**Problema Frontend Engineer**

# **Descriere**

Map

Description automatically generatedPe langa dezvoltare de produse software, echipa noastra are multe alte pasiuni, mai ales legate de sport. Una dintre pasiunile noastre este kitesurfing-ul. Practicantii sunt trasi pe apa de un zmeu, avand in picioare o placa. Viteza cu care te dai este de 30-50km/h, iar partea cea mai tare este ca poti zbura deasupra apei la inaltimi de cativa metri pentru cateva secunde.

Am adunat de-a lungul timpului o baza de date cu o multitudine de destinatii (spot-uri) de practicat kitesurfing din toata lumea. Pentru a veni in ajutorul altor practicanti de kitesurfing, dorim sa dezvoltam o aplicatie **web responsive** care sa le permita sa vizualizeze aceste locatii, sa le filtreze si sa afle mai multe informatii relevante despre acestea.

Avem nevoie de un **super inginer cu cunostinte de frontend (web),** **ca tine,** care sa ne ajute sa dezvoltam aceasta aplicatie folosind **Angular sau React (in functie de intership-ul ales)**. In caz ca vrei sa folosesti altceva, te rugam sa confirmi cu noi inainte.

# **Detalii de implementare**

## Aplicatia web se va conecta la un backend existent prin intermediul unui API. Gasesti detalii despre modul in care vei putea interactiona cu acesta in sectiunea API.

## Utilizatorii se vor putea loga in aplicatie folosind email si parola (LOGIN PAGE).

## Apoi vor putea vedea un ecran (DASHBOARD) unde vor fi afisate toate locatiile de kitesurfing din lume (informatie de la API) pe:

## Harta cu pinuri pentru toate spoturile de kitesurfing (sugestii librarii care sa te ajute: LeafletJS, Google Maps, Mapbox)

## Tabel cu toate locatiile si informatii despre ele: nume, latitudine, longitudine, tara, wind probability, luna cand e sezon de dat.

## Pe dashboard, va exista un buton filtru, cand utilizatorii il vor apasa, vor putea filtra informatia dupa: country si wind probability.

## Cand un utilizator va da click pe o locatie pe harta, i se va deschide un popup (DETAILS PAGE), unde vor fi afisate mai multe informatii despre locatia respectiva.

## Din DETAILS PAGE, utilizatorul va putea sa apese pe butonul **Add to favorites**, pentru a adauga spotul la favorite. Pentru un spot deja adaugat la favorite, utilizatorul va avea optiunea de a il scoate din lista de favorite (**Remove from favorites**).

## Pe harta, dar si in lista, spot-urile favorite vor fi marcate special, cu pin-uri galbene.

## In Dashboard, in dreapta sus, va fi un buton de profil care va afisa un meniu cu o singura actiune: Logout.

## In sectiunea **Design** de mai jos am detaliat design-ul **orientativ** al aplicatiei web.

## Alte functionalitai ale aplicatiei (alege minim 2)

## Lista de spoturi cu search si sortare pentru fiecare coloana

## Responsive Design (aplicatia arata bine si pe web si pe mobile)

## Login si Signup

## Functionalitate de adauga un spot nou (nume, select2 pentru tara, perioada cand e vant bun cu start date si end date, folosind un datepicker). Gasesti design in arhiva care vine impreuna cu enuntul problemei.

## **Recomandari**

Gandeste structura si codul aplicatiei ca si cand urmeaza sa dezvolti la ea timp de 2 ani, va avea zeci de ecrane, va folosi zeci de API-uri diferite si vor lucra diversi programatori pe codul sursa.

Vrem sa scrii cod de care esti foarte mandra/mandru si nu crezi ca mai poate fi imbunatatit :)

## **Instructiuni de upload**

Te rugam sa lucrezi sub un repository **private** de git pe platforma gitlab.com, sa faci commit-uri granulare si dese. Cand ai terminat, te rugam sa inviti pe gitlab userul @AttaSystems la repository-ul tau si sa oferi un acces de tip “Developer”. De asemenea, te rugam sa ne dai si un email in care sa ne anunti ca putem face review.

## **Ce vom urmari**

1. Claritatea codului: cat de ordonat este scris, cat de usor e de inteles
2. Cum aplici principiile OOP si cum structurezi aplicatia
3. Cum implementezi partea de networking (interactiunea cu API)
4. Cum tratezi erorile
5. Cum te asiguri ca interfata este responsive, nu ingheata cand aplicatia comunica cu serverul

# **Detalii API**

BASE URL: [**https://5ddbb358041ac10014de140b.mockapi.io/**](https://5ddbb358041ac10014de140b.mockapi.io/)

Pentru a afla URL-ul pentru fiecare endpoint, e nevoie de concatenarea “BASE URL” cu sufixul asociat fiecarui endpoint.

Exemplu: Pentru “lista de spoturi” se va face un GET Request la URL-ul [**https://5ddbb358041ac10014de140b.mockapi.io/spot**](https://5ddbb358041ac10014de140b.mockapi.io/spot)

**Este obligatoriu sa iti faci o clona a API-ului** deoarece mai este testat de catre alti candidati si are o limita de maxim 100 intrari. De aceea mai apare eroarea „Max number of elements reached for this resource!”. Pentru evitare poti folosi clona ta a API-ului unde poti controla cate intrari ai pentru fiecare endpoint in parte (maxim 100).

Te rugam **sa iti faci o clona la API:** odata ce iti creezi un cont pe [www.mockapi.io](http://www.mockapi.io/) folosesti acest link <https://www.mockapi.io/clone/5ddbb358041ac10014de140c> intr-un browser si da un refresh la pagina. Limita per cont demo este de 1 proiect, asa ca ai grija sa nu clonezi api-ul de mai multe ori.

Daca extinzi functionalitatea lui (adaugi alte entitati sau schimbi formatul API), te rugam sa ne anunti ca sa stim cand iti evaluam rezolvarea.

## **Endpoint-uri API**

API-ul este facut cu un serviciu de Mock API care intoarce rezultate fictive/generate. De aceea la adaugarea unei entitati noi, sa nu te astepti sa gasesti fix acele informatii (vor fi unele generate de serviciu) cand incerci sa citesti datele.

* **Login** 
  + - **POST /login –** simuleaza login (permite autentificarea cu orice username/password ☺), intoarce “userId” pentru utilizatorul autentificat care sa fie folosit ulterior pentru a incarca datele utilizatorului curent
* **User**
  + - **GET /user –** Lista utilizatorilor din system
    - **GET /user/:id –** Informatiile pentru userul cu id-ul “:id”
    - **POST /user –** Adauga un utilizator nou
    - **PUT /user/:id –** Modifica datele pentru un utilizator
    - **DELETE /user/:id –** Sterge un utilizator
* **Spot** 
  + - **GET /spot –** Lista spoturilor
    - **GET /spot/:id –** Informatii pentru spotul cu id-ul “:id”
    - **POST /spot –** Adauga un spot nou
    - **PUT /spot/:id –** Modifica datele pentru un spot
    - **DELETE /spot/:id –** Sterge un spot
* **Favourites**
  + - **GET /favourites –** Listeaza spot-urile favorite pentru utilizatorul curent
    - **GET /favourites/:id –** Informatii pentru entitatea favorites cu id-ul “:id”, fiecare favorites contine id-ul pentru spot-ul care este adaugat in lista
    - **POST /favourites –** Adauga o intrare noua pentru un spot favorit
    - **PUT /favourites/:id –** Modifica datele
    - **DELETE /favourites/:id –** Sterge un spot favorit

# **Testare**

Impreuna cu enuntul problemei, vei primi o arhiva, unde va exista un fisier: **Kitesurf.postman\_collection.json**

Pentru testare, instaleaza gratis aplicatia Postman  ([https://www.getpostman.com](https://www.getpostman.com/)) pe calculator si importa fisierul de mai sus. Odata ce acest lucru e facut, in Postman veti putea vedea toate endpoint-urile configurate sub colectia cu numele “Kitesurf”, in partea din stanga sus.

**Daca intampini orice problema in a folosi API-ul, te rugam sa ne scrii. Acesta este un mock API facut doar pentru aceasta problema si este posibil sa mai aiba anumite aspecte omise.**

# **Design**

Impreuna cu enuntul problemei, vei primi o arhiva, unde vei gasi in folderul Screenshots screenshot-uri cu ecranele aplicatiei, un folder assets, unde vei gasi elementele grafice necesare implementarii aplicatiei si un fisier .sketch cu tot designul aplicatiei in Sketch (https://www.sketchapp.com/). Mai jos poti vedea cum arata ecranele.

**Design-ul este orientativ, iar tu ai libertate sa schimbi ce consideri ca sa faci aplicatia sa fie mai accessibila dpdv UI/UX.**







