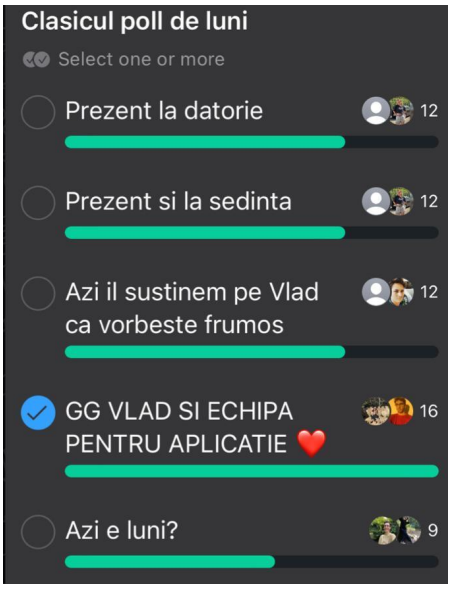
PROBĂ POLL_IT

~site poll-uri~



Opiniile sunt mai importante ca niciodată. Platformele de sondaje permit organizatorilor să culeagă feedback direct de la audiența lor și să înțeleagă mai bine nevoile și dorințele acesteia. Astfel, **Proba tehnica** de anul acesta consta în realizarea unei aplicații web complet funcționale care are ca tema realizarea unor poll-uri de către utilizatori. Mai jos o sa fie descrise detaliat ambele parti ce alcătuiesc aplicația **(frontend + backend)** și diferitele aspecte pe care voi trebuie sa le implementati.

O parte foarte importantă anul acesta este **procesul de legare** ale celor 2 parti, lucru care poate sa fie făcut în 2 moduri:

1. ***Pe parcursul probei*** după fiecare implementare de feature.
    - Dacă optati pentru aceasta metoda, este foarte importantă ordinea în care sunt dezvoltate feature-urile aplicației, așa ca va lăsăm mai jos sugestia noastra in legatura cu ordinea lor:
        - *Navbar & Footer*
        - *Autentificare(Frontend -> Backend)*
        - *Create & Get Poll (Frontend -> Backend)*
        - *Delete & Vote (Frontend -> Backend)*
2. ***La finalul probei*** cand atat partea de backend (API-ul) cat și partea de frontend (UI-ul) sunt terminate.

    Aici aveti atasat linkul catre un mock-up care prezinta cum ar trebui sa arate site-ul, atat pe desktop, cât și pe mobile.

Link pentru resurse utile pe care le puteți folosi de-a lungul probei.

Încărcarea probei

Pentru a va obișnui cu modul de lucru din department, de-a lungul probei veți lucra cu Github. La începutul probei va trebui sa creați un **repository privat,** la care sa oferiti acces mentorului vostru.

Ar fi util pentru voi sa dati commit-uri frecvent cand lucrati la proba pentru a oferi posibilitatea mentorului sa vada evolutia noastra, iar în cazul în care aveți nevoie de ajutor, să aibă acces la codul vostru.

La finalul probei, va trebui sa faceti repo-ul pe care ați lucrat **public**.

Deadline: 6 decembrie 2023, 23:59

BACKEND (API)

Register & Login

> Funcțiile de **înregistrare** și **autentificare** în aplicațiile web sunt esențiale pentru securitatea și personalizarea experientei utilizatorilor. Ele asigură accesul doar utilizatorilor autorizați, permit personalizarea și controlul accesului la funcționalități, precum și urmărirea acțiunilor utilizatorilor în aplicație.

Observație! Un aspect important al comunicării prin intermediul HTTP este statusul requestului, astfel ca este sugerat ca la fiecare răspuns trimis înapoi de către server sa se ataseze si un http status, care nu este nimic altceva decat un cod care poate sa aibe diverse semnificații (de aici provine celebrul *Error: 404 Not Found*)

- (HTTPS STATUS)

Register

- Trebuie să creați un **endpoint** care să permită preluarea datelor necesare pentru înregistrarea unui nou utilizator în entitatea **User** din baza de date: Minimal, aceste va fi format din :
  - ***id: Int,***
  - ***email: String,***
  - ***password: String***
- Utilizatorii **nu** ar trebui să furnizeze **manual ID-urile**; acestea vor fi generate automat de către baza de date.

- În endpoint **verificati** daca **exista** deja un cont cu o **Email-ul** primit

```
{
    "error": "That email address is already in use!"
}
```

Posibila eroare primită în urma unui request catre endpointul de înregistrare

Login

- Trebuie să implementați un **endpoint** care primește **Email-ul** și **Parola** unui utilizator, verifică autentificarea (combinatia email parola sa fie corectă), și apoi furnizează un token (cum ar fi un JWT sau SessionCookies) pentru a identifica acest utilizator în cererile ulterioare.
- În cadrul acestui endpoint **verificati** daca **Email-ul** și **Parola** primite sunt ale unui utilizator **existent**:

Posibila eroare primită în urma unui request către endpointul de Login

CRUD = { Create, Read, Update, Delete }

Operațiile CRUD stau la baza oricărui API, având rolul de a permite utilizatorului să interacționeze cu informațiile stocate în baza de date. Pentru a completa task-ul va fi nevoie să creați un set de endpoint-uri care acoperă cateva din operațiile CRUD necesare aplicației.

- GET /polls = întoarce toate poll-urile din baza de date
- POST /polls = creeaza un poll, îl inserează în baza de date. Acesta trebuie sa aibe 3 variante de răspuns și întrebarea din poll
  - Bonus
- DELETE /polls/:id = șterge un poll din baza de date pe baza id-ului obtinut din url.
  - Bonus
- PATCH /polls/vote/:id = sistemul de votare

BONUS

Password Encryption:

- Pentru a asigura **securitatea** userilor, este important ca orice parolă să fie **criptată** cu un algoritm de tip hash(ex. **bcrypt**) înainte de a fi salvată în baza de date.

Validate email & password:

- Email-ul și parola vor fi **validate** înainte de a procesa requestul (trebuie sa respecte un anumit tipar specific fiecărui field), iar în caz contrar se vor returna mesaje de eroare precum:

```
{
    "email": ["the field must be a valid email ", "the field must end in @gmail.com"],
    "password": ["the field is not between 8 and 32 characters"]
}
```

Posibila eroare primită în urma validării mail-ului și parolei utilizatorului

Validari:

- e-mailul este de tip '**@gmail.com**' (hint: **regex)**
- parola are între **8 si 32 de caractere**
- campul **'parola'** și **'confirma parola'** din formularul de inregistrare trebuie sa fie **la fel**

Endpoint Validation:

- *Create (Post):*
  - Fiecare poll trebuie să aibă un **owner** (pentru a realiza acest lucru, poll-ul trebuie să rețină **id-ul** userului care l-a creat, acest concept fiind o relație între poll și user).
  - De asemenea, prin introducerea unui **JWT/cookie** în header-ul requestului HTTP, userul poate fi extras în cadrul endpoint-ului și validat ca owner al poll-ului.
- *Ștergere (Delete):*
  - Ștergerea unui poll ar trebui **restricționată** dacă userul nu este cel care l-a creat. Trebuie sa se verifice dacă id-ul utilizatorului care a trimis requestul este același ca cel care a creat poll-ul.

Vote a poll:

- Pentru că un poll nu are sens dacă nu-l poți vota, va trebui să implementați un **endpoint** care incrementeaza numărul voturilor pentru răspunsul din **poll** care a fost selectat.
- **Votarea** la un poll ar trebui **restricționată** dacă userul este logat sau nu pentru a putea stoca id-ul utilizatorului în cadrul fiecărui poll.
- Fiecare poll trebuie să rețină **id-urile** userilor care au **votat**. Acest lucru este necesar intrucat dorim sa restrictionam votul utilizatorilor care deja au votat în cadrul poll-ului respectiv.

FRONTEND (UI)

Navbar

Navbar (Navigation bar) este un element al site-ului nostru care ne permite să navigăm prin acesta effortlessly. Acesta conține link-uri care ne ajută să indexăm conținutul de pe pagina noastră pentru a-l accesa mai simplu. În mock, acesta va fi reprezentat de ‘bara’ din partea de sus a ecranului și va conține pentru probă 3 informații esențiale: **Logo**, **Register**, **Login**.

După ce un utilizator a fost logat cu succes, butoanele de Login si Register nu vor mai avea sens, așa că vor fi înlocuite de butonul de Logout. De asemenea, trebuie să menținem constant bara de navigare pe pagină, inclusiv atunci când utilizatorul derulează conținutul.

Footer

Footerul unui website constituie o componentă cu rol asemanător Navbarului. De obicei, pentru aplicațiile noastre și inclusiv proba tehnică, footer-ul va conține informații utile legate de eveniment, mai exact linkuri către pagini LSAC ale evenimentului (spre deosebire de Navbar, Footer-ul nu va fi menținut constant la derulare).

Autentificare

- Butoanele **Login** și **Register** din bara de navigație vor deschide câte o fereastră **pop-up**.
- Un **pop-up** este o *fereastră* care se deschide deasupra paginii principale și nu mai permite utilizatorului să interacționeze cu elementele din spatele ei.

Înregistrare (Register)

Înregistrarea va fi un formular alcătuit din următoarele câmpuri: {**email**, **parolă**, **confirmare_parola**}. După ce utilizatorul apasă butonul de confirmare, se va afișa un mesaj de confirmare sau va fi redirectionat catre pop-up-ul de login, dacă înregistrarea a decurs cu succes. Altfel, un **mesaj de eroare** va fi afișat sub formular.

Conectare (Login)

Procesul de conectare este asemănător cu cel de înregistrare, doar că formularul va conține doar câmpurile pentru **email** și **parolă**. După ce utilizatorul apasă butonul de confirmare, va fi redirectat către pagina principală, dacă totul a mers bine. Dacă nu, se va afișa un **mesaj de eroare**.

De asemenea, platforma trebuie să țină minte datele utilizatorului pentru a-l putea identifica pe viitor. Acest lucru se poate realiza folosind **local storage** sau **cookies**.

Short recap pentru o implementare **blană bombă** a probei:

Navbar-ul va contine inițial:

- **Logo** (luat din figma)
- **Register** = la apăsarea Register vom deschide o modală în care utilizatorul își va putea crea un cont
- **Login** = la apăsarea Login, vom deschide o modală unde utilizatorul se va putea loga deja cu credentialele create anterior
- **Logout** = Register si Login dispar dacă utilizatorul este autentificat.

Pagina de poll-uri

Această pagină reprezintă ***inima platformei***. Aici, utilizatorii pot vedea sondajele disponibile, pot răspunde la întrebări sau pot verifica rezultatele sondajelor.

Fiecare sondaj va fi reprezentat printr-un card individual, care conține **titlul** sondajului, **întrebarea** și **opțiunile** de răspuns. Creați o componenta unică pentru un card de sondaj pe care să o refolosiți mai apoi.

Le vom permite utilizatorilor să interacționeze cu sondajele astfel:

- Selectarea unei opțiuni: odată apăsată o opțiune, va apărea un buton de **Vote** prin care utilizatorul își poate **exprima opinia**
- Apăsarea butonului de **Delete** va comunica backend-ului ștergerea unui poll
- Bonus: Vote Poll

Crearea unui poll

Pe platforma de poll-uri, fiecare poll va trebui să conțină o **întrebare** și **3 variante** de răspuns. Crearea unui poll va fi declanșată de apăsarea butonului de **Create Poll**, care va deschide o modală. Aceasta va contine un formular alcătuit din:

- **4 inputuri**: unul pentru întrebarea poll-ului și 3 pentru variantele de răspuns
- **buton de submit**: care va comunica backend-ului crearea poll-ului folosind datele introduse în formular.

By default, în cadrul fiecărui poll se va putea alege o singură variantă de răspuns.

Bonus: Multiple Options

Bonus: Multiple Answers

Delete Poll

Este necesar, desigur, să avem posibilitatea de a șterge un poll (de pe fața Pământului, sau măcar de pe site). Pentru asta, un poll va avea și un buton de **Delete**. Apăsarea acestui buton trebuie să comunice backend-ului ștergerea poll-ului din baza de date, pentru ca acesta să nu mai apară altor utilizatori.

Bonus: Only Delete My Polls

Responsiveness

Un alt concept esențial în crearea site-urilor noastre este acela de **responsiveness**. La ce se referă? E un buton? E o funcție? Ce este? În contextul unui website, acest concept înglobează abilitatea de a adapta și afișa corect pe dispozitive cu diferite dimensiuni de ecran componentele noastre din website. Cum facem asta? În general, vom stiliza componenta (CSS) în diferite moduri pentru diferite dimensiuni ale paginii.

BONUS:

Only Delete My Polls:

Este recomandat să vă folosiți de sistemul de autentificare pentru a permite **doar** userului **autentificat** să își șteargă **doar propriile** poll-uri. Asta presupune să știți care poll-uri sunt ale cui, implementarea fiind la latitudinea voastră. Astfel, Se va permite doar creatorului poll-ului să șteargă un poll.

Ca parte a **bonusului**, doar poll-urile proprii user-ului logat vor avea buton de delete. În plus, atât atunci când faceți requestul (frontend), cât și când procesați cererea în backend, va trebui să verificați din nou validitatea cererii.

Vote Poll:

La ce sunt bune poll-urile dacă nu putem vota? **La nimic!**

Cum s-a menţionat şi anterior, widgetul de poll va trebui să aibă opţiuni selectabile (radio dacă este un poll cu alegere unică, checkbox dacă vorbim despre alegere multiplă). Odată ce utilizatorul şi-a ales opţiunile, i se va permite să îşi finalizeze şi să îşi confirme alegerea, adică să voteze prin apăsarea butonului de **Vote**. Această acţiune trebuie să comunice backend-ului cererea spre procesare.

Desigur, vom permite **doar** utilizatorilor **autentificaţi** să voteze **o singură dată**. Pe lângă validările ce vor exista în backend, acest lucru trebuie comunicat şi pe frontend:

- Se permite selectarea de opţiuni doar utilizatorilor autentificaţi
- Se afişează butonul de vot doar utilizatorilor autentificaţi
- Dacă utilizatorul autentificat a votat deja la un poll, poll-ul va reflecta alegerea deja realizată şi nu i se va mai permite să selecteze alte opţiuni (hint: disabled)

Multiple Options Poll:

În cadrul acestui bonus, puteţi implementa un poll cu număr variabil de opţiuni. Un poll valid va avea minim 3 variante de răspuns. În cadrul formularului de creare a unui poll, iniţial, vor apărea **3 inputuri** pentru variantele de răspuns ale poll-ului, iar sub acestea va exista un buton de adaugare de variante de răspuns. La fiecare apăsare a acestui buton, în formular, se va mai adăuga încă un input pentru o nouă variantă de răspuns.

Multiple Answers Poll:

Pentru acest bonus, puteţi permite utilizatorilor să aleagă mai multe răspunsuri simultan la un sondaj. În cadrul formularului de creare a unui poll va trebui să adăugaţi o modalitate prin care să se poată selecta tipul poll-ului: **single-choice** / **multiple-choice** (de ex. un dropdown).