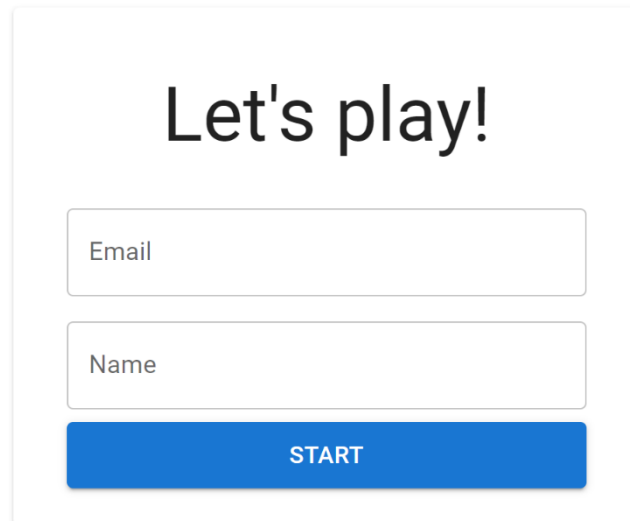


Documentatie Test Tehnic

În următoarele rânduri va voi prezenta aplicatia mea care consta în autentificarea unui user și posibilitatea lui de a participa la un quiz în urma caruia primește un scor pe baza răspunsurilor corecte. Aplicatia este dezvoltata cu ASP.NET Framework pentru backend și React JS pentru frontend, iar pentru baza de date am folosit SQL Server. IDE-urile folosite sunt Visual Studio pentru backend, respectiv Visual Studio Code pentru frontend. În continuare, voi descrie pașii pe care i-am îndeplinit pentru a ajunge la proiectul final.

În primul rând, am creat o aplicație de tip React (gamification-client), iar pentru design, am folosit biblioteca MUI (Material UI). Prima componentă pe care am dezvoltat-o a fost cea de login, cu ajutorul extensiei `reactFunctionalComponent`. Am creat un form cu două câmpuri pentru a introduce adresa de email și numele userului și un buton pentru a începe quiz-ul.

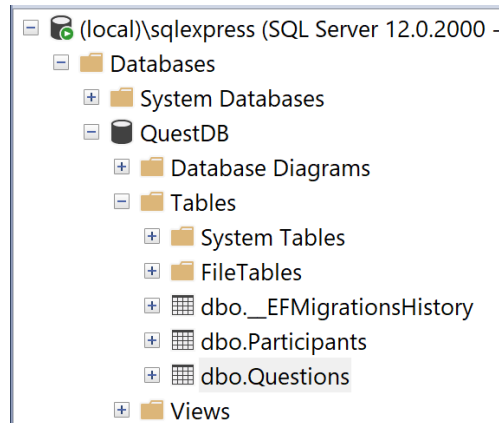


The image shows a login form with a light gray background. At the top, the text "Let's play!" is displayed in a large, bold, black font. Below this, there are two input fields: the first is labeled "Email" and the second is labeled "Name". Both fields have a light gray border and a small shadow. At the bottom of the form is a blue button with the word "START" in white, uppercase letters.

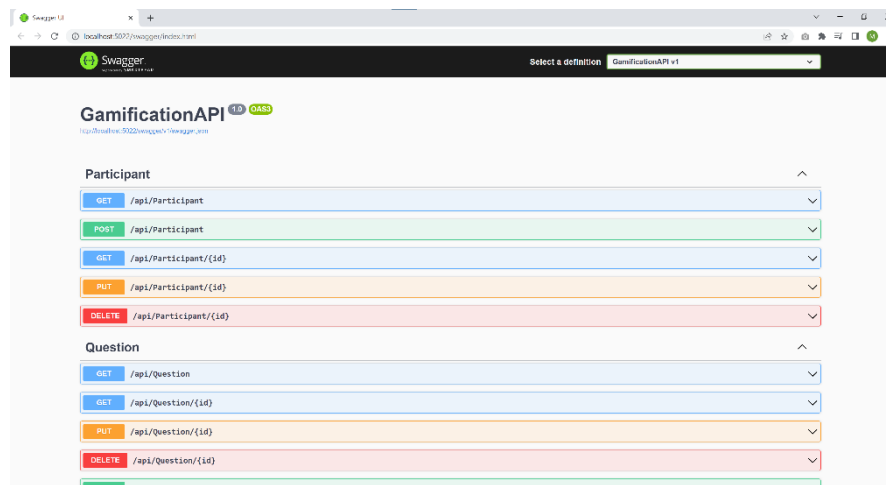
Pentru a obține valorile din cele două câmpuri, în folderul Hooks am creat fișierul `useForm.js` cu care vom putea obține stările valorilor respective.

În pasul următor, am creat proiectul pentru backend (QuestAPI). Pentru a putea crea baza de date și a face legătura cu ea, am instalat Entity Framework, apoi am creat cele două modele (Question și Participant) și `QuestDbContext`. Pentru a putea utiliza `QuestDbContext`, am folosit Dependency Injection design pattern, urmand să fac o migrație și să creez baza de date.

Pentru baza de date am folosit SSMS (SQL Server Management Studio) și am introdus întrebările și răspunsurile în tabelul Questions.



In Visual Studio am creat apoi doua controllere (QuestionController si ParticipantController). Pentru a introduce useri in baza de date dupa ce au completat cele doua campuri, ne vom folosi de metoda POST din controllerul Participant. Pentru a testa metodele API, m-am folosit de Swagger.



Am folosit biblioteca Axios pentru a face legatura dintre backend si frontend. Cu ajutorul ei, am putut sa creez request-uri catre un endpoint si sa returnez date de la un endpoint.

In continuare, am creat componenta Quest unde cu ajutorul metodei GET din controllerul Question am implementat alegerea random a 5 intrebari din totalul de 10.

In ultima componenta, Result, am afisat scorul obtinut de user, alaturi de durata in care a terminat testul si intrebarile cu raspunsurile corecte. Metoda POST din controllerul Question a fost folosita pentru a trimite date cu indexul intrebarilor, respectiv raspunsurile date de user.

In concluzie, a fost o mare placere pentru mine sa fac aceasta aplicatie din care am invatat lucruri noi si am intampinat erori pe care am reusit sa le rezolv dupa cautari multiple. Imi doresc foarte mult sa particip la internship si sa fac parte din echipa Accesa. Sunt pregatita sa dau tot ce am mai bun si sa invat orice imi este dat.

Bibliografie

- <https://axios-http.com/docs/intro>
- <https://legacy.reactjs.org/docs/getting-started.html>
- <https://www.youtube.com/@CodAffection>
- <https://mui.com/material-ui/getting-started/overview/>
- <https://www.geeksforgeeks.org/reactjs-lifecycle-components/>
- <https://stackoverflow.com/>