## Descriere Problema

Dezvoltați o aplicație care poate fi utilizată pentru organizarea de conferințe științifice. Aplicația va avea 3 tipuri de utilizatori: participant conferință, organizator conferință și administrator.

Utilizatorii de tip participant pot efectua următoarele operații fără autentificare:

* Înscrierea la conferință;
* Vizualizarea programului conferinței pe secțiuni.

Utilizatorii de tip participant pot efectua următoarele operații după autentificare:

* Accesarea volumului conferinței.

Utilizatorii de tip organizator conferință pot efectua următoarele operații după autentificare:

* Acceptarea sau respingerea unui participant (notificare prin email);
* Operații CRUD în ceea ce privește persistența participanților și a lucrărilor prezentate;
* Generearea programului conferinței pe secțiuni;
* Filtrarea listei de participanți după secțiunea la care participă;
* Salvare liste filtrate cu informații despre lucrările prezentate în mai multe formate: csv, json, xml, doc.

Utilizatorii de tip administrator pot efectua următoarele operații după autentificare:

* Operații CRUD pentru informațiile legate de utilizatorii aplicației care necesită autentificare;
* Vizualizarea listei tuturor utilizatorilor care necesită autentificare

## Instrumente utilizate

* Framework Windows Presentation Foundation (WPF)
* Limbajul de programare C#
* Baza de date PostgreSQL

## Justificarea limbajului de programare ales

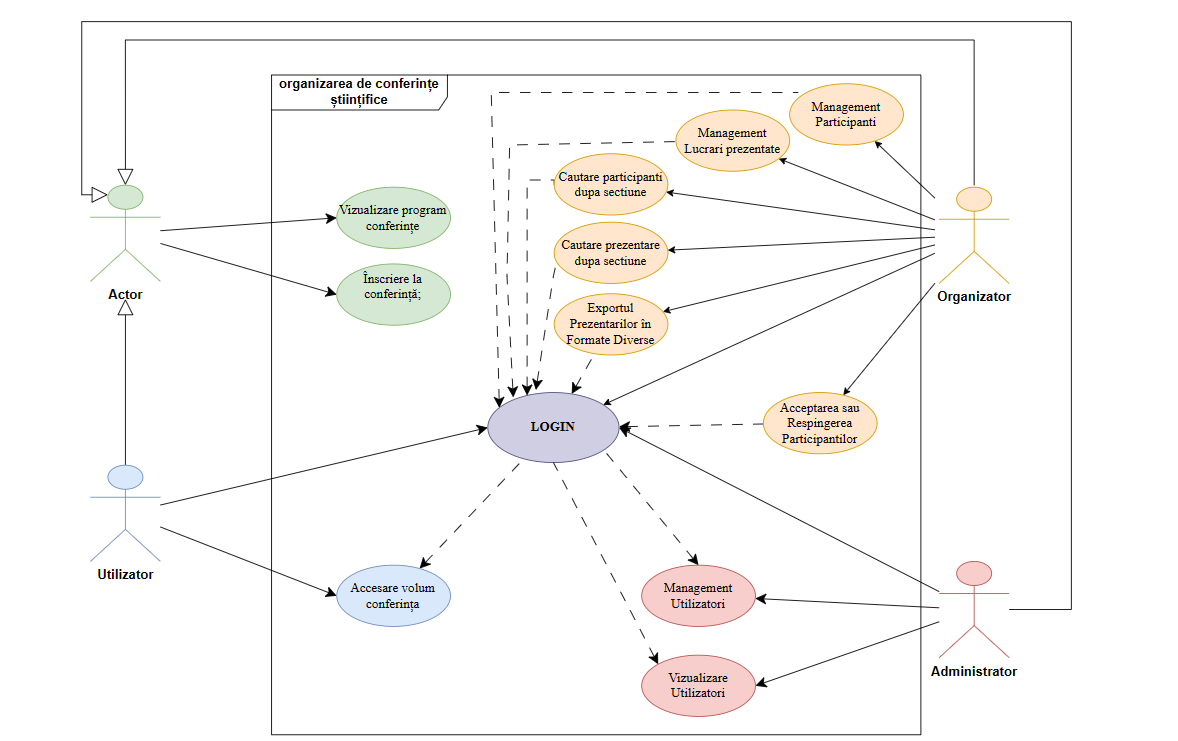
Pentru dezvoltarea proiectului de organizare a conferințelor științifice, am ales să folosim tehnologiile Windows Presentation Foundation (WPF) împreună cu limbajul de programare C#. WPF este un framework de dezvoltare de aplicații desktop pentru platforma Windows, iar C# este un limbaj de programare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

## Diagrama UseCase



Figură 1 Diagrama UseCase

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. În centru, "LOGIN" reprezintă punctul de intrare în sistem, unde utilizatorii trebuie să se autentifice pentru a accesa diferite funcții.

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 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.Ei dețin responsabilitatea de a administra baza de date a conferinței, executând operațiuni CRUD (Create, Read, Update, Delete) pentru gestionarea participanților și a lucrărilor prezentate. Acest lucru asigură că toate informațiile relevante sunt actualizate și accesibile. Unul dintre rolurile cheie este capacitatea de a accepta sau respinge participanții la conferință, cu posibilitatea de a trimite notificări prin email, ceea ce facilitează o comunicare eficientă și promptă. Pe lângă acestea, organizatorii au capacitatea de a genera programul conferinței, organizându-l pe secțiuni pentru o navigare mai ușoară și o mai bună organizare, filtrarea listei de participanți pe secțiuni este, de asemenea, o funcție disponibilă. În cele din urmă, organizatorii pot salva și exporta liste filtrate, împreună cu informații despre lucrările prezentate, într-o varietate de formate de fișiere cum ar fi CSV, JSON, XML și DOC.

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.

## Diagrama UML

* Model

O imagine care conține text, captură de ecran, software, Paralel

Descriere generată automat

Figură 2 Diagrama Model

Imaginea alaturata este o diagramă UML care descrie arhitectura unui sistem de tip Repository 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:

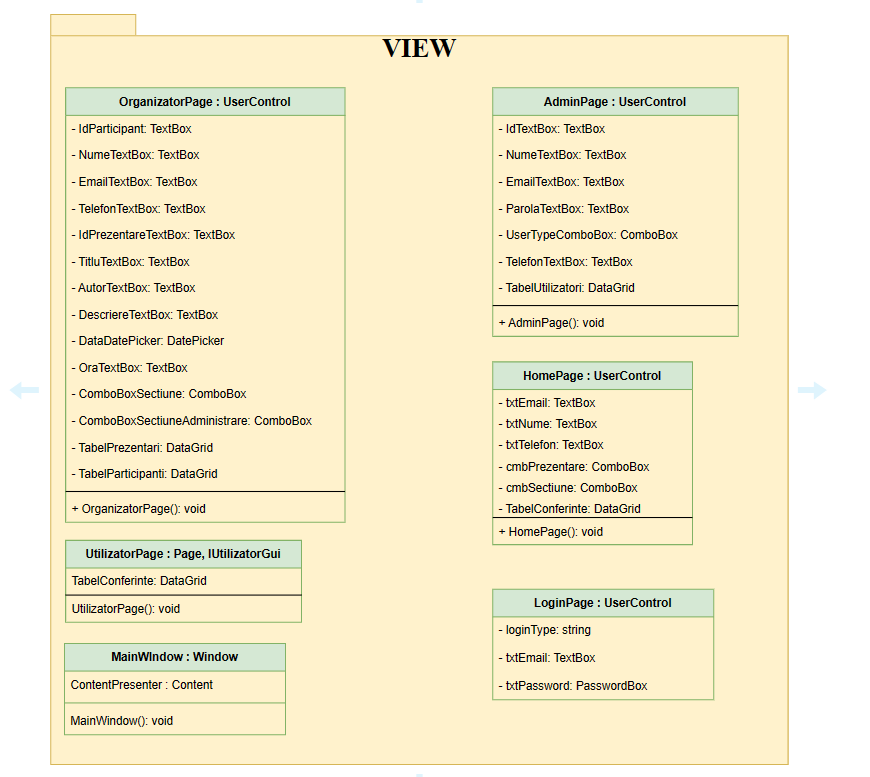
PrezentareRepository Gerează datele legate de prezentări în sistem. Are metode specifice pentru adăugarea, obținerea, actualizarea și ștergerea prezentărilor. Aceasta interacționează cu clasa Prezentare, care stochează detaliile unei prezentări, cum ar fi id, titlu, autor, descriere, data, ora, secțiunea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

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.

* View



Fiecare bloc reprezintă o componentă sau o fereastră distinctă în interfața utilizatorului, iar diagrama arată cum aceste componente se încadrează în ierarhia aplicației.

OrganizatorPage este o pagină destinată unui utilizator cu rol de organizator și conține o serie de controale TextBox, un DatePicker, ComboBox-uri și un DataGrid. Această pagină este folosită pentru introducerea și afișarea datelor referitoare la participanți și prezentări.

AdminPage este o pagină de administrare care conține controale TextBox pentru ID, nume, email, parolă și telefon, un ComboBox pentru tipul de utilizator și un DataGrid pentru a lista utilizatorii. Contine un panou de control pentru a administra conturile utilizatorilor.

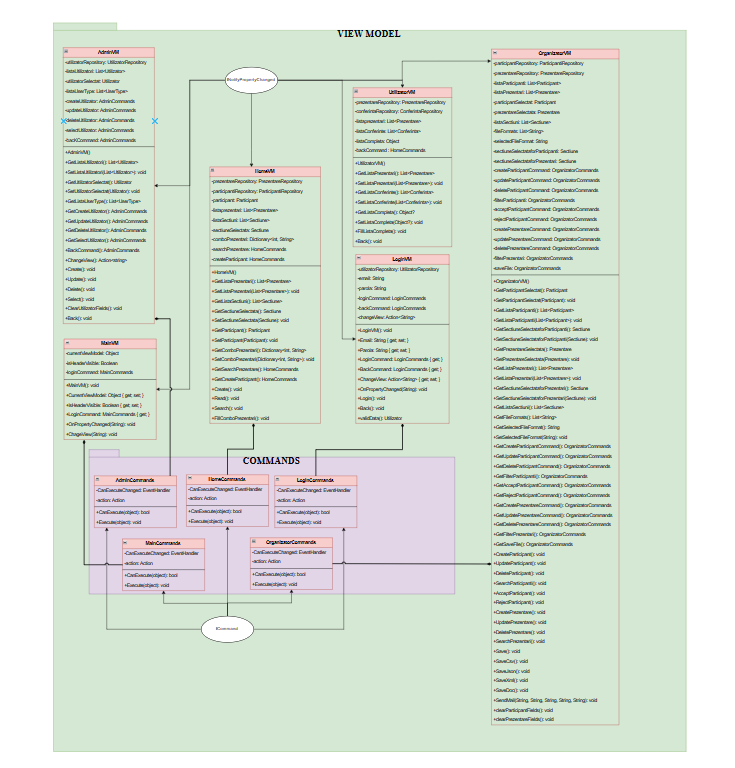
HomePage este pagina de start sau pagina principală a aplicației, care conține TextBox-uri pentru email, nume, telefon, ComboBox-uri pentru prezentări și secțiuni, și un DataGrid pentru a lista conferințele.

UtilizatorPage este o pagina specifică pentru utilizator, care include un DataGrid pentru conferințe.

MainWindow este fereastra principală a aplicației, care conține un ContentPresenter care este folosit pentru a schimba dinamic conținutul afișat în fereastra principală.

LoginPage este pagina de autentificare, care conține un șir pentru tipul de autentificare și controale pentru email și parolă, indicând că aceasta este utilizată pentru procesul de login.

* ViewModel



AdminVM este folosită pentru a gestiona starea și logica specifică pentru interfața de administrare a aplicației. Ea se ocupă de administrarea utilizatorilor, gestionând funcționalități precum adăugarea, actualizarea, și ștergerea acestora. Metodele sale vor interacționa cu un repository pentru a efectua operațiuni asupra bazei de date și vor notifica interfața de utilizator despre schimbările de stare prin intermediul evenimentelor PropertyChanged.

LoginVM Gestionează procesul de autentificare al utilizatorilor în aplicație. Deține proprietăți pentru colectarea datelor de autentificare (email și parolă) și comenzi pentru a efectua operația de login. În cazul în care datele sunt valide și autentificarea reușește, va comuta view-ul la interfața corespunzătoare tipului de utilizator autentificat.

MainVM acest ViewModel servește ca pivot central în aplicație, menținând referința către ViewModel-ul activ. Controlează vizibilitatea header-ului și schimbarea conținutului activ în funcție de interacțiunile utilizatorului.

OrganizatorVM acest ViewModel gestionează logica specifică pentru un utilizator cu rol de organizator. Include proprietăți și comenzi care permit organizatorului să gestioneze evenimente și participanți.

UtilizatorVM se ocupă de funcționalitățile disponibile pentru un utilizator obișnuit, permițând vizualizarea evenimentelor și altor informații relevante.

AddCommand, DeleteCommand, UpdateCommand Reprezintă setul de comenzi pentru CRUD (Creare, Citire, Actualizare, Ștergere) disponibile în ViewModel-uri. Acestea permit modificarea datelor printr-o interacțiune directă a utilizatorului cu interfața.

LoginCommand este o comandă specializată care gestionează logica de autentificare a utilizatorilor în sistem.

ChangeViewCommand această comandă este folosită pentru a schimba View-ul curent, permițând navigarea între diferitele ecrane ale aplicației.

Fiecărui ViewModel îi sunt asociate comenzi specifice care sunt invocate în urma interacțiunilor utilizatorului cu interfața grafică. Aceste comenzi sunt legate de butoane sau alte elemente de control, iar acțiunile executate de acestea reflectă schimbările de stare ale aplicației.

ViewModel-urile implementează interfața INotifyPropertyChanged pentru a notifica UI-ul despre schimbările de proprietăți care necesită actualizarea UI-ului. Atunci când o proprietate a ViewModel-ului se schimbă, o notificare este trimisă către UI pentru a reacționa corespunzător (de exemplu, actualizarea datelor afișate).

* Diagrama Clase

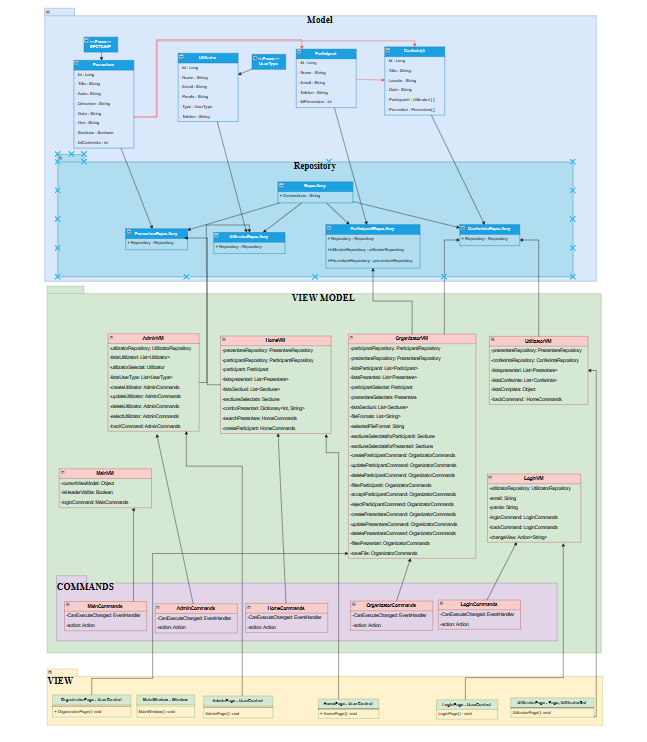
ViewModel-urile depind de modele și repository-uri pentru a obține și actualiza datele. Ele nu interacționează direct cu baza de date, ci prin intermediul repository-urilor.

Fiecare ViewModel conține referințe la obiectele Command necesare pentru a răspunde la acțiunile utilizatorului din View.

Commands folosesc ViewModel-uri pentru a executa acțiuni concrete, invocând logica de afaceri când sunt activate de evenimentele UI.

View-urile sunt legate de ViewModel-uri printr-un sistem de data binding. Orice actualizare în ViewModel trebuie să se reflecte în View și viceversa.

Modelele sunt folosite de ViewModel-uri pentru a actualiza View-ul. Nu există o legătură directă între modele și View-uri în MVVM; toate schimbările sunt mediate de ViewModel-uri.



## Descriere aplicație

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.

Secțiunea de administrare a conferințelor oferă organizatorilor un set robust de instrumente pentru gestionarea eficientă a participanților și a prezentărilor. Organizatorii pot adăuga noi participanți și prezentări, utilizând câmpuri dedicate pentru introducerea detaliilor precum nume, email, telefon și identificatorul prezentării asociate. Funcțiile avansate permit organizatorilor nu doar să adauge, dar și să accepte sau să respingă participanții, precum și să actualizeze sau să elimine înregistrările existente, facilitând un control detaliat asupra cine va participa la eveniment.Interfața include funcționalități de filtrare care permit organizatorilor să afișeze prezentările și participanții în funcție de secțiunea specifică a conferinței, facilitând organizarea tematică și îmbunătățind experiența participanților. În plus, există opțiuni pentru salvarea și exportul detaliilor prezentărilor în diferite formate de fișier precum CSV, JSON, XML și DOC, ceea ce le permite organizatorilor să distribuie programul conferinței și să colaboreze eficient cu alte părți interesate.

Clasele `ViewModel` sunt `AdminVM`, `HomeVM`, `LoginVM`, și `OrganizatorVM`. Fiecare dintre acestea gestionează logica de prezentare pentru diferite părți ale interfeței utilizator (UI). De exemplu, `AdminVM` poate include logica pentru gestionarea utilizatorilor și a prezentărilor, `LoginVM` se ocupă de autentificarea utilizatorilor, iar `OrganizatorVM` oferă instrumente specifice organizatorilor, cum ar fi acceptarea participanților și administrarea prezentărilor.

Clasele `Commands`, cum ar fi `LoginCommands`, `MainCommands`, `AdminCommands`, și `OrganizatorCommands`, par să fie folosite pentru a executa comenzi legate de acțiunile utilizatorului, cum ar fi logarea, schimbarea vizualizării, sau actualizarea datelor.

Clasele modelului, cum ar fi `Utilizator`, `Prezentare`, și `Conferinta`, reprezintă entitățile de bază ale aplicației. `Utilizator` ar putea reține detalii despre utilizatori, `Prezentare` despre prezentările din cadrul conferințelor, iar `Conferinta` ar putea include date despre conferințele însele.

Clasele `Repository`, cum ar fi `UtilizatorRepository`, `PrezentareRepository`, și `ConferintaRepository`, sunt probabil responsabile pentru operațiuni CRUD (Create, Read, Update, Delete) asupra bazei de date sau a altor forme de stocare persistentă, facilitând interogarea și actualizarea datelor.

`Enums` precum `UserType` și `Sectiune` sunt folosite pentru a defini seturi limitate de valori, cum ar fi tipuri de utilizatori (participant, organizator, administrator) și secțiuni ale conferinței (științe, tehnologie, medicină, artă, sport, etc.).

Fiecare `ViewModel` are multiple proprietăți și metode care permit UI să răspundă la acțiunile utilizatorului și să actualizeze datele prezentate, iar `Commands` oferă un mecanism prin care se pot executa acțiuni când sunt declanșate de interacțiuni în UI.

În cele din urmă, la nivel de arhitectură, putem spune că aplicația permite gestionarea utilizatorilor și a conferințelor cu multiple niveluri de acces și funcționalități specifice rolului fiecărui utilizator în sistem.