

**Documentație de proiect Inginerie Software**

Proiect realizat de : Anton Ioana și Fariseu Teodora

Grupa 30235

An universitar 2021-2022

Contents

[Introducere 3](#_Toc92044845)

[Proiectare 4](#_Toc92044846)

[Cerințe și precizări 4](#_Toc92044847)

[Design pattern-uri folosite 9](#_Toc92044848)

[Detalii de implementare 11](#_Toc92044849)

[Tehnologii folosite 11](#_Toc92044850)

[Manual de utilizare 11](#_Toc92044851)

[Funcționalitate în timp real 12](#_Toc92044852)

[Resurse adiționale 13](#_Toc92044853)

[Concluzii 15](#_Toc92044854)

[Bibliografie și resurse 16](#_Toc92044855)

# Introducere

Învățarea este un aspect foarte important al vieții. Tocmai de aceea multe aplicații doresc să transforme acest lucru în ceva ușor și distractiv prin metode cât mai interesante. O modalitate foarte comună de învățare și evaluare folosită de acestea este reprezentată de testele grilă. Ele sunt o metodă clasică și ușor de evaluat a unor cunoștințe, ale căror rezultate pot fi imediat vizibile

Dar pe lângă o metodă foarte ușoară de evaluat similară cu testele grilă, ce ar mai putea motiva pe cineva să învețe? Competiția! De aceea, acest proiect urmărește să combine metodele de învățare prin teste folosite de site-uri precum Kahoot cu un aspect puțin mai competitiv decât cel folosit de Duolingo.

În acest proiect se va urmări participarea la teste de diferite materii și procesarea de date precum scorul și răspunsurile corecte în timp real. După fiecare test la care participă, studentului i se va acorda la statistici numărul de puncte corespunzătoare. Însă deoarece competitivitatea poate deveni nocivă dacă este individuală, se va oferi și posibilitatea de a te alătura unui grup, putând colabora cu membrii din acesta în a concura cu alte grupuri.

# Proiectare

## Cerințe și precizări

Rezultatul la care se dorește să se ajungă este un site unde studenții pot să provoace alți studenți la quiz-uri în timp real, sau chiar să provoace grupuri întregi. Un site pe care profesorii își pot salva quiz-uri cu întrebări pentru ulterioare evaluări. Bineînțeles, ei nu își vor putea face de cap. Vor putea exista moderatori și admini care să se ocupe de respectarea regulilor și administrarea rețelei.

Principalele substantive relevante ce pot fi identificate în enunțarea conceptului sunt ”student”, ”profesor”,”moderator”,”admin”,”grup”,”întrebare”. Astfel, putem deduce următoarea schemă orientativă pentru tabelele din baza de date (creată de Fariseu Teodora):

A picture containing outdoor object, night, dark, star

Description automatically generated

Bineințeles, după cum reiese din diagrama de clase, site-ul este pe bază de roluri, existând mai multe tipuri de conturi. În general, orice tip de cont, pentru a avea puterea de exercitare a drepturilor sale, trebuie ca acesta să fie logat.

**Use Case**: Utilizatorul se loghează în aplicație  
Primary Actor: Utilizator  
Main success scenario:

1. Utilizatorul introduce un nume de utilizator și o parolă

2. Utilizatorul apasă pe un buton de log in

3. Utilizatorul este acum logat în contul său  
Alternative Sequence: Utilizatorul nu există, caz în care trebuie să își creeze un cont

În funcție de tipul de cont, fiecare utilizator poate acea diferite drepturi

* Elev/Student - Poate să :
  + iși creeze un cont, după care să folosească acel cont pentru logare
  + să își vizualizeze datele personale
  + se înscrie sau să părăsească grupuri de studenți
  + să își vizualizeze drepturile într-un grup și diferite statistici (punctaje obținute la jocuri anterioare, jocuri jucate în total, jocuri câștigate etc.). În mod individual, poate provoca alt jucător la o rundă de răspuns la întrebări pe un set de întrebări predefinite ale site-ului

**Use Case**: Provocarea altui student la un meci de răspuns la întrebări  
Primary Actor: Student  
Main success scenario:

1. Utilizatorul se loghează ca student

2. Studentul alege un alt student pe care sa-l provoace

3. Celălalt membru acceptă invitația

4. Cei doi studenți intră în meci  
Alternative Sequence: Utilizatorul provocat refuză invitația, Caz în care utilizatorul care l-a provocat primește o notificare în legatură cu asta

* Profesor : Aceleași drepturi ca ale unui student (în afară de înscrierea într-un grup și provocările individuale). Pe lângă acestea, el poate:
  + face teste cu întrebări predefinite ale site-ului
  + poate crea/șterge grupuri de studenți, precum și edita detaliile acestuia
  + poate adăuga/da afară membri din grup
  + poate provoca alt grup la o întrecere de răspuns la întrebări

**Use Case**: Adăugarea unui student într-un grup  
Primary Actor: Profesor  
Main success scenario:

1. Profesorul se loghează cu un cont de profesor

2. Profesorul accesează grupul în care vrea să adauge studentul și va da click pe un buton corespunzător

3. Profesorul va căuta studentul după nume

4. Profesorul va adăuga studentul dorit în grup  
Alternative Sequence: Studentul respectiv nu este găsit. Profesorul va fi înștiințat de acest lucru și va fi pus să caute din nou studentul dorit

* Moderator: poate interveni când e haos prin grupuri (să dea afară membri, etc.). poate adăuga/sterge/edita întrebări predefinite ale site-ului

**Use Case**: Editarea unei întrebări  
Primary Actor:Moderator  
Main success scenario:

1. Utilizatorul se loghează cu cont de moderator

2. Moderatorul caută întrebarea dorită

3. Moderatorul editează ce dorește din întrebare  
Alternative Sequence: Întrebarea este ștearsă întretimp de alt moderator. Caz în care se primește un mesaj de eroare

* Admin: control asupra oricarei informații (inclusiv conturi)

**Use Case**: Modificarea datelor unui utilizator  
Primary Actor: Admin  
Main success scenario:

1. Utilizatorul se loghează cu cont de admin

2. Adminul caută utilizatorul după nume

3. Adminul selectează utilizatorul și câmpul dorit de modificat

4. Adminul validează noile date printr-un buton  
Alternative Sequence: Adminul a introdus date invalide, caz în care valorile reale nu se vor modifica în baza de date

Adunând toate aceste scenarii și drepturi, obținem următoarea diagamă de clase (realizată de Anton Ioana)

Diagram, schematic

Description automatically generated

Însă lucrurile nu se rezumă doar la clase și drepturi. În unele situații, unele lucruri necesită puțin mai multă gândire, fie pentru că implică mai multe clase simultan, fie pentru că trebuie verificate mai multe condiții

De exemplu, atunci când se trimite o invitație unui alt grup la un meci, această acțiune trebuie trecută prin mai multe etape, care uneori pot fi realizate chiar de alte clase în paralel. Mai multe acțiuni de la mai multe surse, în special venite în paralel, pot părea prea intimidant pentru a fi abordate. Pentru a organiza mai ușor astfel de acțiuni, se poate folosi o diagramă de acțiuni. Aceasta oferă posibilitatea de a organiza acțiunile în funcție de clasa care se ocupă de ele și este un foarte mare ajutor în a vedea ce acțiuni pot fi executate în paralel. Atunci când un membru vrea să inițieze o provocare către alt grup, mai multe lucruri trebuie verificate (Diagrama realizată de Fariseu Teodora):Graphical user interface, application, Teams

Description automatically generated

Anumite funcționalități la nivel de implementare necesită parcurgerea a mai multor metode din clase diferite. Acest lucru poate ridica probleme în urmărirea modului în care clasele implicate comunică între ele și ce date își transmit. Pentru a putea implementa o astfel de funcționalitate este recomandată proiectarea unei diagrame de colaborare în primă fază. Mai jos avem diagrama de colaborare pentru scenariul de utilizare "Add Member", aceasta ilustrează modul în care un membru este adăugat în grup. (Diagramă realizată de Anton Ioana)

Diagram

Description automatically generated

## Design pattern-uri folosite

Design pattern-urile sunt convenții de proiectare bune pentru a rezolva situații des întâlnite

O situație întâlnită destul de des este anunțarea unui observator atunci când obiectul lui de interes își modifică valoarea. În ajutorul acestei probleme vine un design pattern cu nume intuitiv, și anume Observer. Prin principiul lui de a păstra un obiect observabil, alături de o listă de observatori și a anunța observatorii, este perfectă pentru a implementa sisteme de notificări (de exemplu, când grupul tău este provocat la un meci. Diagramă realizată de Fariseu Teodora)

Graphical user interface, application

Description automatically generated

Diagrama de colaborare pentru acest design pattern(realizată tot de Fariseu Teodora):

A picture containing arrow

Description automatically generated

Design pattern-ul Singleton presupune restricționarea instanțierii unei clase la o unică instanță. Am ales de implementat acest design pattern pentru a asigura o unică și sigură conexiune cu baza de date în momentul în care avem nevoie de accesarea ei pentru a executa anumite interogări. Din acest motiv, constructorul clasei Database este privat, dar instanța unică a acesteia este globală, aceasta putând fi accesată de orice clasă Controller.(Diagramă realizată de Anton Ioana)

Diagram

Description automatically generated

# Detalii de implementare

## Tehnologii folosite

1. C# și ASP.NET (MVC) și Javascript pentru back-end
   1. C# și ASP.NET pentru a ne familiariza cu un limbaj nou, iar în urma documentării am decis că sunt modalități relativ ușoare pentru a implementa o aplicație Web
   2. Arhitectură MVC pentru a avea o organizare intuitivă și a putea separa front-end-ul de back-end
   3. JavaScript pentru gestionarea evenimentelor din aplicație(apăsare butoane spre exemplu)
2. MySQL pentru baza de date
   1. Deoarece este o platformă ușor de utilizat și suntem deja familiarizați cu ea
3. HTML și CSS pentru front-end
   1. Deoarece sunt printre cele mai populare limbaje pentru design web, iar CSS permite personalizarea aspectului unei pagini web într-un mod detaliat și ușor
   2. Librăria bootstrap pentru a avea acces la anumite componente deja existente precum butoane sau meniuri dropdown
4. JSON pentru serializare

## Manual de utilizare

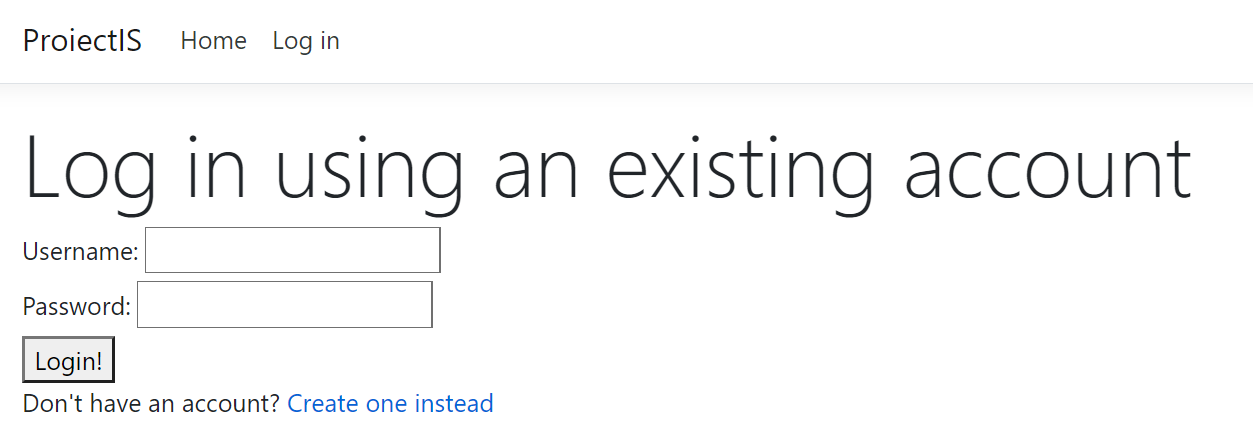
Proiectul complet poate fi găsit la <https://github.com/FTeodora/ProiectIS/tree/master/ProiectIS>.

Rulându-se aplicația, utilizatorul va fi redirecționat la o fereastră de index, unde se află un text de umplutură, cu un meniu de navigare ce conține două opțiuni:

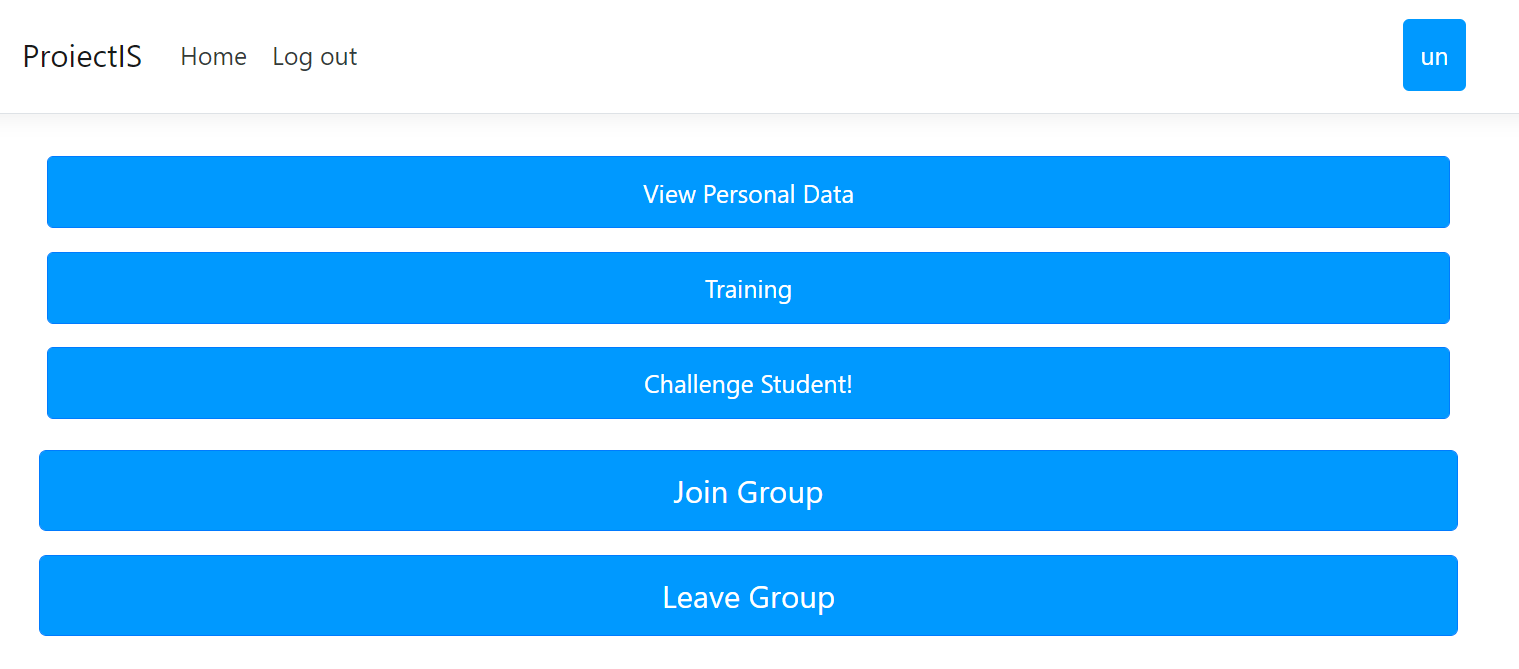


Home: Care te aduce înapoi la această pagină menționată

Login: Care te va redirecționa pe pagina de introducere a utilizatorului și parolei asociate. Nu ai cont? Nicio problemă! Îți poți crea din opțiunea de înregistrare de sub chenarele de text



După ce ți-ai introdus numele și parola și ai apăsat pe butonul de Login, vei fi redirecționat pe o pagină cu opțiunile pe care le ai în funcție de rol. De exemplu, pagina de student ar trebui să arate cam așa:

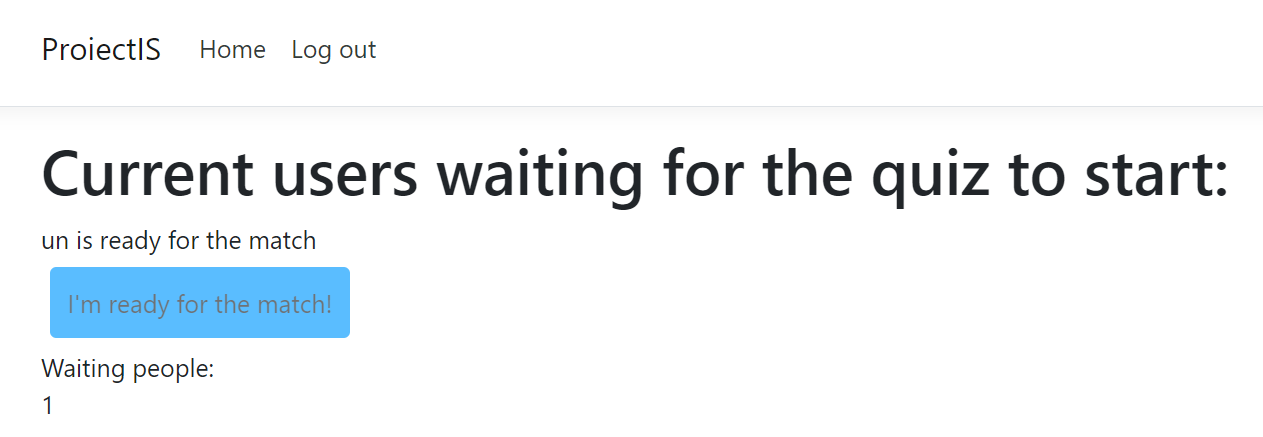


## Funcționalitate în timp real

Atunci când un student vrea să provoace alt student la un meci, ei vor trebui să comunice în timp real unul cu altul (spre exemplu, fiecare să știe când e pregătit să înceapă quiz-ul). Acest lucru se poate realiza ușor cu hub-uri. Un hub este un mod de a conecta mai multe calculatoare împreună, trimițând însă informațiile la toți cei conectați. Deși acest lucru pare un risc de securitate, în cazul nostru este un lucru util, deoarece dorim să anunțăm toți participanții de la un anumit eveniment.

Conceptul acesta de Hub este implementat folosind SignalR. Atunci când un student provoacă alt student la un meci, el este trimis într-o cameră de așteptare cu un anumit ID care va fi trimis și celuilalt participant invitat. Când un student acceptă o invitație la un meci, acesta este redirecționat în camera cu cel ce l-a provocat.

Din acest moment, cei doi sunt pe un Grup de Hub, prin care ei pot comunica dacă sunt gata pentru meci.



Atunci când amândoi sunt gata, vor fi redirecționați la un quiz cu același ID

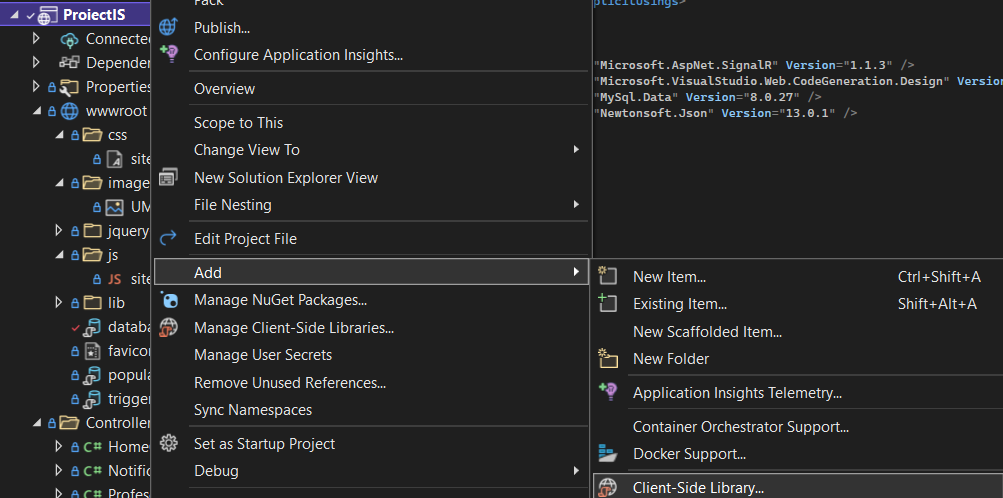
## Resurse adiționale

Pe lângă tehnologiile utilizate, a mai fost nevoie de instalări suplimentare. Acestea nu trebuie realizate dacă proiectul este descărcat de pe git

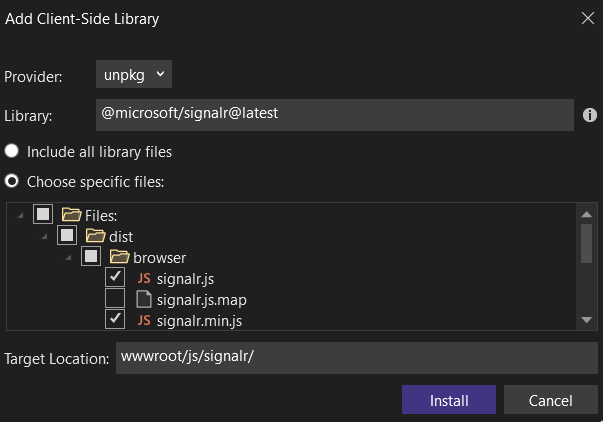
Pentru alegerea datei pe pagina de provocare a unui grup, s-a folosit <https://jqueryui.com/datepicker/>

Pentru ca aspectul de real time să funcționeze, trebuie să se instaleze pachetele necesare pentru a rula SignalR.

Va fi nevoie de dependențe suplimentare care se vor adăuga din Project Manager : Click dreapta pe proiect > Add > Client-Side Library



În fereastra care se va deschide, se selectează la provider unpkg, se scrie @microsoft/signalr@latest la Library și wwwroot/js/signalr/ în Target Location, alegându-se DOAR opțiunile selectate din browser (nimic mai mult de jos) și se va da click pe install



# Concluzii

Acest proiect a fost o demonstrare a faptului că, indiferent care este limbajul de programare, procesele de proiectare și convențiile precum design pattern-urile, sunt aproape la fel.

De asemenea, a fost o oportunitate de îmbunătățire a abilităților de proiectare, observare din timp a unor erori ce pot afecta grav sistemul, și adaptare pe noi limbaje. Am exersat aplicarea unor metode de proiectare, dar am și învățat noi lucruri de implementare, precum metode de comunicare în timp real

# Bibliografie și resurse

<https://docs.microsoft.com/en-us/aspnet/core/tutorials/signalr?WT.mc_id=dotnet-35129-website&view=aspnetcore-6.0&tabs=visual-studio>

<https://docs.microsoft.com/en-us/aspnet/core/signalr/introduction?WT.mc_id=dotnet-35129-website&view=aspnetcore-6.0>

<https://docs.microsoft.com/en-us/aspnet/core/signalr/groups?view=aspnetcore-6.0>

<https://docs.microsoft.com/en-us/dotnet/standard/events/observer-design-pattern>

<https://www.computerhope.com/jargon/h/hub.htm>

<https://www.w3schools.com>

<https://getbootstrap.com/>

<https://getbootstrap.com/docs/5.1/getting-started/introduction/>