Grades

- Arhitectura aplicatiei -

Ovidiu Pricop, Adrian Butnaru, Ciuc Tiberiu, Dorneanu Cristian - FII UAIC - Aprilie 2016

Cuprins

Context

Aplicatia Grades presupune un numar mare de componente functionale pentru a asigura succesul in productie. Este preferabil ca arhitectura sa fie conceputa pe mai multe module. Acest document are ca scop descrierea arhitecturii modulare a aplicatiei Grades, urmarind acoperirea cat mai exhaustiva a modului in care va functiona fiecare unitate a aplicatiei.

In prima parte a documentului se va introduce ideea pe care se bazeaza aplicatia Grades. Documentul va continua cu o serie de diagrame care sa prezinte componentele necesare pentru a asigura acoperirea nevoilor sistemului academic in ceea ce priveste notarea studentilor. In final, se va descrie arhitectura fiecarei componente critice pe care se bazeaza aplicatia.

Introducere

Motivatie

În prezent, nu există o platformă rezervată facultății care să centralizeze notele puse de către profesori la evaluările de pe parcursul semestrului și să ofere studenților o modalitate de a le vizualiza. In acest context, se dorește înștiințarea studenților în legătură cu situația academica personală. Se urmareste evitarea situațiilor în care studentul trebuie să viziteze numeroase site-uri ale materiilor în vederea vizualizarii notelor. Informațiile legate de notare se vor transfera în mod eficient în cadrul aplicației.

Pe scurt, aplicatia "Grades" propune înlăturarea problemelor din sistemul actual de notare prin punerea la dispozitie a unei platforme web.

Actori

Aplicatia Grades este destinata mediului academic. Astfel, principalii actori vor fi profesorii, studentii si personalul administrativ (decan, secretar, etc.). In continuare, vom trece in revista ce va reprezenta aplicatia pentru fiecare actor critic in sistemul de evaluare.

Studenti

Studentii vor beneficia de aplicatia Grades prin faptul ca isi vor putea urmari situatia evaluarilor intr-un singur loc, evitand situatiile in care ar trebui sa urmareasca mai multe site-uri simultan pentru a-si afla rezultatele de pe parcursul semestrului. De asemenea, studentii vor putea alege sa fie instiintati atunci cand li se va pune o nota, economisind timp prin evitarea verificarii periodice a site-urilor profesorilor pentru a afla daca s-au afisat sau nu rezultatele urmarite.

Initial, pentru a accesa aplicatia, studentul va trebui sa se conecteze utilizand un cont personal (Google, Facebook, etc.), iar, apoi, sa isi confirme identitatea in cadrul facultatii (mai multe detalii la sectiunea detaliata a arhitecturii autentificarii). Odata autentificat, studentul va putea sa isi vizualizeze situatia personala la materiile la care este inscris (promovate sau nepromovate), aplicatia permitand utilizatorului sa isi vizualizeze si istoricul.

Studentul va avea:

- Abilitatea de a alege daca primeste sau nu notificari prin email
- Posibilitatea de a solicita un transfer de la o grupa la alta
- Posibilitatea de a solicita reevaluarea unei lucrari

Profesori

Fiecare profesor va fi asociat, in prealabil, de catre un administrator, cu o serie de materii si o serie de grupe. De asemenea, fiecare profesor va avea un rol, care va stabili drepturile sale in cadrul evaluarii materiei. Printre rolurile comune, se numara cel de titular al materiei, profesor la curs si profesor la seminar/laborator. De exemplu, profesorul titular va avea dreptul si obligatia de a seta modalitatea de evaluare a materiilor pentru care este responsabil. Profesorul va putea avea, suplimentar (nu si mutual exclusiv) alte roluri, precum cel de profesor la seminar. In mod similar, un profesor de laborator va putea fi si profesor la curs, fara a fi titular.

Profesorul va putea pune note studentilor de la grupele pentru care este responsabil si va putea edita notele deja puse (in caz ca doreste sa repete o evaluare, sau in caz ca o nota s-a modificat in urma unei contestatii sau mariri). De asemenea, profesorul va putea, in functie de rol, sa accepte sau sa refuze cereri de transfer si sa fie instiintat despre studentii care doresc sa fie reevaluati. Profesorul va putea opta si pentru notificari prin email de anumite evenimente de interes.

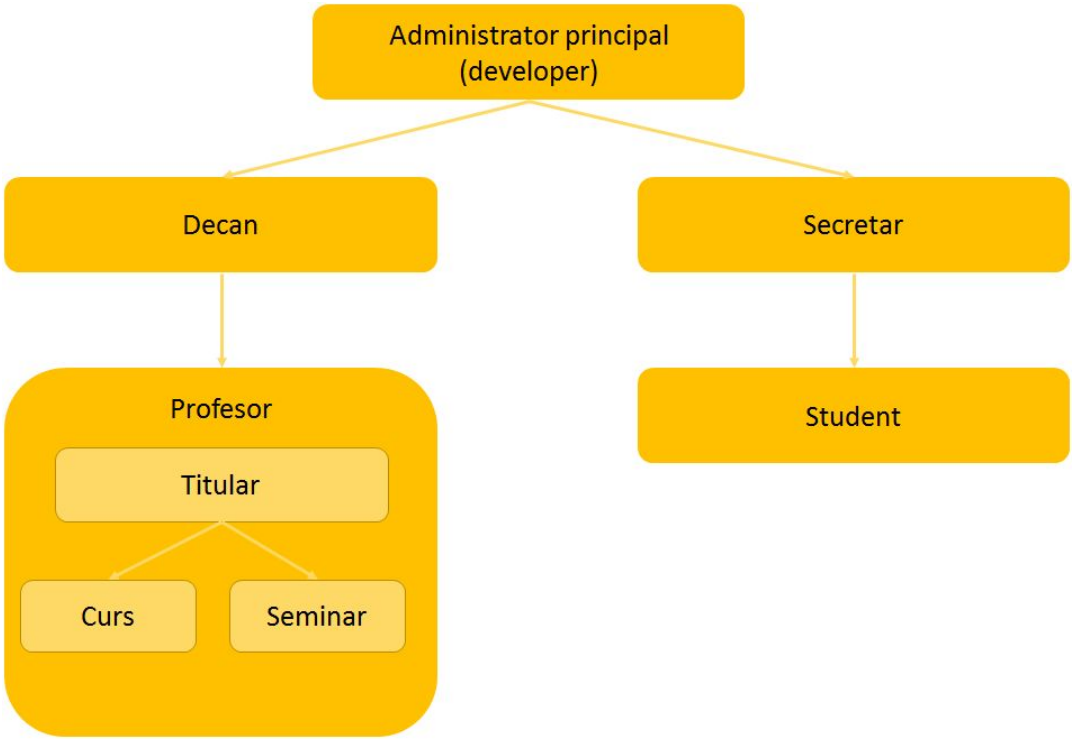
Administratori

Administratorii se vor ocupa de gestionarea profesorilor si studentilor. Administratorii vor avea asociat un rol care le va da anumite drepturi in cadrul aplicatiei. Unul dintre rolurile de baza ale unui administrator va fi cel de a adauga personal in aplicatie. Personalul poate fi reprezentat atat de studenti, cat si de profesori sau alti administratori.

Structura initiala ierarhica, bazata pe roluri si permisiuni, va fi stabilita tot de catre un administrator. Astfel, un administrator va putea adauga, actualiza sau sterge studenti sau profesori, alti administratori si va putea stabili in ce an, grupa, etc. este fiecare student si la ce materii preda un profesor (inclusiv rolul de titular, profesor la laborator, etc.).

Administratorii vor putea vizualiza situatia notelor studentilor din toti anii, in functie de drepturi (de exemplu, rolul de secretar sau decan).

Ierarhie actori



Diagrame use-case

Student	Profesor	Administrator	Configurator (Developer)

Arhitectura componentelor

Diagrama componente - Privire de ansamblu

Sistem login	Student	Profesor	Evaluare	Administrare
TODO	TODO	TODO	TODO	TODO

Modelarea bazei de date

Context

Aplicatia Grades lucreaza cu date ce au caracter persistent (date studenti, profesori, permisiuni, situatii evaluari, modalitati evaluari, etc.). Pentru a stoca aceste date, se va utiliza o baza de date modelata special pentru a acomoda cerintele aplicatiei. Baza de date va fi realizata utilizand MongoDB si va cuprinde urmatoarele entitati:

- Utilizatori
- Inregistrari
- Studenti
- Profesori
- Permisiuni
- Module (evaluare)
- Note
- Cursuri

In continuare, vom realiza o scurta prezentare a fiecarei entitati enumerate mai sus.

Diagrama ERD

https://drive.google.com/file/d/0BwA4LZ-hnPjYSkItRUVhZVB0WjA/view?usp=sharing

Colectii

Utilizatori

Indiferent de rolul pe care il are in aplicatie (profesor, student, administrator, etc.), o persoana care foloseste aplicatia este, inainte de toate, un utilizator. Din motive de securitate, fiecare utilizator trebuie sa fie inregistrat si autentificat, inainte sa poata primi drepturi pentru a accesa modulele dorite.

Un utilizator nou creat va avea doar campul “user” completat, iar campul “identityConfirmed” va fi fals. Utilizatorul va primi pe adresa furnizata in procesul de logare un email de confirmare ce va contine un cod, in cazul in care emailul este prezent in colectie “Inregistrare” (adica rolurile i-au fost deja configurate). Prin introducerea codului secret primit pe adresa de email ce ii confirma identitatea in facultate, contul utilizatorului va deveni confirmat si i se vor asocia o serie de roluri (permisiuni).

Colectia “Utilizatori” este definita astfel:

"Users"

{

{ user: { $type: "string"} }, -- adresa de email pe care utilizatorul o foloseste la logare

{ facultyIdentity: { $type: "string"} }, -- identitatea utilizatorului in cadrul facultatii, adica o adresa de email pe care facultatea si utilizatorul au stabilit-o (de exemplu, webmail); utilizata pentru a asocia un set de roluri unui cont creat si confirmat

{ apiKey: { $type: "string" } }, -- odata autentificat, utilizatorului i se va trimite o cheie asociata cu contul sau, care ii va permite serverului sa autorizeze fiecare cerere venita din partea utilizatorului

{ keyExpires: { $type: "date" } }, -- cheia asociata contului utilizatorului este valabila, din motive de securitate, o perioada scurta de timp (de exemplu, o ora)

{ identityConfirmed: { $type: "bool" } }, -- contul va fi confirmat doar in cazul in care utilizatorul isi va confirma identitatea in cadrul aplicatiei; confirmarea identitatii se va face printr-un mecanism “2-way authentication”, descris, in acest document, la sectiunea “Autentificare”

}

Restrictii

Unique	Required	Index
user	user	user

Inregistrare

Fiecare utilizator va trebui sa isi creeze un cont. Identitatea utilizatorului trebuie verificata pentru a fi siguri ca ea concide cu cea inregistrata la facultate. Este necesar un sistem de asociere a identitatii de la facultate cu un rol care sa ofere permisiuni utilizatorului cu acea identitate. Verificarea presupune, doar la crearea unui cont nou, o metoda de tip “2-way-authentication” care verifica identitatea utilizatorului prin intermediul unui cont de email inregistrat de facultate, unde va fi trimisa un cod de confirmare. Codul de confirmare va trebui introdus de catre utilizator pentru a finaliza procedura de inregistrare.

Colectia “Inregistrari” este definita astfel:

"Registrations"

{

{ facultyIdentity: { $type: "string"} }, -- identitatea utilizatorului in cadrul facultatii, adica o adresa de email pe care facultatea si utilizatorul au stabilit-o (de exemplu, webmail); utilizata pentru a asocia un set de roluri unui cont creat si confirmat

{ roles: [{ $type: "string" }] }, -- rolurile pe care utilizatorul le are in cadrul aplicatiei; unui rol ii sunt asociate o serie de permisiuni; rolul face legatura cu titlul rolului din colectia “Roluri”, unde sunt specificate si actiunile permise; este necesar sa avem mai multe roluri (o lista) pentru a acoperi cazul in care, de exemplu, un masterand/doctorand este si profesor in cadrul facultatii

{ identitySecret: { $type: "string" } }, -- codul secret pe care utilizatorul va trebui sa il introduca, dupa ce s-a autentificat cu un cont personal, pentru a-si confirma identitatea cu cea din facultate

Restrictii

Unique	Required	Index
facultyIdentity	facultyIdentity	facultyIdentity

Roluri

Fiecare utilizator al aplicatiei are anumite drepturi. Aceste drepturi ii permit unui utilizator sa acceseze anumite sub-categorii ale api-ului pus la dispozitie de catre aplicatie. Un rol reprezinta o grupare de drepturi (permisiuni), careia ii este asociata un nume.

Colectia “Roluri” este definita astfel:

"Roles"

{

{ title: { $type: "string" } }, -- titlul rolului; reprezinta doar un alias pentru seria de permisiuni (actiuni) asociate

{ actions: [{ $type: "string" }] } -- permisiunile (actiunile) asociate acestui titlu de rol; permisiunile reprezinta siruri de caractere prin care aplicatia sa poata extrage informatii precum “rolul x are permisiune de GET si PUT pe resursa /studenti/note

}

Restrictii

Unique	Required	Index
title	title	title

Studenti

Studentii reprezinta o categorie speciala de utilizatori, ei reprezentand un actor “critic” in cadrul aplicatiei (aplicatia nu ar avea sens fara studenti). Fiecare student este identificat in mod unic prin numarul sau matricol si trebuie sa apartina unui an (de exemplu licenta_1,master_2) si unei grupe (de exemplu A4, X1). Un student poate avea atribute aditionale precum “data nasterii”.

Colectia “Studenti” este definita astfel:

"Students"

{

{ user: { $type: "string" } }, -- fiecare student este un utilizator; acest camp realizeaza legatura intre utilizatorul inregistrat in aplicatie si identitatea lui de student

{ facultyIdentity: { $type: "string" } }, -- fiecare student va avea asociat un cont de email stabilit de comun acord cu facultatea care va stabili identitatea studentului in cadrul facultatii

{ registrationNumber: { $type: "string" } }, -- numarul matricol al studentului

{ birthDate: { $type: "date" } }, -- data nasterii studentului

{ academicYear: { $type: "string" } }, -- anul in care se afla studentul; este un sir de caractere precum “licenta_1”, “master_2” sau “doctorat_1”

{ academicGroup: { $type: "string" } } -- grupa in care este studentul; exemple: “A4”, “X1”

}

Restrictii

Unique	Required	Index
user, registrationNumber	facultyIdentity, registrationNumber, birthDate, academicYear, academicGroup	User, facultyIdentity, academicYear, academicGroup

Note

Fiecare student va avea o serie de note. Conceptul aplicatiei sustine crearea dinamica a modulelor evaluarii, astfel incat fiecare nota va apartine unei materii si, implicit,

unui modul al acelei materii. Notele apartin doar studentilor, iar legatura intre un student si notele sale se face prin intermediul numarului matricol. Un student poate promova sau nu o materie, asa ca acest lucru trebuie inscris in baza de date.

Colectia “Note” este definita astfel:

"Grades"

{

{ registrationNumber: { $type: "string" } }, -- notele pot apartine doar unui student, asa ca vom identifica notele unui student pe baza numarului sau matricol

{ courseId: { $type: "string" } }, -- identificatorul cursului (materiei) la care un student a primit o nota

{ moduleId: { $type: "date" } }, -- identificatorul modulului cursului (materiei) la care un student a primit on nota

{ grades: **[** -- notele (intr-o lista) obtinute de studentul cu numarul matricol curent, la cursul si modulul curent (cu id-urile specificate)

{

value: { $type: "double" }, -- valoarea notei (pozitiva sau negativa)

date: { $type: "date" } -- data la care a fost pusa nota

}

] }

}

Restrictii

Unique	Required	Index
	registrationNumber, courseId, moduleId, grades	registrationNumber, (courseId, moduleId)*

* Index compus, format din cele doua campuri, strict in acea ordine

Profesori

Profesorii reprezinta o categorie speciala de utilizatori. Entitatea “Profesori” este speciala in cadrul aplicatiei. Fiecare profesor are asociat un grad didactic, ii este repartizata o materie si o serie de grupe.

Colectia “Profesori” este definita astfel:

"Professors"

{

{ user: { $type: "string" } }, -- un profesor este un utilizator; acest camp face legatura intre utilizator si identitatea sa de profesor

{ facultyIdentity: { $type: "string" } }, -- fiecare profesor va avea asociat un cont de email stabilit de comun acord cu facultatea care va stabili identitatea profesorului in cadrul facultatii

{ gradDidactic: { $type: "string" } }, -- gradul didactic al profesorului

{ courses: **[** -- cursurile impreuna cu grupele de la acele cursuri pentru care profesorul este responsabil

{

courseId: { $type: "string" }, -- id-ul cursului predat de profesor

academicGroups: { [$type: "string"] } -- grupele (intr-o lista) de la cursul specificat pentru care profesorul este responsabil

}

] }

}

Restrictii

Unique	Required	Index
user	facultyIdentity, courses, gradDidactic	user, facultyIdentity,

Cursuri

Fiecare profesor va ave asociate anumite cursuri. In mod similar, fiecare student va fi evaluat la anumite cursuri. Fiecare curs fi identificat unic printr-un id, va avea un titlu asociat, un an si un semestru. Pentru ca un student sa poata fi evaluat la un curs, acel curs va trebui sa aiba o modalitate de evaluare definita anterior.

Colectia “Cursuri” este definita astfel:

"Courses"

{

{ courseId: { $type: "string" } }, -- identificatorul unic al cursurlui

{ title: { $type: "string" } }, -- titlul cursului

{ year: { $type: "string" } }, -- anul caruia ii este destinat cursul

{ semester: { $type: "string" } }, -- semestrul caruia ii este destinat cursul

{ evaluation: { $type: "string" } } -- formula evaluarii; acest camp va cuprinde operatii aritmetice pe identificatori (id-uri) de module (de exemplu: (2*modul_curs + 2*modul_seminar)/4)
}

Restrictii

Unique	Required	Index
courseId	courseId, semester title, year	courseId, (year, semester)

Module

Intrucat aplicatia va suporta definirea modalitatii de evaluare la un anumit curs de catre profesorul titular corespunzator, este necesara abstractizarea notiunii de modul evaluat. Un modul apartine unui curs si contribuie, prin formula evaluarii acelui curs, la nota finala a studentului. Cand nota finala a studentului depaseste pragul minim pentru fiecare modul, acesta promoveaza materia.

Modalitatea de evaluare va putea aplica o formula pe unul sau mai multe module. Un modul are un identificator unic si o formula de evaluare, care poate fi aplicata, precum formula evaluarii cursului, doar ca recursiv, pe alte submodule. De asemenea, un modul va avea un prag minim de promovare si un punctaj maxim.

Colectia “Module” este definita astfel:

"Modules",
{
{ moduleId: { $type: "string" } }, -- identificatorul unic al modulului

{ formula: { $type: "string" } }, -- formula dupa care se evalueaza punctajul acestui modul (de exemplu 2*alt_modul + ½*sub_modul)

{ min: { $type: "double" } }, -- punctajul minim care asigura promovarea la acest modul

{ max: { $type: "double" } } -- punctajul maxim posibil la acest modul; daca formula modului genereaza un punctaj mai mare decat acest maxim, punctajul va fi inlocuit de acest maxim
}

Restrictii

Unique	Required	Index
moduleId	moduleId, formula, min, max	moduleId,

Modelarea componentelor aplicatiei