilor în baza de date.

Prezenterii funcționează ca intermediari între model și vizualizare. „LoginPresenter” se ocupă de logica autentificării, verificând credențialele utilizatorilor. „HomePresenter” controlează fluxul principal de activitate, probabil furnizând o vedere de ansamblu asupra conferințelor. „UtilizatorPresenter” gestionează activitățile specifice utilizatorilor, precum înscrierea la evenimente, iar „AdminPresenter” și „OrganizatorPresenter” oferă instrumente pentru administrarea sistemului și coordonarea evenimentelor.

Stratul View este alcătuit din interfețe grafice, fiecare fiind asociată cu un prezenter specific. Aceste GUI-uri permit utilizatorilor să interacționeze cu aplicația, introducând date și comenzi care sunt preluate de prezenterii respectivi.

Interacțiunea între aceste straturi începe când un utilizator acționează printr-un GUI. GUI-ul transmite cererea la prezenterul asociat, care apoi accesează un repository relevant pentru a prelua sau actualiza datele. Odată ce datele sunt procesate, prezenterul îi informează pe utilizatori despre rezultat prin actualizarea GUI-ului. Acest flux asigură separația logică între interfața utilizatorului, logica de afaceri și modelul de date, facilitând o structură clară și mentenabilă a codului.

## Descriere Aplicatie

Aplicația de management al conferințelor este proiectată pentru a oferi o experiență optimizată și completă tuturor celor implicați în procesul de organizare, participare și administrare a evenimentelor academice sau profesionale. Prin intermediul acestei platforme, fiecare utilizator are acces la funcționalități adaptate nevoilor și rolului său, de la înscriere la gestionarea detaliată a evenimentului.

În etapa inițială, participanții sunt întâmpinați de un formular de înscriere amplasat convenabil pe pagina principală. Acest formular este conceput pentru a fi cât mai simplu și accesibil, solicitând doar informații esențiale precum numele, e-mailul, numărul de telefon și o scurtă prezentare. Scopul este de a încuraja înscrierile prin minimizarea barierelor administrative, permițând în același timp organizatorilor să colecteze date relevante pentru configurarea și personalizarea experienței fiecărui participant.

După completarea formularului de înscriere, utilizatorii sunt direcționați către o pagină de login, unde își pot accesa contul introducând adresa de e-mail și parola. Această etapă de autentificare este esențială pentru a asigura securitatea datelor și pentru a oferi acces la funcții specifice, bazate pe tipul de utilizator: participant, organizator sau administrator. Depinzând de acest statut, utilizatorii sunt redirecționați către pagini personalizate, care le afișează informații și opțiuni relevante rolului lor în cadrul aplicației.

În cazul în care apar erori la autentificare, sistemul este programat să afișeze mesaje de eroare intuitive, care să îi ghideze pe utilizatori în rezolvarea problemelor întâmpinate. Aceasta asigură o experiență de utilizare fluidă și reduce frustrările care ar putea apărea în procesul de login.

Pe lângă facilitățile oferite participanților, aplicația pune un accent deosebit și pe panoul de administrare, oferind administratorilor un set complet de instrumente pentru gestionarea utilizatorilor și evenimentelor. În cadrul acestui panou, administratorii pot adăuga sau actualiza înregistrările utilizatorilor, având la dispoziție câmpuri pentru introducerea datelor esențiale precum numele, e-mailul, parola, tipul de utilizator și numărul de telefon. Aceasta permite o administrare eficientă și rapidă a conturilor, asigurând totodată o organizare corespunzătoare a evenimentelor.

La fel de importantă este și secțiunea de administrare a prezentărilor și participanților conferințelor. Administratorii au posibilitatea de a adăuga noi participanți și prezentări, completând informații precum numele, e-mailul, telefonul și identificatorul prezentării. Aceste funcționalități sunt esențiale pentru structurarea și planificarea detaliată a evenimentelor, permițând o organizare logică și accesibilă a conferințelor.

Model: Acest folder conține claselor care reprezintă datele și logica de afaceri a aplicației. De exemplu, Conferinta.cs, Participant.cs, Prezentare.cs, și Utilizator.cs sunt clase care probabil conțin proprietăți și metode specifice pentru gestionarea informațiilor despre conferințe, participanți, prezentări și utilizatori.

Repositories: folosește un pattern de repository pentru a abstrage logica de acces la date. Clasele precum ConferintaRepository.cs, ParticipantRepository.cs, PrezentareRepository.cs, și UtilizatorRepository.cs ar conține metode pentru interogarea și persistența datelor din și într-o bază de date sau alt mediu de stocare.

Folderul "Presenter" conține clase care gestionează interacțiunea între model și view. Clasele AdminPresenter.cs, HomePresenter.cs, LoginPresenter.cs, MainPresenter.cs, OrganizatorPresenter.cs, și UtilizatorPresenter.cs servesc ca intermediari între UI (User Interface) și datele aplicației, manipulând datele pentru a le prezenta în view.

View: Acest folder are de rolul de a stoca fișierele care se ocupă de interfața grafică cu utilizatorul.Interfaces contine interfețe precum IAdminGui.cs, IGui.cs, IHomeGui.cs, ILoginGui.cs, IMainGui.cs, IOrganizatorGui.cs, și IUtilizatorGui.cs, care definesc metodele și proprietățile ce trebuie implementate de orice clasă care prezintă un anumit tip de interfață utilizator.Folderul Pages conține fișiere XAML, care sunt folosite în WPF (Windows Presentation Foundation) pentru a defini elementele de UI și layout-ul acestora. Fiecare fișier XAML precum AdminPage.xaml, HomePage.xaml, LoginPage.xaml, OrganizatorPage.xaml, și UtilizatorPage.xaml reprezintă diferite pagini sau componente ale interfeței utilizatorului.

Clasa UtilizatorRepository: Este definită o clasă publică UtilizatorRepository care conține metodele și proprietățile necesare pentru a interacționa cu tabela utilizatori dintr-o bază de date.Variabile Membre: Clasa are două variabile membre private:Este o instanță a clasei Repository, care pare să fie o clasă de ajutor pentru executarea interogărilor și operațiunilor asupra bazei de date.

Metoda UtilizatorTable: Această metodă execută o interogare SQL pentru a selecta toate înregistrările din tabela utilizatori și le încarcă în utilizatoriTable. Există o verificare pentru a asigura că datele au fost returnate și că tabelul nu este gol.

Metoda rowToUtilizator: Converteste un rând (DataRow) din utilizatoriTable într-o instanță a clasei Utilizator, mapând fiecare coloană a rândului la proprietatea corespunzătoare a clasei Utilizator.

addUtilizator: Adaugă un nou utilizator în baza de date.

GetUtilizatori: Returnează o listă de utilizatori din baza de date.

GetUtilizatorById: Caută un utilizator după ID și îl returnează.

updateUtilizator: Actualizează datele unui utilizator existent în baza de date.

deleteUtilizator: Șterge un utilizator din baza de date.

GetUtilizatorbyNume: Caută utilizatori după nume.

GetUtilizatorsbyUserType: Returnează o listă de utilizatori filtrată după tipul de utilizator.

GetUtilizatorbyEmailandParola: Caută un utilizator după email și parolă.

GetUtilizatorbyEmail: Caută un utilizator după email.

Clasa AdminPresenter este responsabilă pentru manipularea interfeței de administrare a utilizatorilor.

Metoda CreateUser:Această metodă încearcă să creeze un nou utilizator folosind datele validate. Dacă adăugarea este reușită, un mesaj de succes este afișat utilizatorului prin interfața grafică, iar lista de utilizatori este actualizată. Dacă utilizatorul există deja, se afișează un mesaj de eroare.

Metoda DeleteUser :Această metodă șterge utilizatorul selectat din `DataGrid`. După ștergere, lista de utilizatori este reîncărcată și actualizată în `DataGrid`, iar un mesaj de confirmare este afișat.

Metoda validData:Validează datele de intrare preluate de la interfața utilizatorului. Verifică dacă id-ul este pozitiv, iar câmpurile nume, email, parola și telefon nu sunt goale. De asemenea, verifică dacă tipul utilizatorului este selectat. Dacă toate verificările sunt trecute, se creează și se returnează un nou obiect `Utilizator`; altfel, se afișează un mesaj de eroare și se returnează `null`.

Metoda UpdateUser: Actualizează datele unui utilizator existent cu noile valori validate. Dacă actualizarea este reușită, un mesaj de succes este afișat și lista de utilizatori este actualizată. Dacă actualizarea eșuează, se afișează un mesaj de eroare.

Metoda SetFormFields : Când un utilizator este selectat în `DataGrid`, această metodă setează câmpurile formularului cu valorile utilizatorului selectat, pregătind interfața pentru editare.