LSAC IT - Proba Tehnică 2022

Deadline: 14.12.2022, ora 23:59

Frontend

Proba de frontend va avea la bază mock-ul primit din partea departamentului de design: mock (*credits: Diana Preda*). O dată ce vă logați pe Figma, veți putea explora elementele mock-ului folosindu-vă de meniul din partea stângă. Din meniu puteți lua și informații precum font, dimensiuni, poziții, culori și puteți da export la poze.

Task-uri:

1. Navbar

- Plasat în partea de sus a paginii, Navbar-ul conține butoanele 'Logare' și 'Creare Cont', precum și logo-ul site-ului. Reprezintă elementul din pagină ce permite utilizatorilor să navigheze către alte pagini.
- **Cerințe**:
    - apăsarea butoanelor 'Logare' și 'Creare Cont' trebuie să deschidă modalele (pop-up-urile) corespunzătoare logării și creării unui cont
    - **afișarea condițională a elementelor** din navbar:
        - dacă user-ul nu este logat, afișăm butoanele de 'Creare cont' și 'Logare'
        - dacă user-ul este logat, butoanele de mai sus dispar, iar în locul lor apare butonul 'Delogare'
          Hint: cookie/local storage
    - afișarea constantă a navbar-ului în pagina, păstrându-se chiar și când user-ul dă scroll

2. Landing page

- Prima interacțiune a utilizatorului cu site-ul, după ce acesta a accesat link-ul
- Aceasta trebuie să respecte (pe cât posibil) mock-ul primit de la departamentul de Design
- Butonul de 'Upload a meme' trebuie să ducă utilizatorul în josul paginii, în dreptul formularului de încărcat meme-uri

3. Login/Register

- Butoanele "Logare" și "Creare Cont" din Navbar vor deschide două pop-up-uri (denumite și modale), reprezentate în mock în stânga paginii de prezentare. Acestea se vor deschide deasupra paginii de prezentare, adăugând un shadow effect (adică pagina inițială, aflată în spatele modalului, va fi umbrită) și vor avea un buton de X pentru a putea fi închise.
- După ce se face submit cu date în formularele din pop-up-uri se va face redirect către starea de user logat (descrisă mai sus)

4. Formular de upload imagini

- Butonul de 'Upload a meme' va duce utilizatorul în dreptul secțiunii unde se pot încărca meme-uri.
- Ne dorim ca formularul să funcționeze pentru a încărca, atât cu ***drag & drop***, cât și cu ***click & select***, imagini.
- Să nu se permită încărcarea altor documente, într-un format diferit de ***.png***, ***.jpg***, ***.jpeg*** și ***.gif***.

5. Footer

- Zona din partea cea mai de jos a paginii, conținând un mesaj de Copyright și iconițe ce redirecționează către paginile LSAC:
    - instagram
    - twitch
    - facebook

6. Responsivitatea design-ului

- În mock, aveți disponibil și design-ul site-ului pentru versiunea mobile. Pentru a completa acest task trebuie să vă asigurați că site-ul vostru arată plăcut vizual indiferent de rezoluția browser-ului, pe orice device.
- Insistăm foarte mult pe ideea de **responsiveness** deoarece, în ziua de azi peste 50% din accesările unui site provin de pe un device mobil.

Tehnologii recomandate:

- HTML, CSS, JavaScript.
- Framework-uri de CSS: **Bootstrap, Materialize.**
- Framework-uri pentru frontend: **React, Angular, Vue** (în general, vă fac viața mai ușoară dacă doriți să reutilizați componente fără să duplicați cod și sunt foarte robuste)

Backend

Pentru proba de backend veți avea de realizat un **REST API**, conform specificațiilor date de noi, care comunică cu o bază de date. Aplicația pe care o veți scrie va fi un API al unei platforme de meme-uri (precum cea reprezentată în mock).
Disclaimer: nu abordăm, ca task al probei, realizarea conexiunii dintre frontend și backend pentru platformă, acesta reprezentând un bonus și fiind la alegerea voastră dacă vă doriți să îl implementați.

Task-uri:

1. DB schema for users

Pentru orice backend funcțional este vital ca informațiile transmise de useri să fie stocate. Acest lucru este realizat cu ajutorul bazelor de date. Există două tipuri principale de baze de date:

- **SQL** - ex: MySql
- **NoSQL** - ex: MongoDB

Pentru a realiza proba puteți alege să folosiți oricare dintre ele.

Este important să începeți cu definirea entităților din baza de date, pentru a ști cum trebuie să arate datele transmise de către user.

Pentru acest task va trebui să definiți:

- **entitatea de user, cu cel puțin următoarele câmpuri:**
  - **ID** (un cod unic care identifică un user, acesta poate să fie un număr sau un string randomizat)
  - **Email** (email-urile ar trebui să fie unice în baza de date)
  - **Username** (username-urile ar trebui să fie unice in baza de date)
  - **Password**
- **entitatea pentru meme-uri, alcătuită din câmpurile:**
  - **ID**
  - **Description**

2. 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 toate operațiile CRUD pentru meme-uri (un meme fiind reprezentat de entitatea definită mai sus).

GET /memes

- întoarce toate meme-urile

GET /memes/:id

- întoarce un meme pe baza id-ului
- în cazul în care nu există niciun meme cu acest ID, returnați un răspuns cu statusul 404 și mesaj sugestiv

POST /memes

- creează un meme, inserându-l în baza de date

PATCH /memes/:id

- modifică (în baza de date) descrierea unui meme identificat prin id

DELETE /memes/:id

- șterge un meme din baza de date pe baza id-ului

3. Register & Login

Majoritatea site-urilor din ziua de azi au funcționalitatea de *creare de cont*, având, prin urmare, logica implementată pentru Login/Register.

Register

- trebuie să definiți un endpoint în care se preiau datele necesare pentru crearea unui entry în entitatea User din baza de date.
- pentru a nu reprezenta un pericol de securitate, orice parola ar trebui hash-uită cu un algoritm (e.g. **bcrypt**) înainte să fie stocată în baza de date.
- ID-urile nu trebuie sa fie primite de la utilizator, acestea vor fi generate automat de baza de date.

Login

- trebuie să definiți un endpoint care primește username-ul și parola unui utilizator, verifică dacă aceste credențiale sunt corecte și returnează un token sau un cookie (e.g. JWT sau SessionCookies) care are rolul de a identifica acest utilizator în requesturile lui următoare
- dacă utilizatorul nu trimite un set de date valide se returnează un raspuns corespunzator (https://developer.mozilla.org/en-US/docs/Web/HTTP/Status)

Pentru explicații adiționale și exemple, verificați documentul de aici.

4. Input Validation

Orice API trebuie să verifice datele introduse de user, pentru a asigura securitatea aplicației.
În cadrul endpoint-urilor create anterior, va trebui să adăugați verificări, cum ar fi:

`Register`:

- **`Email`** - trebuie să respecte formatul de email și să reprezinte o adresă de student la automatică (*@stud.acs.upb.ro*)
- **`Username`** - trebuie să aibă între 8 și 32 de caractere
- **`Password`** - trebuie să aibă între 8 si 32 de caractere

`Memes`:

- **Description** - lungime maximă 2500 de caractere

Pentru acest task răspunsurile de eroare ar trebui să conțină și erorile de validare.
Exemplu:

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

5. Protected Endpoints

Câteva dintre endpoint-urile create ar trebui să fie accesibile numai utilizatorilor cu anumite drepturi.

În cazul API-ului nostru:

- meme-urile ar trebui să fie create doar de utilizatori autentificați
- un meme ar trebui să îi aparțină user-ului care l-a creat (pentru a realiza acest lucru un meme ar trebui sa stocheze id-ul utilizatorului care l-a creat, acest concept numindu-se relație)
- un utilizator poate să modifice numai meme-urile care îi aparțin

Se poate verifica dacă un request este făcut de un utilizator autentificat prin introducerea token-ului sau cookie-ului în headerul request-ului HTTP și verificarea acestuia în cadrul endpoint-ului.

Pentru acest task răspunsurile de eroare pot fi avea ca **status code** și **body**:

- **401** - pentru cazul în care cineva încearcă să creeze un meme fără a fi logat

```
{
    message: "The user should be logged in to create a meme"
}
```

- **403** - pentru cazul în care cineva încearcă să modifice un meme care nu îi aparține

```
{
    message: "You can modify only your memes"
}
```

6. BONUS: File Upload

Fiind o platformă de uploadat meme-uri, trebuie să se permită upload-ul de imagini. Acest lucru poate fi făcut numai prin request-uri de tipul **form-data**. Trebuie să modificați endpoint-ul de creare a meme-urilor pentru a accepta un fișier de tip PNG de maxim 2MB.

Stocarea imaginii poate sa fie facută în 2 moduri:

1. În entitatea de meme-uri se adaugă o coloană unde va fi inserată poza cu totul (e.g. Blob in MySql, BSON in MongoDB)
2. (Recomandat) Poza va fi stocată local în sistemul de fișiere a calculatorului, iar in baza de date se va ține minte numele fisierului în care se salvează poza. De asemenea este necesară crearea unui alt endpoint, de tip GET, care returnează poza pe baza numelui.

Tehnologii recomandate

- Pentru cod backend:
    - JavaScript (NodeJS, împreună cu Express)
    - PHP (Laravel)

- Pentru baze de date:
    - MySQL
    - MongoDB

- Pentru a comunica cu baza de date:
    - PHP (Laravel) cu
        - MySQL: Eloquent
    - JavaScript cu
        - MySQL: Sequelize
        - MongoDB: Mongoose

- Pentru testare API:
    - Postman

Trimitere proba

Pentru a rezolva probă trebuie să vă creați un repository nou (inițial privat) de Github și să îi oferiți mentorului vostru acces de colaborator.
Vă încurajăm să interacționați cu Github pe tot parcursul realizării probei, prin commit-uri suggestive.
Pentru a finaliza proba, repository-ul vostru trebuie să conțină:

- fișierele sursă
- un fișier **README** în care să specificați:
    - ce task-uri ați implementat
    - **cum se rulează aplicația**
        - aici vă rugăm să includeți orice requirements: pachete care trebuie instalate, configurații, migrării, comenzi de compilat sau rulat, tot ce am avea nevoie ca să testăm aplicația voastră local

- detalii despre implementare
- ce challenge-uri ați intâmpinat și cum le-ați rezolvat

V-am pregătit și următorul set de resurse pentru probă: **resurse.**

Disclaimer: Daca vreți să utilizați tehnologii pe care nu le-am recomandat mai sus, va asumați că mentorul vostru probabil nu va putea oferi foarte mult suport tehnic pe parcursul probei.