**UNIVERSITATEA “ALEXANDRU IOAN CUZA” DIN IAȘI**

**FACULTATEA DE INFORMATICĂ**

****

LUCRARE DE LICENȚĂ

**Student Assistant**

**propusă de**

***Marius-Cătălin Mihai***

**Sesiunea:** *februarie, 2020*

**Coordonator științific**

Colab. Florin Olariu

**UNIVERSITATEA “ALEXANDRU IOAN CUZA” DIN IAȘI**

**FACULTATEA DE INFORMATICĂ**

Student Assistant

*Marius-Cătălin Mihai*

**Sesiunea:** *februarie, 2020*

**Coordonator științific**

*Colab. Florin Olariu*

Avizat,

Îndrumător Lucrare de Licență

Titlul, Numele și prenumele \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Data \_\_\_\_\_\_\_\_\_\_\_\_ Semnătura \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**DECLARAȚIE privind originalitatea conținutului lucrării de licență**

Subsemntatul(a) ………………………………………………………………………………………

domiciliul în …………………………………………………………………………………………………..

născut(ă) la data de ………………..…., identificat prin CNP ………….……………..………………..., absolvent(a) al(a) Universității „Alexandru Ioan Cuza” din Iași, Facultatea de ………………………. specializarea …………………………………………………………, promoția …………………………., declar pe propria răspundere, cunoscând consecințele falsului în declarații în sensul art. 326 din Noul Cod Penal și dispozițiile Legii Educației Naționale nr. 1/2011 art.143 al. 4 si 5 referitoare la plagiat, că lucrarea de licență cu titlul: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_elaborată sub îndrumarea dl. / d-na \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, pe care urmează să o susțină în fața comisiei este originală, îmi aparține și îmi asum conținutul său în întregime.

De asemenea, declar că sunt de acord ca lucrarea mea de licență să fie verificată prin orice modalitate legală pentru confirmarea originalității, consimțind inclusiv la introducerea conținutului său într-o bază de date în acest scop.

Am luat la cunoștință despre faptul că este interzisă comercializarea de lucrări științifice in vederea facilitării fasificării de către cumpărător a calității de autor al unei lucrări de licență, de diploma sau de disertație și în acest sens, declar pe proprie răspundere că lucrarea de față nu a fost copiată ci reprezintă rodul cercetării pe care am întreprins-o.

Dată azi, ………………………… Semnătură student …………………………

DECLARAȚIE DE CONSIMȚĂMÂNT

Prin prezenta declar că sunt de acord ca Lucrarea de licență cu titlul „*Titlul complet al lucrării*”, codul sursă al programelor și celelalte conținuturi (grafice, multimedia, date de testetc.) care însoțesc această lucrare să fie utilizate în cadrul Facultății de Informatică.

De asemenea, sunt de acord ca Facultatea de Informatică de la Universitatea „Alexandru Ioan Cuza” din Iași, să utilizeze, modifice, reproducă și să distribuie în scopuri necomerciale programele-calculator, format executabil și sursă, realizate de mine în cadrul prezentei lucrări de licență.

Iași, *data*

Absolvent *Prenume Nume*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(semnătura în original)

C**uprins**

[Introducere 8](#_Toc31905840)

[Motivație 8](#_Toc31905841)

[Gradul de noutate 8](#_Toc31905842)

[Metodologia folosită 8](#_Toc31905843)

[Structura lucrării 9](#_Toc31905844)

[Contribuții 10](#_Toc31905845)

[Capitolul 1: Descrierea problemei 11](#_Toc31905846)

[Capitolul 2: Descrierea soluției 12](#_Toc31905847)

[2.1 Arhitectura aplicației 12](#_Toc31905848)

[2.1.1 Modulul Front End 13](#_Toc31905849)

[2.1.2 Modulul Back End 14](#_Toc31905850)

[2.1.3 Firebase Cloud Functions 17](#_Toc31905851)

[2.2 Tehnologii utilizate 18](#_Toc31905852)

[2.2.1 Android SDK 17+ 18](#_Toc31905853)

[2.2.2 Java JDK 1.7 împreună cu Language Level 8 18](#_Toc31905854)

[2.2.3 Firebase Authentication 18](#_Toc31905855)

[2.2.4 Firebase Realtime Database 19](#_Toc31905856)

[2.2.5 Firebase Cloud Functions 19](#_Toc31905857)

[2.3 Scenarii de utilizare a aplicației 20](#_Toc31905858)

[2.3.1 Înregistrarea în aplicație 20](#_Toc31905859)

[2.3.2 Utilizarea aplicației ca și student 22](#_Toc31905860)

[2.3.3 Utilizarea aplicației ca și profesor 24](#_Toc31905861)

[2.3.4 Utilizarea aplicației ca și administrator 26](#_Toc31905862)

[Concluziile lucrării 28](#_Toc31905863)

[Bibliografie 30](#_Toc31905864)

[Anexe 31](#_Toc31905865)

[Anexa 1 31](#_Toc31905866)

[Anexa 2 31](#_Toc31905867)

# Introducere

În zilele noastre, datorită avansurilor tehnologice și a faptului că trăim în era informației, dorim să fim conectați constant și să avem acces la informații într -un mod cât mai rapid, mai facil și mai centralizat. Ținând cont de acest aspect al cotidianului, consider că această normă a interconectării rapide și facile se aplică nu numai în viața noastră socială, cât și în ceea ce privește studiile noastre, mediul universitar fiind un candidat excelent pentru a aplica aceste noi standarde de comunicare și de transfer de informații.

## Motivație

Fiind un absolvent al acestei facultăți, și un pasionat al aplicațiilor mobile, mi -am dorit să creez acest mediu interconectat, în care informațiile sunt transmise rapid și facil atât către studenți cât și către profesori și în acest sens, am realizat această aplicație, Student Assistant.

Scopul aplicației este acela de a transmite informații în timp real de la profesori la studenți, prezentând celor din urmă toate informațiile referitoare la facultate, într -un mod centralizat și intuitiv.

## Gradul de noutate

Până la momentul actual, nu au fost implementate aplicații mobile ce doresc a oferi toate funcționalitățile aplicației Student Assistant, această fiind o inovație adusă facultății.

Este de menționat faptul că există servicii ce expun aspecte similare celor oferite de către Student Assistant (precum WebMail – transmiterea de anunțuri și mesaje, eSims – gestionarea notelor și a situației școlare, orarul FII), însă Student Assistant oferă avantajul portabilitatii și a centralizării datelor, atât studentul cât și profesorul având acces în timp real la aceste informații cu o simplă atingere de ecran.

## Metodologia folosită

Pe parcursul elaborării licenței am încercat să lucrez cât de cât structurat, pe etape, neavând în minte de la început o imagine clară a unui produs final. Am început cu luarea deciziilor referitoare la arhitectura aplicației, decizii precum o abordare server-less versus construirea și hostarea unui server, alegerea unei baze de date, reflectând în același timp asupra posibilelor funcționalități ce ar putea fi impactate de către luarea acestor decizii. În urmă acestor decizii am putut stabili cum vor fi implementate diversele funcționalități ale aplicației și am putut elabora o diagramă arhitecturală a aplicației, pe baza căreia am început dezvoltarea.

Atât pentru a avea un backup al aplicației, cât și pentru a putea lucra de pe mai multe calculatoare, am folosit Git-ul ca și sistem de versionare, asigurându-mă astfel că pot vedea istoricul schimbărilor, că pot face rollback la o versiune precedentă, în cazul în care o abordare a unei funcționalități a fost greșită și că la orice schimbare majoră a aplicației dispun de o versiune stabilă și funcțională a acesteia.

Timpul alocat dezvoltării aplicației nu a fost unul constant, obiectivele de timp alocat lucrării de licență fiind uneori ratate, dar depășite în alte zile. În mare parte, am alocat în medie două ore pe zi, iar dacă nu reușeam să îndeplinesc acest număr de ore, suplimentam în timpul weekend-ului și a vacanțelor.

Din disciplinele studiate în timpul facultății, în cadrul elaborării acestei lucrări s-au utilizat cunoștințe și concepte din următoarele:

* *Tehnici de programare pe platforma Android* – în ceea ce privește funcționalitățile specifice Android;
* *Programare avansată* – în ceea ce privește cunoștințele referitoare la limbajul de programare Java;
* *Programare orientată obiect* – oferă cunoștințe și principii ce stau la baza elaborării acestui proiect, întrucât Java este un limbaj de programare orientat obiect;
* *Ingineria programării* – alegerea și implementarea de arhitecturi și Design Patterns;
* *Structuri de date* – compararea diferitelor structuri de date utilizabile în cadrul implementării și alegerea celei mai potrivite dintre acestea;

## Structura lucrării

Lucrarea pe care o propun este structurată în două capitole în care sunt prezentate în detaliu elementele ce stau la baza creării acestei aplicații și interacțiunea dintre ele: ·

Capitolul 1 – prezintă pe scurt problemele existente ce au dus la crearea acestei aplicații. Practic aici vom vedea care sunt problemele ce pot fi rezolvate cu ajutorul lucrării propuse de mine.

Capitolul 2 – acesta va fi cel mai stufos capitol în care se vor prezenta pe rând arhitectura aplicației, tehnologiile utilizate pentru crearea aplicației și scenariile de utilizare ale acesteia.

# Contribuții

Contribuțiile mele în realizarea acestei lucrări de licență sunt următoarele:

* Crearea unei aplicații mobile pentru platforma Android, aplicație ce dispune de o interfață grafică prietenoasă și intuitivă, utilizatorul având, astfel, o experiență cât mai plăcută de-a lungul utilizării ei.
* Oferirea posibilității unui utilizator al aplicației de a-și crea un cont și de a se autentifica cu acesta pentru a accesa funcționalitățile aplicației.
* Oferirea posibilității unui profesor de a transmite informații referitoare la cursul, la laboratoarele sau seminariile acestuia, la orarul pe care această disciplină îl urmează, cât și la punctajele de laborator sau din examen studenților.
* Oferirea posibilității unui profesor de a partaja fișiere relevante disciplinelor sau laboratoarele sale, fișiere vizibile doar studenților înscriși la disciplina respectivă.
* Suport pentru notificări. Utilizatorul este notificat intermediul aplicației în momentul modificării unei discipline la care acesta este înscris, spre exemplu modificarea orarului, a notelor, a fișierelor partajate.
* Oferă posibilitatea profesorilor sau administratorilor aplicației de a trimite atât mesaje multicast, precum cele adresate unei grupe sau unui an, cât și mesaje individuale, adresate unui singur utilizator.
* Oferă studenților posibilitatea de a-și verifica notele, orarul, de a vedea informații și fișiere referitoare la disciplinele studiate, informații actualizate în timp real.
* Oferă studenților un ajutor în alegerea cursurilor opționale pentru semestrul următor de studiu, pe baza notelor obținute de aceștia în semestrele precedente.

# Capitolul 1: Descrierea problemei

Precum am menționat în introducere, nu există niciun motiv pentru a priva mediul universitar de un transfer facil, centralizat și în timp real de informații, un grad de conectare a participanților, atât a studenților cât și a profesorilor, aducând beneficii majore, simplificându-ne viața de zi cu zi.

Atât informațiile descentralizate ce sunt disponibile individual într-un număr mare de locații, lucru care face urmărirea acestora dificilă, cât și anunțurile ce nu au ajuns în timp util la destinatari reprezintă impedimente atât în viața unui student cât și a unui profesor, generând de multe ori probleme ambelor părți.

O simplă soluție la aceste probleme ar fi centralizarea acestor informații, astfel acestea devin mai ușor de accesat și de urmărit, și transmiterea lor în timp real împreună cu notificarea persoanelor pentru care acestea prezintă interes, pentru a elimina posibilitatea nerecepționării acestora în timp util.

# Capitolul 2: Descrierea soluției

În acest capitol voi prezenta pe ansamblu arhitectura aplicației, tehnologiile folosite pentru crearea aplicației și scenariile de utilizare ale aplicației.

## 2.1 Arhitectura aplicației

Aplicația descrisă în această lucrare este o aplicație Android, compusă din două module, modulul de Front End și modulul de Back-end, la acestea adăugându-se funcțiile din cadrul Firebase Cloud Functions.

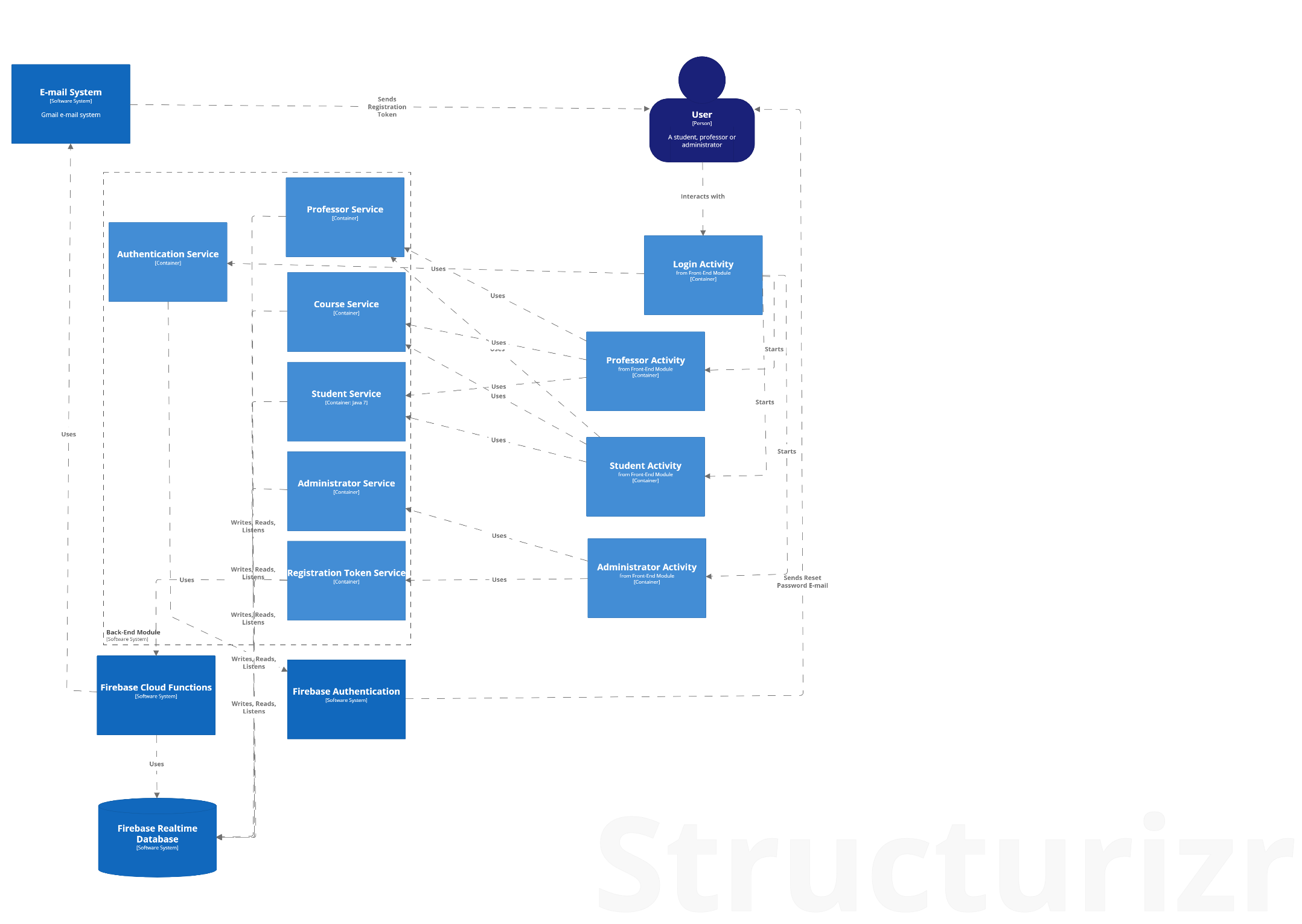
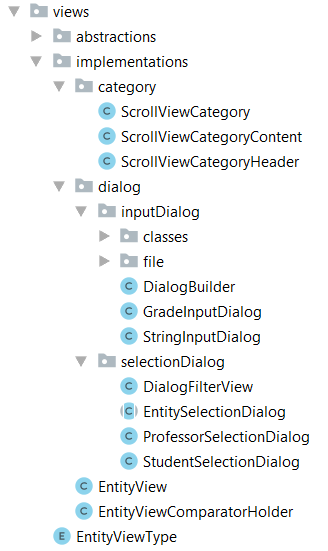


Figura 1 – Arhitectura Aplicației

### 2.1.1 Modulul Front End

Modulul de Front End expune utilizatorilor elementele de UI împreună cu notificările pe care aceștia le vor primi, acesta reprezentând Presentation Layer-ul aplicației. Elementul principal al acestui modul este

Activitatea, care în cadrul sistemului Android reprezintă un proces, firul său principal de execuție având responsabilitatea afișării și modificării elementelor grafice disponibile pe ecran. Elementele grafice disponibile în aplicație sunt descrise atât programatic cât și utilizând fișiere de tip XML.



Fiecare element de UI (view) generat programatic reprezintă o clasă independentă ce respectă Single Responsibility Principle, funcționalitățile acesteia fiind restrânse la propria sa entitate (figura 2).

Sincronizarea și modificarea datelor în timp real la nivelul acestor componente se realizează cu ajutorul tehnologiei de Data Binding, disponibilă în SDK-ul Androidului, tehnologie ce se bazează pe Design Pattern-ul de Observer. Fiecare instanță a unui View reprezintă un observator asupra unui set de date (Observable), iar la modificarea acestor date, fiecare dintre elementele de UI ce le folosesc vor fi actualizate.

Este de menționat faptul că majoritatea acestor elemente de UI sunt definite în cadrul fișierelor XML disponibile în aplicație, acestea definind structura generală a componentei de UI și stilul ei, acest lucru crescând gradul de reutilizare a codului.

Figura 2 – View-uri generate programatic

Fișierele de tip XML (figura 3) ce ajută la definirea elementelor grafice conțin taguri și atribute specifice platformei Android (taguri precum LinearLayout, RadioButton și atribute precum android:layout\_height, android:onClick, android:background), aceste fișiere fiind parsate la compile time cu scopul validării acestora, și la run time cu scopul creării elementelor grafice specificate în tagurile XML-ului, procedeu numit „inflation”.

La nivelul acestui modul am implementat arhitectura MVC (Model-View-Controller) în următoarea manieră:

* Model – reprezintă seturile de date utilizate de către Data Binding, modele ce reflectă entitățile afișate în View-uri.
* View – element grafic definit în fișierele XML și / sau în propria să clasa, acțiunile și evenimentele efectuate asupra lui fiind delegate Activității ce conține respectivul view.
* Controller – este reprezentat de către Activitate. această reacționând la evenimentele transmise de către View și efectuând acțiunile necesare modificării interfeței grafice în concordanță cu evenimentul transmis.

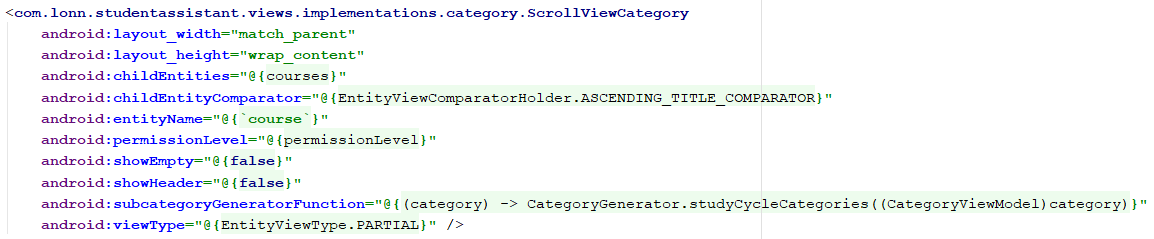


Figura 3 – Exemplu de utilizare a unui View generat programatic, cu atribute custom si valori setate prin Databinding

### 2.1.2 Modulul Back End

Modulul de Back End conține straturile de Business Logic și Data Access ale aplicației, acest modul fiind compus din o multitudine de servicii ce vor fi apelate de modulul prezentat anterior. Aceste servicii scriu și citesc date în /din Firebase Realtime Database, ascultând în același timp schimbările de date efectuate de către ceilalți utilizatori, modificând datele expuse în interfață grafică.

Acest modul expune către modulul de front-end 16 servicii, 13 dintre acestea modificând în Data Access Layer-ul lor câte un „tabel” al bazei de date (fiind o bază de date non- relațională, aceasta nu conține tabele precum o baza de date SQL, conceptul de tabel fiind reprezentat de un nod de pe al doilea nivel al arborelui JSON). Fiecare apel către servicii se realizeaz ă asincron, datorită naturii prin care API-ul Firebase comunică cu aplicația noastră, și anume prin Listeneri și metode de callback.

#### Business Logic Layer

Precum am menționat anterior, 13 dintre cele 16 servicii prezintă un Dată Access Layer propriu, iar la nivelul B.L.L. acestea asigura validitatea datelor, verificând câmpurile entităților, existența entităților agregate prin chei străine, etc. De asemenea, un serviciu nu accesează D.A.L.-ul altui serviciu, întrucât asta ar încalcă principiul Single Responsibility și principiul de Separation of Concerns(întrucât fiecare serviciu poate fi văzut drept un modul sau o componentă autonomă). În situația în care entitățile agregate transmise unui serviciu necesită validare la rândul lor, acesta va comunica cu serviciile destinate acestora, astfel arhitectura acestui modul poate fi asemănată cu o arhitectură bazat ă pe microservicii.

Este de menționat faptul că toate metodele expuse de servicii sunt asincrone, din motivul elaborat anterior, însă, neutilizand JDK 8+, interfețele funcționale prezente în acesta nu sunt disponibile(interfețe precum Future, Function, Comparable, Consumer, etc). În această situație s-a dovedit a fi utilă utilizarea Language Level-ului 8, ce permite definirea de interfețe funcționale proprii, implementând astfel interfețele funcționale necesare facilitării procesului de dezvoltare (aceleași interfețe enumerate anterior).

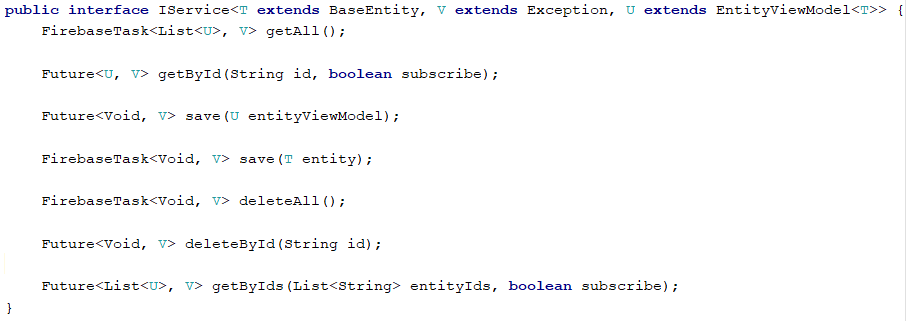


Figura 4 – Interfata implementata de serviciile cu D.A.L.

Din figura 3 se poate observa faptul că serviciile nu expun entitățile propriu-zise, ci un ViewModel al acestora, pentru a nu dezvălui detaliile de reprezentare a acestora din baza de date, ci a transmite doar date relevante consumatorilor.

De asemenea se poate observa absența unei metode de update, lucru datorat faptului că Firebase Realtime Database tratează metodă de save și cea de update î ntr-un mod similar.

#### Data Access Layer

Data Access Layer sau D.A.L. este stratul inferior al serviciilor, acesta intervenind între B.L.L. și baza de date, acesta ocupându-se de operațiile C.R.U.D. și de setarea Listenerilor asupra entităților din baze de date, entități referentiate prin path-ul absolut ce îl au în baza de date (Spre exemplu „/Files/Metadata/{uid}” este path-ul absolut către metadata cu id-ul „uid” al unui fișier).

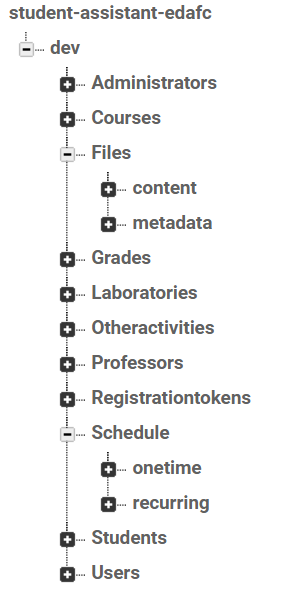
Pentru implementarea D.A.L. am optat pentru o abordare generică, întrucât metodele utilizate de acesta sunt similare, indiferent de entitatea asupra cărora se realizează acestea. În ceea ce privește partea de Design Patterns, nu am optat pentru Repository, întrucât memorarea datelor în acesta ar fi cauzat probleme de sincronizare, crescând gradul de dificultate al procedurii de actualizare a datelor în timp real. De asemenea, alegerea de a nu implementa pattern-ul de Repository nu are consecințe negative asupra performanței, a lățimii de bandă utilizate de c ă tre aplicație sau a cantității de date transmise între aplicație și baza de date, deoarece API-ul expus de Firebase utilizează caching local, astfel interogările multiple asupra aceleiași resurse vor rezulta într -o singură interogare la baza de date, la primul apel, și citiri din cache la următoarele, în condițiile în care resursa în cauză nu se modifică.

Figura 5 – Structura bazei de date

În cadrul D.A.L. se setează Listeneri asupra entităților din baza de date, Listeneri ce vor fi apelați în momentul modificării resursei respective, asigurând astfel sincronizarea în timp real a datelor. În acest sens s-a utilizat Language Level 8, Listener reprezentând o interfață funcțională ce poate fi ușor înlocuită cu o expresie lambda, ceea ce a crescut lizibilitatea codului.

### 2.1.3 Firebase Cloud Functions

În cazul funcționalităților a căror implementare în aplicația Android ar fi întâmpinat obstacole, s-a utilizat Firebase Cloud Functions. Acestea reprezintă funcții JavaScript cărora li s-a făcut deploy în Cloud și sunt disponibile fie drept endpoint- uri accesibile prin HTTPS, fie drept triggere la scriere pe baza de date. Principalele funcționalități ce folosesc Firebase Cloud Functions sunt transmiterea de e-mailuri ce conțin tokenul de înregistrare al profesorului sau administratorului și transmiterea de notificări, ce a fost realizată prin intermediul triggerelor.



Figura 6 – Exemplu de functie din cloud, disponibila prin HTTPS, ce trimite tokenul de inregistrare prin email

## 2.2 Tehnologii utilizate

În elaborarea lucrării de licență am utilizat tehnologii moderne, de actualitate, ce au adus valoare proiectului, sporind gradul de utilitate a acestuia și facilitând procesul de dezvoltare prin funcționalitățile oferite de acestea. În continuare voi enumera principalele tehnologii utilizate si motivul alegerii lor:

* ***Android SDK 17+***
* ***Java JDK 1.7 împreună cu Language Level 8***
* ***Firebase Authentication***
* ***Firebase Realtime Database***
* ***Firebase Cloud Functions***

### 2.2.1 Android SDK 17+

Punând în balanță alegerea unei versiuni de Android SDK mai noi, ce aduce funcționalități ce facilitează procesul de dezvoltare și asigurarea unui grad cât mai mare de compatibilitate cu dispozitivele prezente pe piață, am ales să utilizez SDK 17+, acesta îmbinând cele două criterii menționate. Pentru detalii vezi anexa 1.

### 2.2.2 Java JDK 1.7 împreună cu Language Level 8

Din motivul elaborat anterior, și anume acela de a păstra compatibilitatea cu cât mai multe dispozitive Android, am ales să implementez aplicația cu ajutorul versiunii 1.7 de JDK, împreună cu Language Level 8. Pentru detalii vezi anexa 2.

### 2.2.3 Firebase Authentication

Dorind a elabora o aplicație server-less, pentru gestionarea conturilor, a utilizatorilor și a procesului de login, de înregistrare și de resetare a parolei, am optat pentru integrarea API-ului de Firebase în cadrul aplicației.

Acesta asigură confidențialitatea datelor conturilor de utilizator, unicitatea și validarea e-mailurilor, procesul de resetare a parolei prin e-mail, toate acestea reducând gradul de complexitate a aplicației în aceste funcționalități.

### 2.2.4 Firebase Realtime Database

După lungi considerente, din dorința transmiterii datelor în timp real, menținând toți utilizatorii conectați în permanentă și optând pentru o aplicație server-less, am ales ca bază de date Firebase

Realtime Database. Aceasta este o bază de date NoSQL, datele fiind memorate sub forma unui JSON, oferind posibilitatea de a actualiza datele aproape instant pe toate dispozitivele conectate la aceasta.

O altă motivație în această alegere sunt regulile de securitate expuse de către Firebase Realtime Database, putând seta reguli de citire și scriere asupra fiecărui nod din arborele JSON, asigurând astfel confidențialitatea și integritatea datelor aplicației.

### 2.2.5 Firebase Cloud Functions

Un alt produs folosit din cadrul pachetului oferit de către Firebase este Firebase Cloud Functions, tehnologie aleasă din necesitatea setării de triggere asupra bazei de date, triggere ce nu pot fi create altfel și care contribuie la implementarea uneia dintre principalele funcționalități ale aplicației, și anume a sistemului de notificări.

De asemenea, în cadrul elaborării acestei lucrări s-au utilizat și următoarele instrumente:

* ***Android Studio 3.5.1***
* ***Android Emulator 29.3.4 cu Android SDK 24***
* ***Visual Studio Code 1.36.1***
* ***Postman***
* ***Git Bash***

## 2.3 Scenarii de utilizare a aplicației

În acest capitol voi etala funcționalitățile aplicației, modul lor de folosire și interacțiunea utilizatorului cu aplicația.

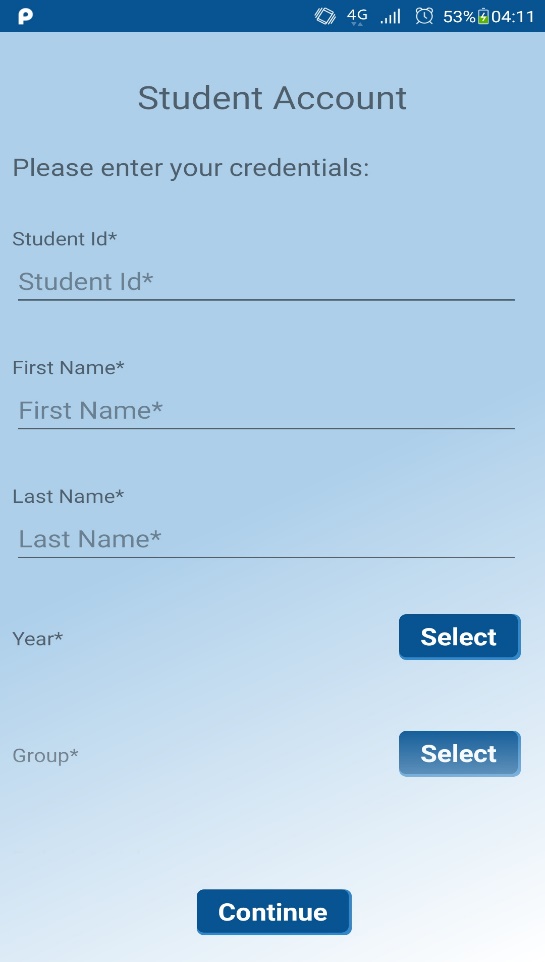
### Înregistrarea în aplicație

La deschiderea aplicației, utilizatorul este întâmpinat de o pagină de login, iar pentru a continua este necesar ca el să se autentifice. Pe această pagină el va avea acces la trei butoane: „Login”, „Sign up” și „Forgot password”.

Apăsarea butonului de Sign up îl va redirecționa către o pagină în care utilizatorul va selecta tipul de cont pe care dorește să îl creeze, având opțiunile „Student”, „Profesor” și „Administrator”.

#### Înregistrarea ca și student

În cazul în care utilizatorul dorește să își creeze un cont de student, el va trebui să completeze un formular de creare a profilului, unde va trebui să introducă date obligatorii precum numărul matricol, ce va rămâne confidențial, numele, prenumele, anul și grupa. De asemenea, el va putea introduce și date opționale precum e-mailul, pagina sa web, etc (figura 7).

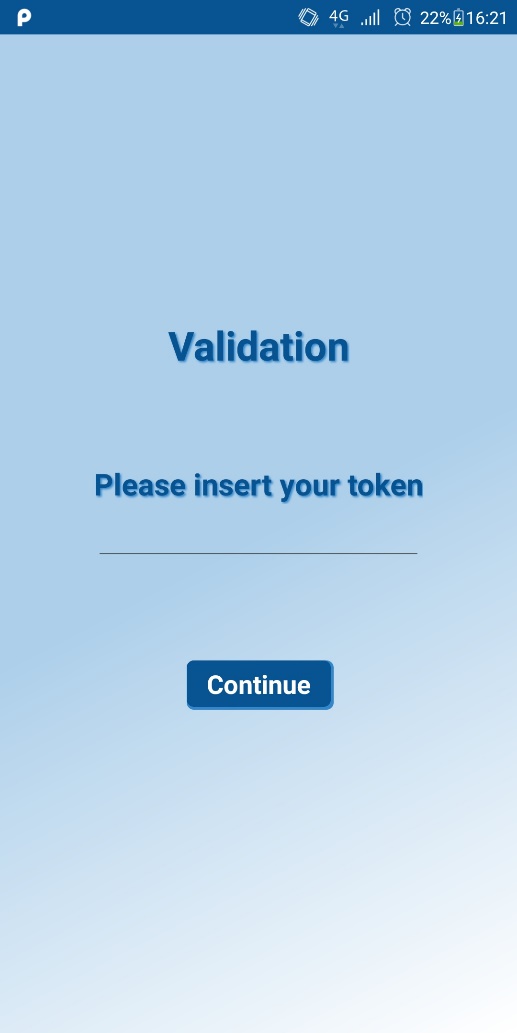
 În urma completării formularului de profil, el va fi redirecționat către pagina de creare a contului, unde își va introduce e-mailul asociat contului (care poate fi diferit de e-mailul introdus în secțiunea de profil) și o parolă. În cazul există deja un cont cu numărul matricol introdus anterior sau cu e-mailul introdus la acest pas, utilizatorul nu își va putea crea cont.

Dacă procesul de creare a contului s-a terminat cu succes, utilizatorul va fi redirecționat înapoi către pagina de login.

Figura 7 – Pagina de creare a profilului studentului

#### Înregistrarea ca și profesor sau administrator

La selectarea opțiunii de a crea un cont de profesor sau de administrator, utilizatorul va fi redirectat către o pagină unde i se va cere introducerea unui cod de înregistrare, cod transmis de către un administrator existent al aplicației prin e-mail (figura 8).

Acest token de înregistrare este format din 6 caractere alfanumerice, este utilizabil o singură dată și expiră în 24 de ore de la generare. De asemenea, un token generat cu scopul creării unui cont de profesor nu poate fi folosit pentru a crea un cont de administrator și vice- versa.

După introducerea unui token valid, utilizatorul va fi redirectat către o pagină de creare a profilului, unde profesorul își va introduce numele, prenumele, gradul, e-mailul, etc, iar administratorul numere, prenumele și e-mailul. În urmă validării datelor, utilizatorul va fi redirecționat către pagina de creare a contului, ca și în cazul studentului.

Figura 8 – Pagina de verificare a tokenului de inregistrare

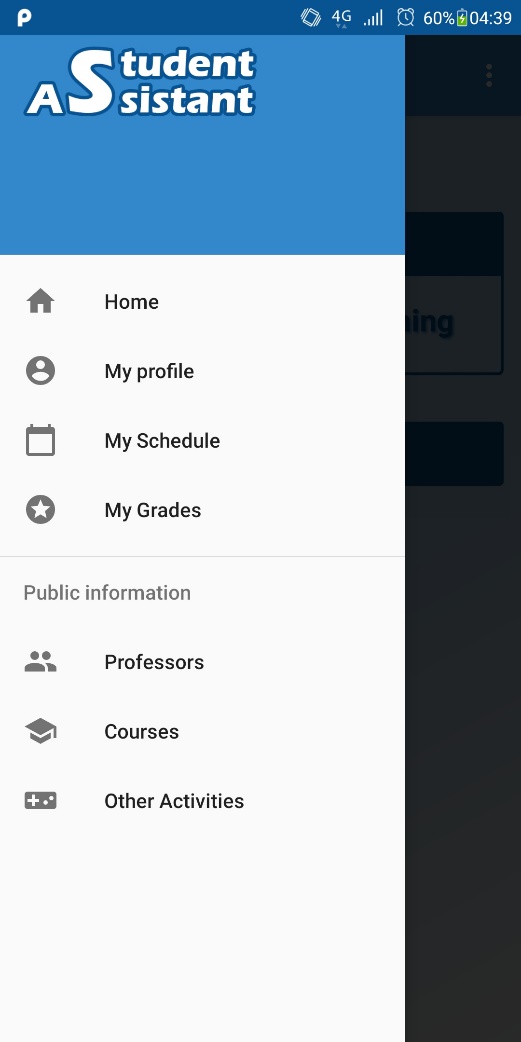
#### Resetarea parolei

Apăsând pe butonul „Forgot password”, utilizatorului i se va cere să își introducă adresa de e-mail utilizată la crearea contului, iar dacă aceasta este validă, i se va trimite un e-mail cu ajutorul unei funcții Cloud și a serverului SMTP de Gmail, e-mail ce conține un link către o pagină de resetare a parolei.

### Utilizarea aplicației ca și student

Odată logat, studentului i se expune o pagină de start unde își poate vedea următoarea ora de clasa și timpul rămas până la începerea acesteia, și următorul examen și timpul rămas până la începerea acestuia.

Deschizând meniul aplicației, el va avea acces la următoarele pagini: Home, My Profile, My Schedule, My Grades, Professors, Courses și Other Activities (figura 9).

**Pagina My Profile** – pe această pagină studentul își va putea vedea profilul creat în pasul de creare a contului și își va putea edita datele, cu excepția numărului matricol și a numelui. De asemenea, studentul își va putea vedea cursurile și activitățile la care este înscris și va putea să adauge o imagine de profil sau să o elimine, în cazul în care a făcut deja asta. De asemenea, studentul se poate retrage din cadrul unui curs sau a unei activități, procedura de înscriere a acestuia la o disciplină fiind descrisă în următoarele secțiuni.

**Pagina My Schedule** – pe această pagină studentul își va putea vedea orarul, acesta fiind împărțit în două categorii: „Regular” și „Special”. Orarul „Regular” conține orele de clasa săptămânale, având liste expandabile pentru

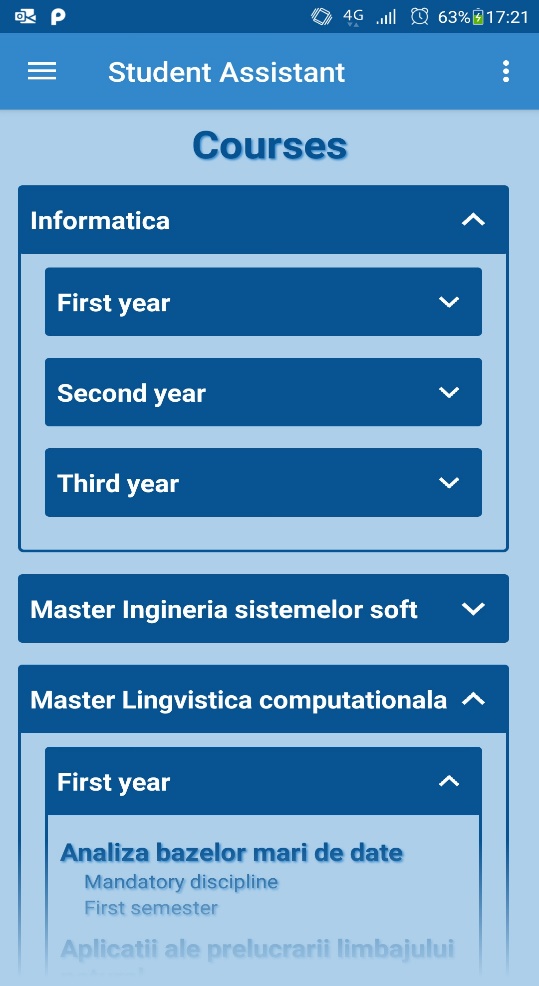
Figura 9 – Meniul aplicatiei din perspectiva studentului

fiecare zi a săptămânii, iar orarul „Special” conține orele ce au loc o singură dată, precum examenele, ele fiind grupate după dată la care au loc. La adăugarea unei noi ore în orar, studenții înscriși la disciplina sau activitatea respectivă vor fi notificați în timp real, datele lor fiind actualizate aproape instantaneu.

**Pagina My Grades** – oferă studentului posibilitatea de a-și vedea notele, acestea fiind grupate după tipul de notă, acestea putând fi „Laborator”, „Examen”, „Examen parțial”, „Proiecte”, „ Restanță ”, „ Restanță parțial”, „ Restanță proiecte”. Aici va putea vedea notele luate la toate materiile și data la care acestea au fost acordate. De asemenea, în momentul în care un profesor adaugă o notă unui student, acesta va fi notificat iar această secțiune va fi actualizată în timp real.

Secțiunile de Professors, Courses și Other Activities sunt secțiuni globale, acestea putând fi accesate de toți utilizatorii, însă ele vor oferi funcționalități diferite în funcție de permisiunile fiecărui utilizator.

**Secțiunea Professors** – îi permite studentului să vadă toți profesorii înscriși în aplicație, putând accesa profilele lor, cu scopul de a vedea fișierele partajate de ei, cursurile pe care le predau, activitățile pe care le coordonează și orarul lor.

** Secțiunea Courses (figura 10)** – permite studentului să vadă o listă cu toate cursurile, acestea fiind grupate după specializări și anii de studiu, putând accesa pagina fiecărui curs.

**Pagina cursului** – oferă studentului informații despre curs, precum și o scurtă descrierea cursului, orarul cursului și profesorii ce predau cursul respectiv. Pentru a vedea mai multe informații referitoare la curs, unui student care nu este înscris la cursul respectiv i se va afișa mesajul „You are not enrolled in this course”, urmat de un buton pentru trimiterea unei cereri de înscriere. În momentul trimiterii cererii de înscriere, profesorii cursului vor fi notificați, aceștia putând acceptă sau anula cererea. Dacă cererea de înscriere a fost acceptată, studentul va avea acces la laboratoarele cursului, la fișierele partajate din cadrul cursului, iar profesorii ii vor putea pune note.

Figura 10 – Sectiunea Courses din perspectiva studentului

**Pagina laboratorului** – De asemenea, de pe pagina de curs este posibilă vizualizarea laboratoarelor de către studenții înscriși la cursul respectiv, pe această pagină fiind afișată o scurtă descriere a laboratorului, fișierele referitoare la acesta, fișiere precum tutoriale folositoare pentru laborator, materiale didactice și fișe cu exerciții, studentul având posibilitatea de a descarcă aceste fișiere. Tot pe pagina laboratorului, studentul își poate vedea nota primită la acest laborator, împreună cu data la care ea a fost acordată.

**Secțiunea Other Activities** – această secțiune oferă studentului posibilitatea vizualizării unei liste cu toate activitățile înregistrate în aplicație. Acestea reprezintă activități ce nu pot fi încadrate la categoria de disciplină, ele neavând note și nefiind structurate precum o disciplină. Exemple de activități sunt: Consultații, Îndrumare pentru licență, Cerc Științific, Matematică pentru concursuri, etc. Din această secțiune se poate naviga pe pagina fiecărei activități individuale, comportamentul acesteia fiind similar cu cel al paginii de curs, lipsind însă notele și laboratoarele.

### Utilizarea aplicației ca și profesor

Similar studentului, profesorul va dispune de un meniu principal ce va cuprinde toate paginile și secțiunile enumerate anterior, cu excepția paginii de My Grades.

**Pagina Home** – este similară că paginii Home a studentului, profesorului arătându-i-se următoarea oră de clasă și următorul examen, și în cât timp acestea vor începe.

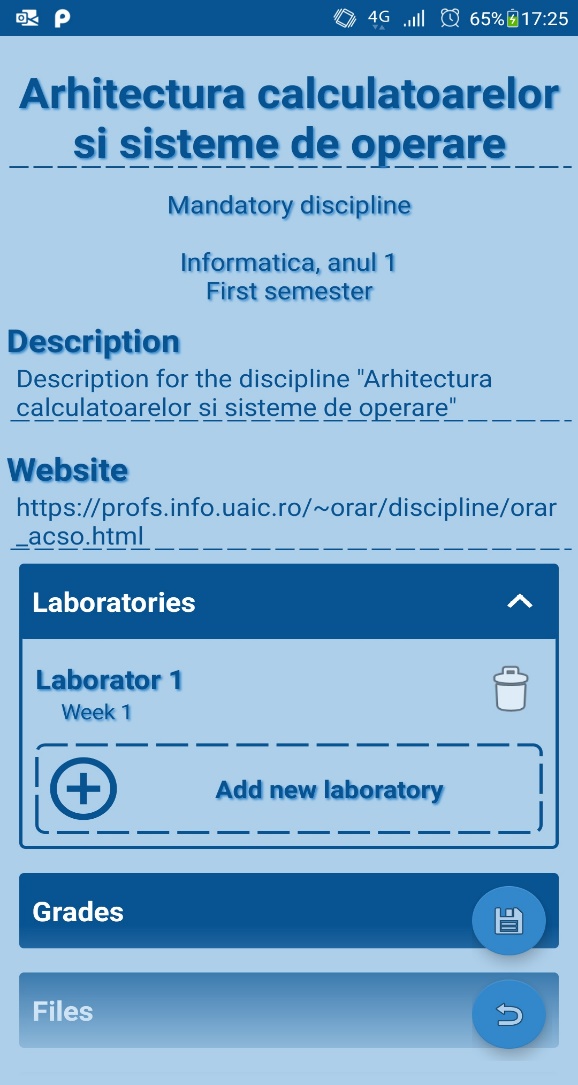
**Pagina My Profile** – prin intermediul acestei pagini profesorul își poate vizualiza profilul, având posibilitatea de a-și edita datele personale, de a adaugă o imagine de profil și fișierele ce dorește să le partajeze cu ceilalți participanți ai aplicației. De asemenea, aici profesorul își poate

vedea cursurile și activitățile sale, având posibilitatea de a se retrage de la una din acestea.

**Pagina My schedule** – aceasta oferă posibilitatea profesorului de a- și vizualiza orarul, atât cel normal cât și cel special, această pagină neputând fi editată. Pentru ștergerea unei ore din orar, profesorul va trebui să acceseze pagina cursului sau activității respective.

**Secțiunea Professors** – similar studentului, profesorului i se afișează o lista cu toți profesorii înscriși în aplicație, putând accesa paginile acestora, vizualizând informațiile oferite de către profesorii participanți. Această pagină reflectă datele introduse pe pagină de profil a profesorilor respectivi, astfel în momentul în care un profesor își editează profilul, pagina profesorului respectiv se va actualiza în timp real tuturor celorlalți utilizatori.

**Secțiunea Courses –** această secțiune oferă profesorului posibilitatea vizualizării cursurilor grupate pe ciclu de studiu, pe specializare și pe ani, profesorul putând crea un nou curs, el fiind înscris automat la disciplina respectivă în momentul în care aceasta este creată.

**Pagina cursului (figura 11)** – pe pagina cursului, un profesor ce nu este titular la cursul respectiv va putea vedea doar datele publice, și anume descrierea cursului, orarul și profesorii participanți. Spre deosebire de student, acesta nu poate trimite o cerere de înscriere la curs, responsabilitatea de a adăuga profesorii participanți revenind profesorului ce crează cursul.

În cazul în care profesorul este înscris la cursul respectiv, acesta va putea vedea, descarcă și șterge fișierele partajate cursului, dar va putea încărca și el fișiere la rândul său. De asemenea profesorul va putea șterge cursul, acțiune ce va șterge și toate datele referitoare la cursul respectiv (laboratoare, note, orarul cursului, fișierele cursului, fișierele laboratorului, etc), va putea crea și șterge laboratoare, va putea adăuga ore noi în orar, atât în cel normal cât și cel special, va putea adăuga alți profesori la curs, elimina studenți din curs, având posibilitatea atât de a răspunde cererilor de înscriere a studenților cât și de a-i adăuga manual.

Figura 11 – Pagina unui curs din perspectiva profesorului titular

O altă funcționalitate oferită profesorului înscris la curs pe această pagină este aceea de a adaugă note studenților, pe pagină cursului putând fi adaugate note la examene, examene parțiale, proiecte, restanțe, etc, note ce vor referenția studentul prin intermediul numărului său matricol. Dacă în momentul adăugării unei note la un număr matricol, studentul respectiv nu este înscris la curs, acesta va fi înscris automat, iar dacă studentul nu are cont în aplicație, în momentul în care acesta își va crea cont, notele îi vor fi deja atribuite.

**Pagina laboratorului** – de pe pagină cursului profesorul poate naviga către pagină fiecărui laborator, aici oferindu-i-se posibilitatea de a partaja și șterge fișierele relevante laboratorului, și de a adaugă note studenților pentru laboratorul respectiv, procesul de adăugare a notelor la laborator desfășurându-se similar cu cel de adăugare a notelor la curs, descris anterior.

**Secțiunea Other activities** – în această secțiune profesorul va putea vizualiza toate activitățile înregistrate în aplicație, având posibilitatea de a crea o nouă activitate, procedeu similar celui de a crea un nou curs.

**Pagina activității** – pe această pagină profesorului i se oferă aceleași funcționalități că și pe pagină cursului, menționând faptul că o activitate nu va avea laboratoare și nici note, profesorul titular putând doar edita lista de fișiere, studenți, profesori și orarul acestei activități, schimbări ce vor fi transmise în timp real celorlalți utilizatori, cei înscriși în cadrul activității primind și o notificare.

### Utilizarea aplicației ca și administrator

Administratorul reprezintă un super-user al aplicației, acesta având drepturi de titular asupra oricărei entități descrise în aceasta, drepturi ce vor fi folosite doar în cazul în care situația o dictează, aceste situații urmând a fi descrise în următoarele secțiuni.

**Pagina My profile** – similar și celorlalți utilizatori, administratorul își va putea edita profilul pe această pagină, însă el nu poate fi înregistrat la cursuri sau activități.

**Pagina Registration (figura 12)** – precum am menționat în descrierea acțiunii de înregistrare a profesorilor și a administratorilor, aceștia au nevoie de un token de înregistrare pentru a-și crea cont. Acest token va fi generat pe această pagină de către administrator.

Pagina Registration oferă posibilitatea administratorului de a selecta ce tip de token de înregistrare dorește să creeze, având două butoane radio, unul pentru profesor și unul pentru administrator, și un input text unde va scrie e-mailul la care dorește acest token să fie trimis. În urmă acestei acțiuni, un token de 6 caractere află -numerice va fi generat, token valabil 24 de ore și utilizabil o singură dată. Un token de administrator nu va putea fi folosit cu scopul creării unui cont de profesor și viceversa.

Este de menționat că această procedura este complet securizată: tokenul nu este vizibil administratorului, el nu va fi memorat în baza de date (întrucât, acesta fiind necesar în procesul de înregistrare, nu poate avea reguli de securitate în cadrul bazei de date Firebase), ci doar hash-ul MD5 al acestuia va fi memorat, eliminând posibilitatea compromiterii sale.

Figura 12 – Pagina Registration, disponibila exclusiv administratorilor

Având 6 caractere alfanumerice, atât uppercase cât și lowercase, posibilitatea unei persoane cu intenții malițioase de a „ghici” acest token fiind una infimă, existând peste 56 de miliarde de posibile valori ale tokenului, iar „spargerea” sa în 24 de ore este aproape imposibilă.

**Secțiunile Courses și Other activities** – similar profesorului, administratorului i se acordă posibilitatea de a crea cursuri și activități noi, însă el nu va fi înscris la acestea în momentul creării lor.

**Paginile cursului și activității** – având rolul de super-user, un administrator edita orice informație a oricărui curs și a oricărei activități, permisiune ce poate fi de ajutor în cazul în care un curs sau o activitate nu mai are niciun profesor, astfel niciun profesor și niciun student nu se mai poate înscrie la acea disciplină sau în cazul în care singurul profesor asignat unei activități sau unui curs nu mai are acces la contul sau (nu mai are acces la contul de e-mail cu ajutorul căruia și-a făcut cont în aplicație, neputând astfel să își reseteze parola).

# Concluziile lucrării

Lucrarea a atins în mare parte toate punctele dorite. În acest moment aplicația oferă soluții pentru toate problemele prezentate în Capitolul 1: Descrierea problemei.

Consider că, aplicația Student Assistant poate fi cu adevărat folositoare atât pentru studenții cât și pentru profesorii care o folosesc, aceștia având un mijloc de centralizarea a datelor relevante lor, găsind toate informațiile pe care le caută într -un singur loc, actualizate în timp real.

Reușita de a oferi utilizatorilor un mediu conectat și actualizat în timp real în care pot căuta informațiile pe care doresc să le afle, este un plus mare pe care îl aduc, astfel încât în ziua de azi rapiditatea transmiterii informațiilor contează uneori mai mult pentru utilizatori decât funcționalitățile pe care le oferă acestea.

Un alt plus pe care îl aduc este modul simplu de utilizare al aplicației, astfel încât interfață disponibilă este intuitivă, existând un număr destul de mic de pași de urmat pentru aproape orice acțiune pe care un utilizator o poate face în aplicație. Pentru a fi și mai clar în înțelegerea scenariilor de către utilizatori, am venit și cu mesaje specifice la anumiți pași din aplicație pentru a elimina orice grad de confuzie.

În concluzie pot spune că Student Assistant este un succes pentru mine, deoarece mi-a oferit ocazia să învăț o multitudine de tehnologii noi, de actualitate și folositoare de care sunt sigur că îmi vor fi de ajutor pe viitor.

#### Potențiale direcții de viitor ale aplicației

Student Assistant este un proiect ce poate evolua mult mai mult. Idei pe care le-am pentru a aduce valoare mai mare aplicației:

* Aducerea în aplicația a unei pagini ce îndrumă studentul în alegerea cursurilor opționale, îndrumare bazată pe  un algoritm de Machine Learning ce va avea drept date de antrenament notele studenților din anii precedenți.
* Implementarea unui sistem de anunțuri multicast și single cast, pentru a se putea transmite mult mai ușor informații către anumiți studenți sau către categorii de studenți.
* Posibilitatea adăugării restanțelor în aplicație și a criteriilor de promovare, lucru care ar putea îndrumă studenții către ce materii trebuie să își îndrepte atenția, trăgând un semnal de alarmă în momentul în care aceștia nu au trecut pragul de promovare.
* Adăugarea unui sistem de Crash Reporting & Analytics pentru a genera statistici referitoare la utilizarea aplicației și a o îmbunătăți prin analize de user experience.
* Implementarea unui sistem de logging centralizat, pentru a putea monitoriza evenimentele semnificative ce au loc în aplicație.

# Bibliografie

<https://proandroiddev.com/reducing-boilerplate-in-gradle-multi-module-projects-2ff2dde5bf95>

<https://softauthor.com/firebase-querying-sorting-filtering-database-with-nodejs-client>

<https://developer.android.com/reference/android/os/AsyncTask>

<https://stackoverflow.com/questions/39024702/firebase-querying-data>

<https://vladmihalcea.com/a-beginners-guide-to-java-time-zone-handling/>

<https://gist.github.com/codediodeio/6dbce1305b9556c2136492522e2100f6>

<https://stackoverflow.com/questions/38687594/firebase-rule-that-works-like-a-filter>

<https://firebase.google.com/docs/functions/database-events>

<https://untitledcodingblog.blogspot.com/2019/12/a-list-of-commonly-used-roles-and.html?fbclid=IwAR0maoPmG6QuWVRw5Zh1sNrFrR5qyCTe0BDqaWqTCbp1CgidN4Fajlp19-4>

<https://medium.com/@ali.muzaffar/android-developers-are-you-a-macgyver-or-macgruber-with-custom-views-a155f24b7479>

<https://developer.android.com/topic/performance/vitals/render#common-jank>

<https://developer.android.com/training/improving-layouts/optimizing-layout.html>

<https://medium.com/@ubuntudroid/using-string-placeholders-with-data-binding-394615354464>

<https://firebase.google.com/docs/cloud-messaging/concept-options.html#notifications_and_data_messages>

<https://firebase.google.com/docs/cloud-messaging/send-message>

<https://stackoverflow.com/questions/38549329/setting-message-text-appearance-in-alert-dialog-using-xml-android-api-level-23>

<https://www.androidauthority.com/android-push-notifications-with-firebase-cloud-messaging-925075/>

<https://firebase.google.com/docs/functions/get-started?authuser=0>

<https://stackoverflow.com/questions/44547676/send-push-notifications-using-cloud-functions-for-firebase>

[https://stackoverflow.com/questions/4345032/how-to-send-a-simple-e-mail-programmatically-exists-a-simple-way-to-do-it](https://stackoverflow.com/questions/4345032/how-to-send-a-simple-email-programmatically-exists-a-simple-way-to-do-it)

<https://medium.com/@edigleyssonsilva/cloud-functions-for-firebase-sending-e-mail-1f2631d1022e>

<https://firebase.google.com/docs/functions/callable>

<https://mobikul.com/define-custom-attribute-custom-view/>

<https://stackoverflow.com/questions/29035384/android-shape-circle-with-crossplus>

<https://medium.com/@elia.maracani/android-data-binding-passing-a-variable-to-an-include-d-layout-3567099b58f>

<https://android.jlelse.eu/android-architecture-communication-between-viewmodel-and-view-ce14805d72bf>

<https://medium.com/upday-devs/android-architecture-patterns-part-3-model-view-viewmodel-e7eeee76b73b>

<https://firebase.google.com/docs/auth/admin/custom-claims#access_custom_claims_on_the_client>

<https://firebase.google.com/docs/database/android/offline-capabilities>

<https://firebase.google.com/docs/database/web/structure-data>

# Anexe

## Anexa 1

Android SDK 17+ este unul dintre cele mai utilizate SDK-uri din rândul dezvoltatorilor, datorită gradului de compatibilitate cu dispozitivele ce se pot găsi deja pe piața, asigurând, conform statisticilor Google o compatibilitate de peste 70%.

În același timp SDK 17+ asigură componente importante precum RadioGroup, ConstraintLayout, o gamă variată de obiecte de tip Listener și multe alte elemente de UI. Fiind o versiune relativ veche de Android, acesta folosește, din păcate Java 7 cu Language Level 8.

## Anexa 2

Acest nivel de limbaj asigura posibilitatea utilizării de lambda expressions, definirea interfețelor funcționale proprii, interfețe precum Function<T,V>, Predicate<T>, Consumer<T>, interfețe ce au fost implementate în cadrul aplicației, întrucât acestea facilitează procesul de development, oferind o paletă mai largă de situații în care se pot utiliza Lambda Expressions. Utilizarea acestora îmbunătățește drastic gradul de lizibilitate a codului, multe obiecte disponibile in SDK 17+, în special cele de tip Listener, putând fi înlocuite prin lambda expressions, reducând numărul de clase și clase anonime ale aplicației.