

UNIVERSITATEA “ALEXANDRU IOAN CUZA” DIN IAȘI
FACULTATEA DE INFORMATICĂ



LUCRARE DE LICENȚĂ

„AskME”

propusă de

DANIEL URSE

Sesiunea: 07, 2019

Coordonator științific

Drd. Colab. Florin Olariu

UNIVERSITATEA “ALEXANDRU IOAN CUZA” DIN IAȘI
FACULTATEA DE INFORMATICĂ

„AskME”

DANIEL URSE

Sesiunea: 07, 2019

Coordonator științific

Drd. Colab. FLORIN OLARIU

DECLARAȚIE DE CONSIMȚĂMÂNT

Prin prezenta declar că sunt de acord ca Lucrarea de licență cu titlul „*AskME*”, codul sursă al programelor și celelalte conținuturi (grafice, multimedia, date de test etc.) care însoțesc această lucrare să fie utilizate în cadrul Facultății de Informatică.

De asemenea, sunt de acord ca Facultatea de Informatică de la Universitatea „Alexandru Ioan Cuza” din Iași, să utilizeze, modifice, reproducă și să distribuie în scopuri necomerciale programele-calculator, format executabil și sursă, realizate de mine în cadrul prezentei lucrări de licență.

Iași, *data*

Absolvent *Daniel Urse*

Cuprins

Introducere	1
Istoric.....	1
Tema	3
Motivație	3
Gradul de noutate al temei	4
Obiective generale.....	4
Metodologie.....	5
Descrierea sumară a soluției	6
Structura lucrării	7
Contribuții.....	8
Capitolul 1. Descrierea problemei	9
Capitolul 2. Abordări anterioare.....	10
Upwork.com	10
Toptal.com.....	10
Fiverr.com	11
Concluzie	11
Capitolul 3. Descrierea soluției	12
Node.js	12
Express.js.....	13
Web sockets.....	13
MongoDB	13
JWT.....	13
Angular.....	13
Prezentare.....	14

Dashboard.....	15
Antet.....	16
Notificări	18
Adăugare întrebare (funcționalitate client)	19
Alegere freelancer (funcționalitate client).....	19
Profil utilizator.....	21
Stream video	22
Istoric plăți/încasări.....	24
Statistici vizionări-freelancer	25
Concluziile lucrării.....	27
Opinie personală	27
Direcții viitoare.....	27
Bibliografie	29
Anexa 1 – Node.js.....	30
Anexa 2 – Express.js	33
Anexa 3 – Web Sockets	34
Anexa 4 - MongoDB	36
Anexa 5 - JWT	37
Anexa 6 - Angular.....	39

Introducere

Istoric

Termenul de „freelancer” este folosit din ce în ce mai mult în zilele noastre. Inițial, tindem să credem că acest termen datează abia din perioada apariției internetului, dar surprinzător, el a fost utilizat cu mult înaintea acestuia.

Mulți dintre noi, atunci când o persoană ne spune că este freelancer, adesea ne gândim că acesta ar fi programator / designer / grafician dar, spre surprinderea noastră, domeniile cuprinse sunt mult mai vaste. „Freelance” este un termen destul de vechi, chiar mai vechi decât internetul.

A apărut inițial în *”The Life and Times of Hugh Miller by Thomas N. Brown – 1809”*. La acea vreme erau două categorii de muncitori: cei plătiți de rege în funcție de cât credea acesta de cuviință și soldații mercenari, care își vindeau propriile servicii. Aceștia din urmă erau considerați a fi „freelancers”. Cu timpul, acest termen a fost apreciat și folosit în domeniul afacerilor până a devenit o normalitate.

Forța de lucru din partea freelancerilor a devenit substanțială în SUA. Aceasta a crescut de 3 ori din anul 2014 până în prezent¹, ajungând până la cifra de 50 milioane de muncitori independenți la acea dată. Acest lucru poate fi observat în graficul de mai jos (Figura 1), realizat de *Bureau of Labor Statistics @ 2019*.

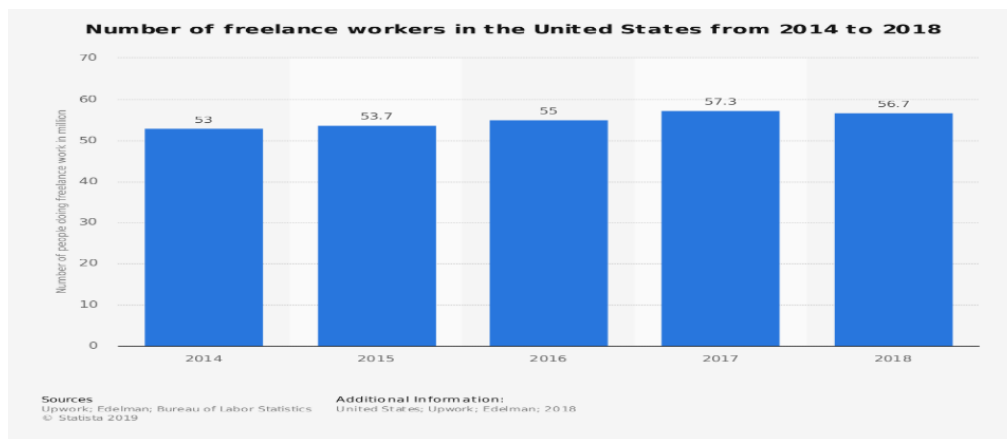


Fig 1

Mulți muncitori preferă astăzi acest tip de job „liber-profesionist” în detrimentul muncii în cadrul unei companii, din mai multe cauze:

- Varietate în cadrul domeniului de lucru

¹ <https://www.rent-acoder.com/blog/history-of-freelancing/29>.

- Lucru multiplu pentru diverși clienți, simultan
- Diversitate financiară în funcție de serviciile prestate
- Program flexibil și zile libere planificate personal
- Impuls creativ și libertate creativă
- Lucru de acasă

S-a dovedit în ultimii ani că numărul persoanelor freelance și domeniile activate de aceștia au crescut exponențial odată cu evoluția internetului și a tehnologiei: smartphones, tablete, calculatoare, etc. Făcând o paralelă cu freelancerii din anii 1800 – mercenarii – aceștia foloseau uneltele proprii (armele); acum uneltele folosite sunt dispozitivele, care ajută atât la oferirea serviciilor dar și la promovarea acestora, fiind folosite și ca mijloc de comunicare.

Cuvântul „*freelance*” a suferit trei modificări până a ajunge în forma actuală, formă care a fost folosită încă din anul 1720 până în prezent. În graficul de mai jos se poate vedea această evoluție: „*free lance*”, „*free-lance*”, „*freelance*”. După cum se poate observa, varianta finală și cea folosită astăzi este „*freelance*”² (Figura 2).

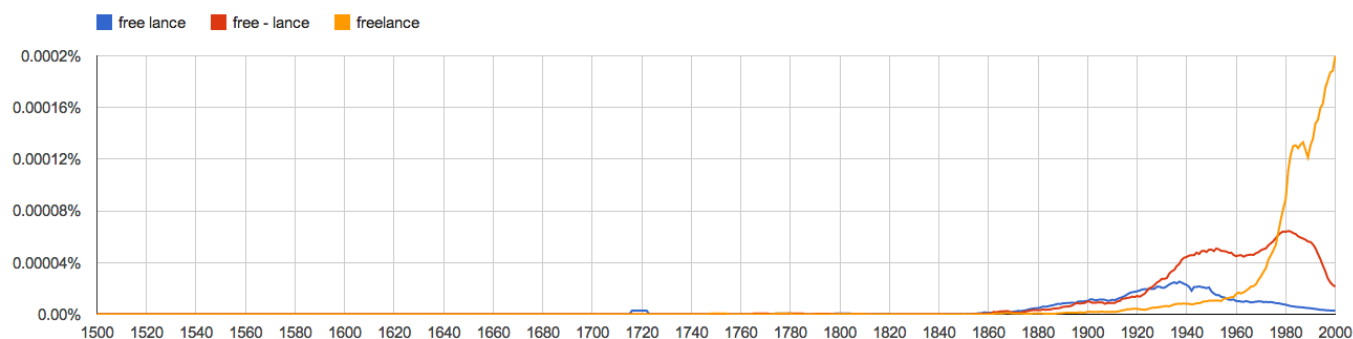


Fig 2

Prima platformă online dedicată persoanelor liber-profesioniste a apărut în anul 1999 și se numea *Elance*. Surprinzător, aceasta este prezentă și acum în mediul online și este una dintre cele mai mari platforme de acest gen din mediul virtual, activând însă astăzi sub numele de *Upwork*.

Apariția acestora a încurajat înființarea multor altor website-uri, predominant după anul 2007, exemple concludente fiind *Freelance.com* și *Fiverr.com*. Din acel moment, numărul de

² <https://medium.com/@patterncapturer/the-origin-of-the-word-freelance-and-why-it-should-make-us-happy-84e46a206348>.

joburi din mediul online a crescut foarte mult și, treptat, companiile au început să colaboreze cu freelanceri.

Astăzi există peste 1000 de portaluri de acest gen pe internet, acest lucru facilitând competiția în industrie atât pentru muncitori cât și pentru website-uri, oferind o mare dinamică și un interes cât mai diversificat pentru domeniul de activitate.

Freelancerii pot căuta platforma care li se potrivește cel mai bine, atât din punct de vedere financiar, cât și a domeniului de lucru sau a clientelei-țintă. Aspectul cel mai important care vizează interesele unui freelancer este acela de a fi comisionat cât mai puțin de către platforma pe care activează – comision pe care aceasta îl încasează pentru munca desfășurată de către freelancer. De aceea, platforma *AskME* se vrea a fi o platformă având comisioane cât mai mici și fără restricții, permițând fiecărui individ să își stabilească prețul personal pentru serviciile oferite.

Avantajul dezvoltării de aplicație web pentru freelanceri constă în posibilitatea utilizării acesteia și de pe orice smartphone ce dispune de o cameră video și microfon. Astfel, utilizatorii pot accesa de oriunde și oricând platforma pentru a fi în permanență „conectați” la tot procesul dintre client și furnizor, dar și de notificări în timp real sau de noi cereri de parteneriat.

Tema

Tema aleasă reprezintă o aplicație web care se adresează persoanelor interesate de acest tip de muncă - freelance. Scopul se focusează în egală măsură asupra clientului cât și asupra furnizorului, pentru a facilita utilizarea platformei fără restricții. Această platformă are ca obiectiv comunicarea în timp cât mai rapid dintre cele două entități: client și furnizor.

Motivație

Fiind o persoană care a lucrat ca freelancer, dar și prin prisma discuțiilor cu diferite persoane de interes, am constatat în domeniul acesta lipsa facilității de a oferi anumite servicii prin intermediul *video-streamingului*. De aceea, platforma are ca obiectiv dezvoltarea acestei facilități în viitor, pentru a putea fi folosită chiar și de către persoanele care momentan nu s-au gândit la munca de acest gen, întrucât domeniile regăsite în '*AskME*' sunt numeric superioare față de portalurile actuale de freelance.

Acestea nu sunt doar domenii uzuale, de pildă: IT, design, video, audio, etc. Datorită posibilității comunicării video în timp real, aria se extinde la multe alte domenii, cum ar fi: matematică, automobile, lingvistică, home-care, ș.a.m.d.

Gradul de noutate al temei

Tema, așa cum s-a expus în ceea ce privește motivația, își dorește a oferi o perfecționare calitativă în rândul aplicațiilor pentru freelanceri. Astfel, diferite grade de inovație sunt prezente în rândul mai multor aspecte, cum ar fi:

1. Tehnologiile utilizate
2. Soluția finală
3. Arhitectura folosită

1) Tehnologiile utilizate sunt cele mai noi și mai stabile din industrie, astfel AskME rulează și se bazează pe cele mai calitative limbaje de programare și de stocare a datelor pentru a oferi o experiență cât mai bună, atât din punct de vedere al utilizatorilor dar și din perspectiva dezvoltării aplicației în viitor. Tehnologiile utilizate sunt prezentate în capitolul 2 din lucrare.

2) Soluția finală vine cu o nouă metodă de a face freelance în mediul online, această metodă fiind video streaming-ul. Aplicația iese din tiparul obișnuit al lucrului prin chat sau mesagerie și își dorește dezvoltarea și extinderea ariilor de lucru sau a metodologiei de rezolvare a problemelor prin oferirea posibilității de explicare/implementare de idei și nu numai prin ajutorul conversațiilor live. Inovarea platformelor de freelance este principalul obiectiv al acestei aplicații, iar toate metodele de implementare sunt centralizate în jurul ei pentru rezolvarea soluției finale.

3) Arhitectura folosită este „*single page application*” și oferă o experiență UI/UX foarte plăcută datorită elementelor rapide care sunt afișate utilizatorilor. Din punct de vedere al arhitecturii de sistem, aceasta este “*MEAN*” bazată pe Javascript, unul dintre cele mai puternice și folosite limbaje de programare, iar aplicațiile de tip MEAN sunt foarte des întâlnite în ultimii ani. Implementarea aplicațiilor de acest tip oferă o siguranță și o garanție pentru aplicațiile care folosesc multe elemente live și de aceea AskME folosește această arhitectură.

Obiective generale

AskME dorește să crească nivelul de practicabilitate al aplicațiilor online de freelancing prin următoarele obiective:

- Design simplu și modern
- Viteză foarte mare de procesare a informațiilor
- Stream video rapid și fără întreruperi
- Calitate video

- Statistici detaliate pentru freelanceri
- Notificări în timp real
- Filtrări de utilizatori conectați în funcție de scorul lor oferit de gradul de interes și de cunoștințele necesare
- Ajutarea noilor utilizatori prin modalități de promovare
- Eliminarea timpilor pierduți pentru aflarea soluțiilor anumitor probleme întâmpinate de clienți

Astfel, pentru a putea implementa toate aceste obiective este necesară o arhitectură de sistem foarte bine pusă la punct, dar în același timp și de o calitate bună a codului sursă scris pentru a nu exista repetiții de cod sau anumite funcționalități care pot încetini aplicația și serverul.

Metodologie

Pentru a alege tema și aplicabilitatea ei în zilele noastre am studiat timp de 2 săptămâni piața existentă legată de aplicații freelance. Am prezentat acest studiu în primul capitol din lucrare, unde exemplific problemele și avantajele acestei teme.

Odată aleasă tema și obiectivele ei, am căutat cea mai bună arhitectură de implementare pe baza necesităților de implementare și, după o analiză a diferite articole sau aplicații existente, am ales arhitectura “*MEAN*”. Această arhitectură se potrivește foarte bine pentru aplicațiile cu acțiuni și evenimente în timp real.

În momentul începerii de implementare al aplicației am stabilit că trebuie folosit un sistem de versionare al întregului proiect încă de la început pentru a avea un istoric și un ansamblu al evoluției aplicației. Astfel, am decis folosirea “github.com” pentru aplicația AskME. Pe repository-ul specific se regăsesc toate fișierele aplicației, pașii dezvoltării ei dar și lucrarea de licență scrisă.

Înainte de a începe implementarea am creat structura bazelor de date, a serverului de backend, dar și a modulelor din frontend, pentru a ști foarte bine ce pași trebuie să urmăresc pentru a duce aplicația într-un punct final. După finalizarea acestei structuri am început procesul de implementare și am urmat firul structurii, lucrând în paralel la frontend, backend și baze de date pentru a nu neglija o componentă dintre acestea în detrimentul alteia.

Pe parcursul implementării au fost alocate resurse și timp pentru cercetare de anumite tehnologii sau de cazuri de implementare, în special la WebSockets și la video stream pentru a

realiza evenimente live și instantanee. Aplicația a suferit câteva modificări în ceea ce ține de codul scris în momentul în care am realizat că pot fi îmbunătățite sau rescrise anumite funcționalități.

Când am știut exact care este finalitatea aplicației și ce se dorește a fi rezolvat am început lucrarea scrisă deoarece aveam la dispoziție toate informațiile de care aveam nevoie despre AskME.

Descrierea sumară a soluției

AskME oferă posibilitatea utilizatorilor să interacționeze prin stream video cu freelanceri din diferite domenii de activitate. Acest mod de interacțiune este contorizat și monitorizat de către aplicație pentru a crea o transparență dar și pentru a avea un istoric concret al acțiunilor.

Clienții pot vedea istoricul conversațiilor și plăților efectuate prin grafice care indică acest lucru în fiecare lună a anului. Freelancerii pot vedea ce încasări au avut pe tot parcursul unui an și pot vedea prin rapoarte informative clienții care le-au vizitat profilul în ultima săptămână, plus locația de unde s-a făcut vizionarea.

Soluția finală presupune ca aceste funcționalități să fie prezente și să execute fără probleme.

Aplicația dorește a servi toate uneltele necesare unui freelancer și unui client pentru a face cât mai simplă și utilă comunicarea dintre aceștia prin intermediul unei interfețe ușor de folosit. După un studiu asupra website-urilor concurente am regăsit câteva probleme pe care *AskME* dorește să le elimine. Câteva din problemele găsite:

- lipsa actualizării în timp real a notificărilor și a mesajelor, fără necesitatea reactualizării platformei în cauză;
- pagini statice – lipsa unei interfețe plăcute și dinamice, care să-i ofere utilizatorului o experiență plăcută de utilizare, astfel mulți dintre posibili clienți sau furnizori devenind reticenți față de aplicație încă din primele minute de utilizare;
- munca depusă de furnizori sau chiar calitatea clienților nu este afișată într-un mod transparent față de ambele părți sau chiar față de niciuna – această problemă este îndepărtată prin intermediul unor recenzii date atât de către client, cât și de către furnizor;

- durată mare de timp de așteptare din partea clientului pentru a intra în contact cu furnizorul, soluție care este depășită prin conexiune directă instantanee cu acesta sub formă de notificare de conexiune.
- clienții nu pot vedea în timp real furnizorii care sunt conectați în acel moment pentru a putea începe o discuție instantaneu, fără a mai aștepta ca freelancerul să se conecteze în aplicație.

Prin prisma celor enumerate mai sus, mi-am propus ca aceste probleme să fie rezolvate, oferind o soluție cât mai bună pentru toate persoanele interesate de acest business.

Structura lucrării

Primul capitol cuprinde descrierea problemei alese în cadrul lucrării de licență și metoda de a o rezolva.

Al doilea capitol cuprinde o analiză asupra platformelor existente pe internet. Analiza are ca scop stabilirea elementelor necesare realizării aplicației AskME dar și a îmbunătățirii domeniului freelance.

În capitolul 3 voi enumera sumar principalele tehnologii folosite pentru realizarea aplicației, aducând argumente și explicații pentru a motiva folosirea acestora în detrimentul altor tehnologii. Acesta cuprinde și prezentarea platformei din perspectiva fiecărui tip de utilizator pentru a examina rezultatul final și a analiza în detaliu fiecare funcționalitate prezentă.

Contribuții

Contribuțiile aduse lucrării sunt atât în domeniul specificațiilor tehnice, cât și în ceea ce privește lucrarea și finalizarea implementării necesare aplicației dar și a documentației scrise.

Pentru atingerea scopului final, și anume de a finaliza o aplicație de freelance funcțională, am studiat și ales cele mai bune tehnologii atât din punct de vedere funcțional, cât și din punct de vedere al noutății acestora și al posibilelor actualizări pe termen lung primite. Astfel, aplicația poate fi dezvoltată în viitor fără nicio problemă.

Funcționarea aplicației în starea finală propusă în obiectivele generale a fost posibilă doar prin scrierea unui cod cât mai corect atât din punct de vedere funcțional, dar și din punct de vedere estetic. Am construit funcții și o logică simplă de înțeles și de urmărit pentru a putea cunoaște în permanență statusul fiecărei bucăți de cod.

În final, contribuțiile aduse sunt constituite de finalitatea obiectivelor generale, care au fost realizate într-un mod cât mai corect funcțional și tehnic.

Capitolul 1. Descrierea problemei

Problema ridicată este lipsa unui mediu online pentru freelanceri în care să fie posibilă comunicarea și rezolvarea rapidă de probleme. De asemenea, platformele tradiționale de freelance se axează toate pe aceleași domenii, și anume: design, IT, video, traduceri, ș.a.m.d.

Astfel, am analizat aceste aplicații (analiză prezentată în detaliu în capitolul 2), dar și nemulțumirile/lipsurile freelancerilor și am rezumat următoarele probleme existente:

- lipsa notificărilor și a mesajelor în timp real
- UI/UX complicat care nu satisface necesitățile clienților
- utilizatorii noi ajung foarte greu să primească primele comenzi
- lipsa transparenței dintre client/freelancer
- freelancerii nu cunosc istoricul clienților
- nu există platforme de freelance care să folosească video stream
- statistici foarte puține sau inexistente pentru a se menține o evidență clară asupra activității

Am decis că toate aceste probleme trebuie rezolvate prin intermediul unei aplicații care să abordeze un nou mod de lucru, printr-o interfață deosebită din punct de vedere estetic și ușor de utilizat. Astfel, problema care se ridică – în principal prin lipsa video streamingului – va fi rezolvată iar freelancerii care necesită o modalitate de lucru ușoară și rapidă pot beneficia de pe urma acestei rezolvări.

Ultima parte a problemei este reprezentată de lipsa unei modalități de avantajare a freelancerilor activi pentru a putea fi clasificați corect în funcție de rating, recenzii, timpi de răspuns și numărul accesărilor de profil. Acest aspect este des întâlnit în actualele aplicații de freelance și îmi doresc soluționarea sa.

Capitolul 2. Abordări anterioare

În mediul online există, în prezent, peste 1000 de aplicații destinate temei alese. Cu toate acestea, multe dintre ele oferă doar funcționalități de tip „anunț”, acestea fiind predominant portaluri de anunțuri pentru clienți și prestatori de servicii. În rândurile de mai jos am enumerat o listă de aplicații care au o anumită vechime în mediul online și care oferă mai multe opțiuni de utilizare pe care am dorit să le îmbunătățesc.

Upwork.com

În opinia mea, după analiza și utilizarea mai multor aplicații am ajuns la concluzia că aceasta se apropie cel mai mult de nevoile utilizatorilor. Având și o vechime considerabilă în industrie (de aproape 10 ani), Upwork are un număr foarte mare de clienți și furnizori.

Funcționalități regăsite:

- clientul creează anunțuri de angajare
- furnizorii aplică pentru aceste anunțuri oferind un CV sumar
- posibilitate de a plăti pe oră, în funcție de dorințele clientului
- aplicație de tip anunț-angajare
- design atrăgător

Cu toate că aplicația pare inițial ideală pentru utilizatori, din perspectiva freelancerilor nu este totul chiar atât de simplu și convenabil, deoarece clienții caută persoane foarte experimentate – întrucât majoritatea clientelei este constituită din companii, unele chiar foarte mari, care nu sunt de fapt în căutarea unui freelancer, ci mai degrabă a unui specialist care să lucreze remote.

Toptal.com

Aplicația garantează regăsirea unui minim de 3% din totalul de freelanceri din toată lumea, doar că nu este la fel de deschisă asupra lor precum celelalte platforme.

Pentru a fi freelancer pe Toptal trebuie să ai un nivel mare de experiență. Mai pe scurt, încă din primele rânduri ale formularului de înscriere este afișată necesitatea unei experiențe în domeniul de lucru destul de mari și a disponibilității de lucru de măcar 20 de ore pe săptămână (indicat ar fi 40 de ore).

Funcționalități regăsite:

- Design simplist

- Dificil de aplicat
- Calitatea utilizatorilor datorită experților prezenți în cadrul platformei
- Aplicație dedicată mai mult angajărilor remote decât angajării freelancerilor

Fiverr.com

Fiverr oferă un sistem puțin diferit față de cele 2 platforme menționate anterior, acesta fiind utilizat și în cadrul aplicației AskME. Sistemul menționat presupune schimbarea modului de angajare a unui freelancer, în sensul în care clientul nu lansează un anunț despre job-ul respectiv, ci clientul caută un freelancer care se încadrează în necesitățile sale – astfel, fiecare utilizator filtrează toți furnizorii pentru a-și găsi partenerul cel mai bun.

De asemenea, Fiverr oferă o gamă mult mai largă de domenii de activitate, iar nivelul diversității freelancerilor este ridicat datorită nivelului de experiență diferit al fiecăruia dintre ei.

Funcționalități găsite:

- Mesaje în timp real
- Design plăcut
- Ușor de utilizat
- Sistem de raportare a neclarităților
- Plăți securizate între client-furnizor

Concluzie

Aplicațiile curente au un sistem de comunicare destul de depășit, raportându-ne la posibilitățile actuale, lucru pe care îmi doresc să îl îmbunătățesc în aplicația AskME. Designul aplicațiilor este deseori neplăcut și neintuitiv atât pentru client, cât și pentru freelancer.

Nu există un alt sistem de freelancing prin intermediul streaming-ului video decât prin metoda clasică, mai exact cea prin mesaje cumpărător-furnizor. Majoritatea platformelor nu avantajează deloc freelancerii începători, care pot fi capabili în rezolvarea a destul de multe cereri ce nu necesită un nivel de cunoștințe foarte ridicat, ci poate chiar un nivel începător.

AskME dorește să diversifice concurența între freelanceri oferind fiecărui utilizator posibilitatea de a-și construi propriul profil în așa fel încât să atragă atenția și, în final, colaborarea cu posibili clienți, dar și setând un preț bazat în mod real pe capacitățile sale, precum și la nivel concurențial față de ceilalți utilizatori.

Capitolul 3. Descrierea soluției

Denumirea aplicației care se dorește a fi soluția finală este AskME. Aceasta este o platformă realizată pe arhitectura “*MEAN*”, arhitectură pe o voi prezenta atât în acest capitol cât și în anexele aferente.

AskME este formată dintr-un server Node.js care rulează în același timp atât fișierele frontend, cât și backend. Astfel, nu există decuplări sau probleme care necesită acces prin proxy sau rezolvare de erori CORS. Structura întregului sistem este prezentată în figura de mai jos³:

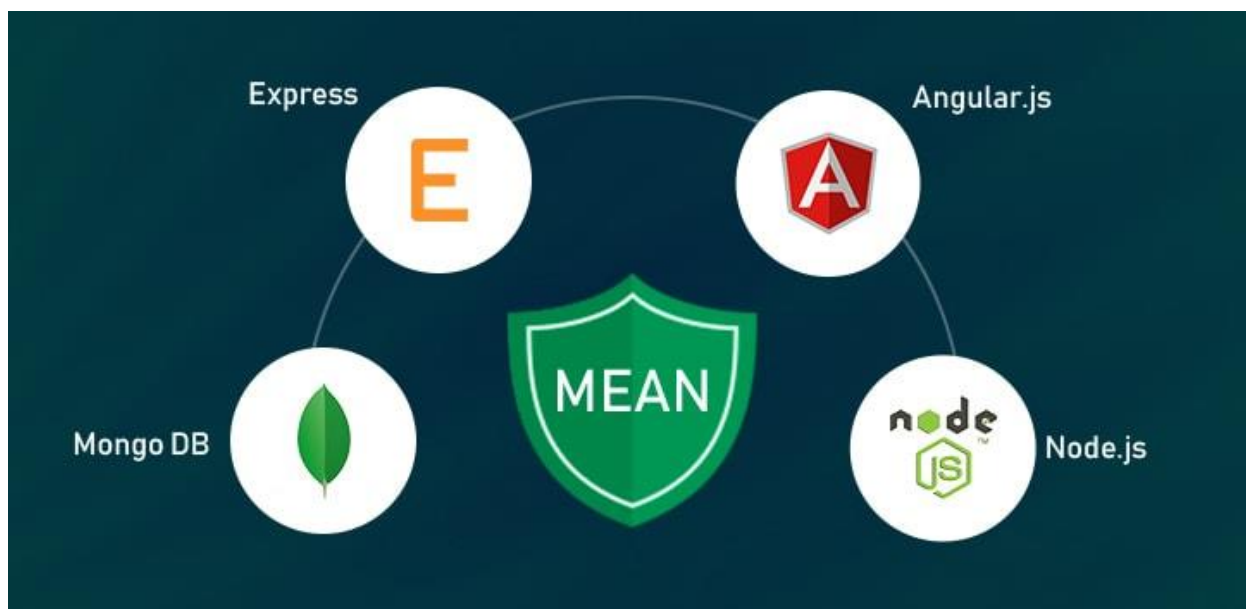


Fig 8

Fiecare componentă a sistemului contribuie la funcționarea aplicației în egală măsură și, astfel, alocarea timpului pentru implementarea fiecărei părți a fost egal. Angular este componenta ce ține de frontend, Express.js și Node.js sunt legate de backend iar MongoDB este baza de date de tip NoSQL aleasă în acest caz.

Node.js

Datorită temei aplicației AskME, și anume folosirea mai multor căi de comunicare continue și asincrone, am decis utilizarea acestei tehnologii pentru a se ocupa de toată structura arhitecturală ce ține de server. Aplicația rulează pe un server de Node.js care este activ în

³ <https://www.peerbits.com/blog/the-power-of-mean-stack-development.html>.

permanență și așteaptă noi cerințe și apeluri din diferite locații. (continuare anexa 1)

Express.js

Întrucât AskME se bazează pe servicii REST, am considerat că utilizarea acestui framework este necesară pentru administrarea conexiunii dintre frontend și Node.js. Serverul este servit de către Express.js și expus pe internet sub un port specific. La acea adresă pot fi accesate toate API de care aplicația are nevoie și fără de care nu poate funcționa. (continuare anexa 2)

Web sockets

Ceea ce este imperios necesar este o modalitate de a crea o conexiune persistentă, având latență scăzută și care să poată suporta tranzacțiile inițiate fie de client, fie de server. Asta este exact ceea ce oferă Web Sockets și tocmai de aceea sunt utilizate în cadrul aplicației AskME. (anexa 3)

MongoDB

Datorită metodologiei de administrare și de operare a datelor în mod dinamic, fără restricții, am ales utilizarea acestui tip de baze de date NoSQL, deoarece interacțiunea clienților diferă în funcție de tipul de utilizare individual al fiecăruia. Anumite entități care trebuie stocate în baza de date sunt diferite unele față de altele, iar din acest motiv este necesară utilizarea MongoDB. (anexa 4)

JWT

Pentru că AskME folosește date confidențiale, care nu trebuie expuse către alte entități, am considerat necesară utilizarea unei metode foarte sigure de securizare pentru întreaga arhitectură. Am ales JWT deoarece este cea mai folosită implementare, aceasta fiind utilizată chiar și de marile companii precum Google, Youtube, ș.a.m.d. (anexa 5)

Angular

Datorită unor numeroase actualizări ale frameworkului din 2016 până în prezent și a numărului mare de website-uri care folosesc această tehnologie, am considerat că aplicarea ei în cadrul AskME este benefică și poate beneficia de actualizări în viitor, în cazul continuării dezvoltării acesteia. (anexa 6)

Prezentare

Pentru a putea intra mai în detaliu în ceea ce privește arhitectura și cum funcționează fiecare tehnologie în parte, precum și în privința aportului acestora în cadrul aplicației, trebuie să observăm principalele atribuții și metode de lucru pe care AskME le oferă. Astfel, voi prezenta fiecare modul și funcționalitățile acestuia, precum și legătura dintre ele.

Modulul principal de la care pleacă toată logica este reprezentat de secțiunea de autentificare și înregistrare. Pentru a avea acces în aplicație și la toate funcționalitățile acesteia este necesară deținerea unui cont înregistrat în sistem. Acest lucru se poate face prin componentele de înregistrare și autentificare în care sunt prezente restricții pentru validarea informațiilor:

- parolă securizată, de minim 8 caractere, cu minim o majusculă și o cifră/caracter
- email unic și validat conform standardelor web
- numele și prenumele trebuie să nu conțină cifre
- username unic, care poate conține doar litere și cifre

Autentificarea/înregistrarea este validată de către server cu ajutorul JWT. În momentul în care un utilizator face o cerere de înregistrare, iar informațiile trimise de acesta corespund standardelor prezentate anterior, JWT verifică, ajutându-se de Express.js, dacă în MongoDB utilizatorul și emailul sunt unice. Dacă aceste condiții sunt îndeplinite se returnează statusul 200.

În acest moment, utilizatorul se poate autentifica trimițând numele de utilizator și noua parolă către server, moment în care JWT verifică dacă informațiile primite sunt valide: dacă username-ul existent și parola criptată sunt corespunzătoare, atunci se returnează statusul 200 iar în antetul răspunsului este atașat tokenul necesar pentru utilizarea aplicației și a tuturor serviciilor securizate de acesta.

Figura 9 este reprezentată de interfața de înregistrare iar figura 10 de cea de autentificare:

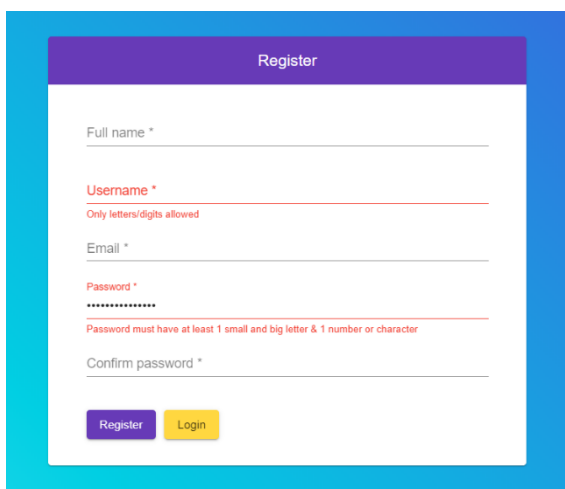

 A registration form titled "Register" with a purple header. It contains input fields for "Full name *", "Username *" (with a red error message "Only letters/digits allowed"), "Email *", "Password *" (with a red error message "Password must have at least 1 small and big letter & 1 number or character"), and "Confirm password *". At the bottom are "Register" and "Login" buttons.

Fig 9 Înregistrare

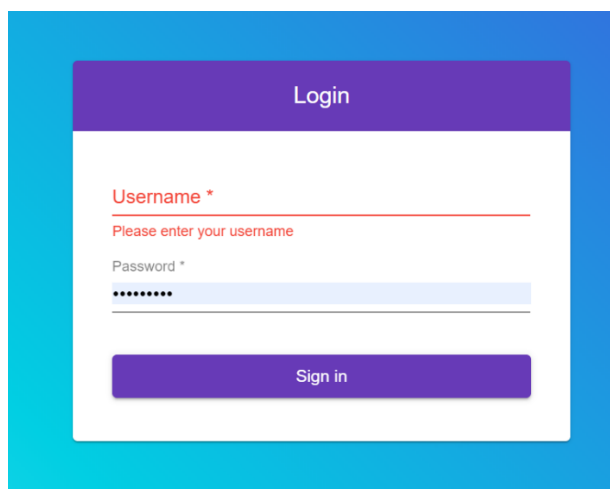

 A login form titled "Login" with a purple header. It contains input fields for "Username *" (with a red error message "Please enter your username") and "Password *". At the bottom is a "Sign in" button.

Fig 10 Autentificare

După ce am stabilit modulul care se ocupă cu tot ce ține de securitate, voi prezenta mai jos principalele secțiuni ale aplicației și funcționalitățile acestora.

Dashboard

Pagina principală oferă o descriere sumară a avantajelor și a beneficiilor utilizării AskME. Această pagină este disponibilă tuturor utilizatorilor deoarece scopul acesteia este de informare și de prezentare a atuurilor aplicației. În prima parte a paginii este prezentă sigla aplicației alături de un motto specific.

În partea inferioară sunt prezentate 3 principale calități oferite de aceasta:

- Smart & simple
- Find answers
- Get inspired

Această prezentare beneficiază de un design plăcut, alături de câteva animații simple pentru a produce un impact minimalist, dar memorabil și pentru a atrage atenția posibilor noi utilizatori. În figurile 10 și 11 de mai jos sunt afișate aceste secțiuni:

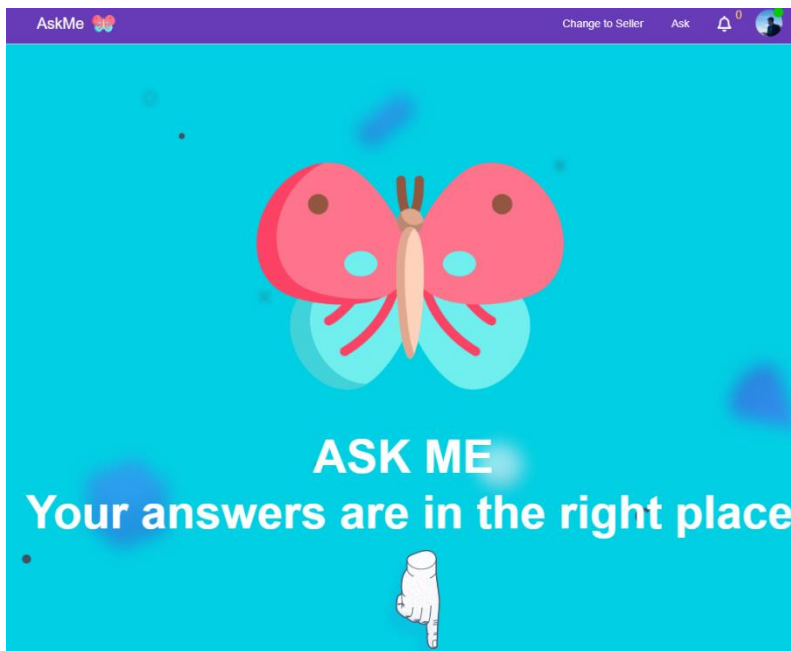


Fig 10

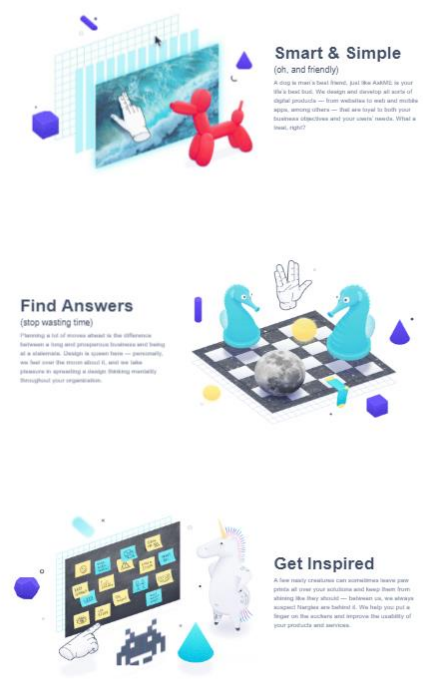


Fig 11

Antet

Fiind o aplicație destinată freelancerilor, AskME conține două tipuri de entități: seller și buyer. Aceste denumiri desemnează cele două tipuri de utilizatori ai platformei: clienți și freelanceri. Astfel, clientul are denumirea de seller iar freelancerul are denumirea de buyer.

Pentru a reprezenta oricare dintre tipurile de utilizator prezentate mai sus este necesară deținerea unui singur cont de utilizator. Astfel se micșorează logica bazei de date, a requesturilor dar și a complexității din backend/frontend. Acest lucru a fost posibil prin asocierea fiecărui cont al unui tip de utilizare, acest tip având denumirile specificate anterior.

Pentru schimbarea tipului fiecărui cont este necesară doar apăsarea unui buton din cadrul antetului, metodă afișată în figura 12 și figura 13.



Fig 12 – Antet pentru client



Fig 13 – Antet pentru freelancer

Prin apăsarea butonului „Change to Buyer”, respectiv „Change to seller” se schimbă tipul utilizatorului pentru client, actualizându-se instant modificarea și în baza de date. Astfel, toate

elementele grafice sunt schimbate, dar și drepturile asupra anumitor servicii sau pagini web sunt acceptate/restricționate în funcție de tipul fiecărui utilizator.

Fiecare utilizator are, în funcție de metoda de lucru aleasă, diferite opțiuni de navigare și de vizualizare a informațiilor. Pentru a vedea aceste opțiuni este necesar un click pe iconița corespunzătoare imaginii de profil. În figura 14 sunt prezentate opțiunile clientului, iar în figura 15 sunt cele ale freelancerului:

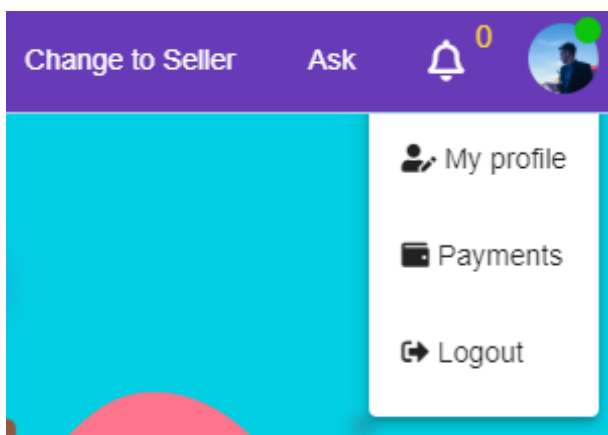


Fig 14 Opțiuni client

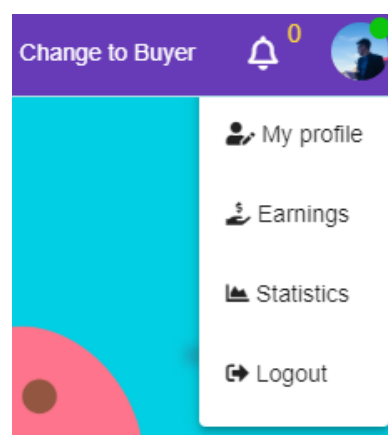


Fig 15 Opțiuni freelancer

Pentru a ieși din cont este necesar un click pe butonul “Logout”, funcționalitate identică indiferent de tipul de cont asociat. În momentul accesării acestei opțiuni, utilizatorul nu mai are statusul online, acest lucru fiind actualizat instant către toți utilizatorii aplicației în acel moment. Astfel, utilizatorul în cauză nu va mai apărea în căutările clienților deoarece nu mai este prezent în aplicație.

Acest lucru se realizează cu ajutorul WebSockets și al canalului de comunicare către toate persoanele autentificate în sistem. În tabelul 2 este prezentat socketul care se ocupă de acest lucru:

```

socket.on('disconnectNow', data => {
  User.findOne({username:
data}).then(user => {
    if (user) {
      user.online = false;
      users = users.filter(function
(obj) {
        return obj !== data;
      });
      user.save();
    }
  });
  socket.disconnect();
  io.emit('user disconnected');
}

```

Tabelul 2

Notificări

AskME are în componența sa un sistem de notificări în timp real, realizat prin WebSockets. Aceste notificări au scopul de a oferi atât clienților cât și freelancerilor mesaje, dar și un mic istoric al cererilor de conexiune acceptate sau respinse. În momentul în care un client dorește să intre în conversație cu un prestator de servicii, acesta, pe lângă notificarea de tip pop-up, primește și o notificare în antetul paginii care îi aduce la cunoștință acest lucru.

Notificările pot fi marcate individual ca fiind citite sau pot fi marcate toate în același timp. Totodată, ele pot fi șterse în cazul în care utilizatorul nu mai are nevoie de informațiile oferite de acestea. Mesajele sunt sugestive și vin alături de data și ora primirii lor. Notificările au un design plăcut și sunt disponibile pentru toate tipurile de utilizator.

În figura 16 se poate observa cadrul specific unei liste de notificări:

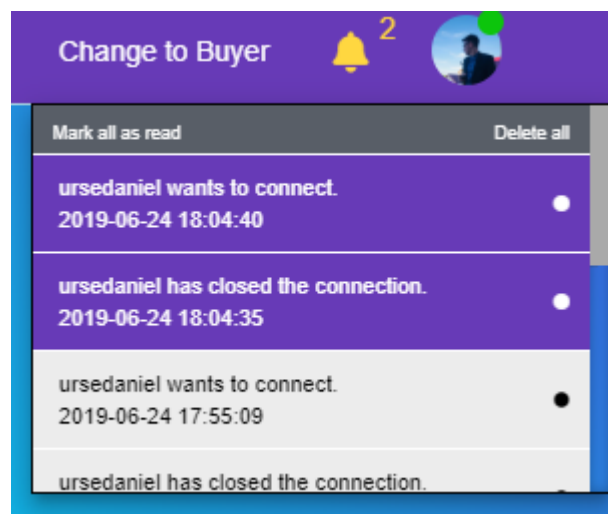


Fig 16

Adăugare întrebare (funcționalitate client)

Această pagină este vizibilă și accesibilă doar pentru clienți, freelancerii neavând permisiune către aceasta. Toate funcțiile și conexiunile dintre frontend și backend sunt securizate pentru a permite accesul exclusiv al cumpărătorilor. Această secțiune este compusă din 3 părți:

1. Alegerea categoriei serviciului
2. Expunerea pe scurt a necesității
3. Alegerea freelancerului potrivit

Categoriile sunt dispuse într-un mod simplist și intuitiv, sub formă de aliniere pe căsuțe (aliniere grid). În momentul în care categoria este aleasă, trebuie completată întrebarea care se dorește a fi adresată pentru ca freelancerul căruia îi este solicitată conexiunea să știe dinainte de a răspunde cererii dacă poate răspunde la ea.

Ajuns în pasul 3, clientul poate alege din lista de utilizatori conectați pe cel pe care îl consideră potrivit pentru a îi răspunde la eventuale nelămuriri. În figura 17 este prezentată interfața de alegere a categoriei:

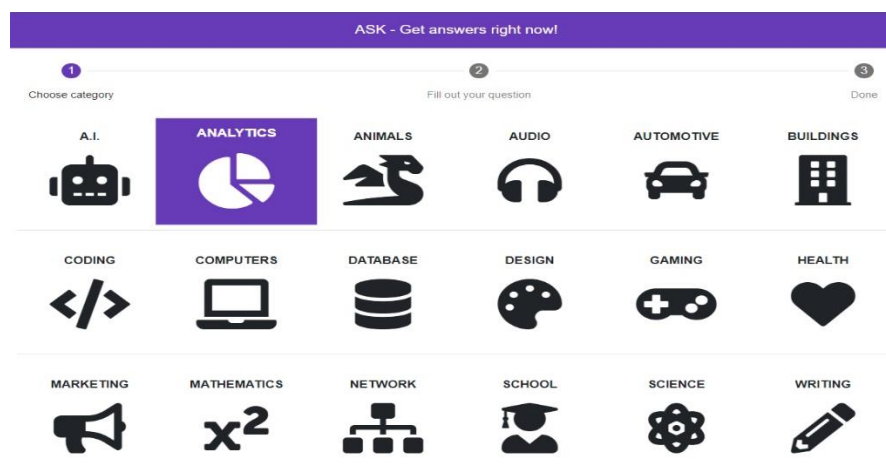


Fig 17

Alegere freelancer (funcționalitate client)

Odată ce categoria a fost aleasă și întrebarea a fost introdusă se ajunge la etapa de alegere a freelancerului potrivit. Aici apare o listă doar cu persoanele conectate în acel moment la aplicație și care au setat în cadrul profilului că doresc să răspundă la întrebările adresate în acea/acele categorii.

Pentru a face o diferențiere între freelancerii conectați am construit un algoritm în Node.js în scopul trierii acestora. Astfel, se calculează un scor în funcție de mai multe atribute, iar lista persoanelor disponibile este afișată descrescător în funcție de acesta. Scorul se calculează astfel:

- Rating
- Număr recenzii
- Accesări profil zilnice
- Media timpului de răspuns

Fiecare atribut poate avea o valoare de maxim 100 de puncte. Totodată, orice freelancer are șansa egală de a avea punctajul maxim. Acest punctaj se schimbă automat în funcție de acțiunile celorlalți utilizatori.

Ponderea de valoare a fiecărui atribut este calculat în felul următor:

- a) Rating – $(\text{nr. recenzii} / 5) * 100$. Fiecare utilizator poate avea un rating de maxim 5 puncte, astfel dacă ratingul are scorul de 5, punctajul final este de 100 de puncte.
- b) Număr recenzii – se obține punctajul în funcție de: $(\text{număr recenzii utilizator} / \text{media recenziilor tuturor utilizatorilor}) * 100$. Dacă ratingul acestuia depășește media recenziilor se obține punctajul de 100 de puncte.
- c) Accesări profil – fiecare freelancer poate primi pe zi maxim 2 vizionări per user, din 2 locații: tabelul de utilizatori conectați și accesarea profilului. Calculul atributului: 1 vizionare = 1 punct. Dacă se obține un scor mai mare de 100 de vizionări, punctajul obținut este de 100 de puncte.
- d) Media timpului de răspuns – Toți utilizatorii care au răspuns măcar la o întrebare au o medie a timpului de răspuns proprie: cu cât această medie este mai mică, cu atât mai bine. Calculul acestui atribut este efectuat astfel: $(\text{timpul de răspuns} / \text{media de răspuns a tuturor utilizatorilor}) * 100$. Dacă timpul de răspuns depășește media utilizatorilor se obține punctajul de 100 de puncte.

Astfel, algoritmul având aceste criterii per utilizator în permanență, filtrează mereu aceste informații și calculează un scor individual. Metoda de obținere a acestui scor:

$$T = 0.2\% \text{ punctaj recenzii} + 0.3\% \text{ punctaj rating} + 0.4\% \text{ timp răspuns} + 0.1\% \text{ accesări}$$

În urma acestui calcul, AskME oferă o modalitate de concurență și de diferențiere între cei mai activi și calitativi freelanceri conectați în momentul respectiv la aplicație. Astfel, persoanele liber-profesioniste sunt motivate să ofere răspunsuri cât mai bune și să fie cât mai active în cadrul platformei.

Pentru a afișa clienților care doresc să intre în discuție cu un freelancer sunt folosite WebSockets pe partea de backend pentru a ști în fiecare moment ce persoane sunt conectate în aplicație. Serverul este în permanentă legătură cu baza de date și cu frontend pentru a transmite informații în timp real.

În figura 18 este afișată interfața pentru vizualizarea persoanelor conectate în aplicație:

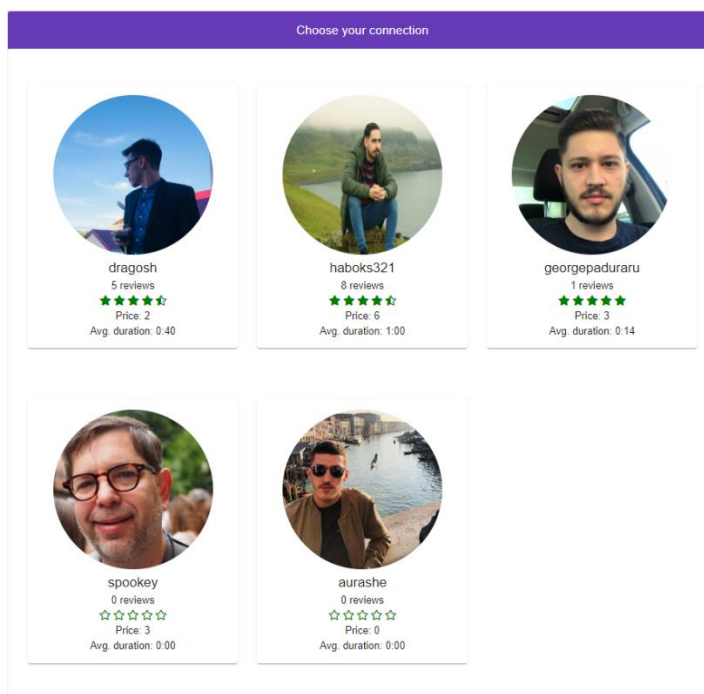


Fig 18

În această componentă se pot vedea pe scurt câteva elemente importante ale conturilor de utilizatori freelanceri: numele de cont, numărul de recenzii, ratingul, prețul pe minut și media de răspuns a convorbirilor. În momentul în care se apasă pe iconița unuia dintre utilizatori, clientul este redirecționat către profilul acestuia pentru a obține un raport mai în detaliu al informațiilor despre freelancer, precum și istoricul necesar.

Profil utilizator

Fiecare utilizator își poate schimba datele profilului, indiferent de tipul acestuia. Secțiunea conține informații personale despre client/freelancer, precum și recenziile pe care acesta le-a primit. În acest mod se oferă o transparență și un istoric care sunt utile tuturor utilizatorilor.

Informațiile care pot fi schimbate:

- Nume & prenume

- E-mail
- Parolă
- Preț pe minut (pentru freelancer)
- Descriere personală
- Categoriile de activare (pentru freelancer)

Lista de recenzii este prezentă în partea de jos a profilului și poate fi vizionată de orice utilizator. În momentul în care o persoană accesează un profil diferit de cel personal, persoanei accesate i se contorizează 1 vizionare, în scopul de a se monitoriza numărul total de vizualizări/profil al fiecărui utilizator.

Totodată, un buyer care este atras de detaliile profilului unui freelancer poate solicita o conexiune cu acesta apăsând click pe butonul din josul paginii. În figura 19 este prezentată secțiunea de profil:

The screenshot displays a user profile interface. At the top, there is a purple button labeled "Edit your profile". Below it is a circular profile picture of a man, followed by the text "8 reviews", a star rating of 4.5 (represented by 4 full green stars and 1 half-filled star), and a date "2019-12-25" with a clock icon and "Duration: 0:00".

The main profile section is divided into two parts. The left part, titled "Personal data", contains fields for "Full name" (filled with "Uria Daniel"), "Username" (filled with "ursedaniel"), "Price per minute" (filled with "8"), "Email" (filled with "daniel.ursa@hotmail.com"), and "Password" (masked with dots). Below these is a "Description" field (filled with "I am an experienced fullstack developer") and a "Categories" dropdown menu. An "Update profile" button is at the bottom. The right part, titled "List of reviews", shows a table of reviews.

Date	User	Rating	Review
2019-06-21 16:28:14	ursedaniel	★★★★★	good seller!
2019-06-21 16:26:26	ursedaniel	★★★★★	nice person
2019-06-21 16:25:58	ursedaniel	★★★★★	12312312
2019-06-21 15:29:01	ursedaniel	★★★★☆	very good
2019-06-21 13:16:40	ursedaniel	★★★★☆	nice customer

Below the reviews table is a purple "Connect" button.

Fig 19

Stream video

Atunci când un client dorește să apeleze un freelancer, va trimite o cerere către acesta sub formă de notificare și fereastră pop-up, în care freelancerul poate vedea categoria întrebării, utilizatorul care cere conexiunea dar și întrebarea propriu-zisă. Freelancerul nu poate face modificări în aplicație până ce nu confirmă sau anulează cererea de conexiune primită.

Figurile 20 și 21 de mai jos arată momentul în care clientul trimite cererea și așteaptă începerea conexiunii, iar freelancerul vede conexiunea și alege metoda de a răspunde:



Fig 20 – Așteptare conexiune

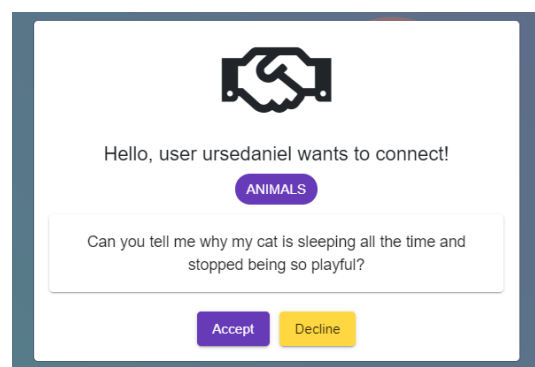


Fig 21 Opțiuni freelancer

Această conexiune, precum și sistemul de notificări sunt implementate prin WebSockets și se efectuează instant, fără a fi nevoie de reactualizarea aplicației. În cazul în care freelancerul anulează conexiunea, clientul primește o notificare în plus despre această acțiune și o fereastră modală în care îi este comunicat acest lucru.

Dacă conexiunea este acceptată, prestatorul de servicii este redirecționat în aceeași secțiune ca și clientul și este inițiat stream-ul video. Acest stream este temporizat din prima secundă de convorbire și este taxat conform prețului pe minut ales de către freelancer. Ambele persoane pot vedea în același timp aceste informații, dar doar clientul poate opri conexiunea atunci când a primit toate răspunsurile necesare. În cazul întreruperii de conexiune, taxarea se oprește și se așteaptă un timp de 15 secunde ca oricare dintre cei 2 utilizatori să se reconecteze.

Mai jos sunt prezentate cadrele celor 2 utilizatori și ce interacțiuni au aceștia (fig 22 & 23):

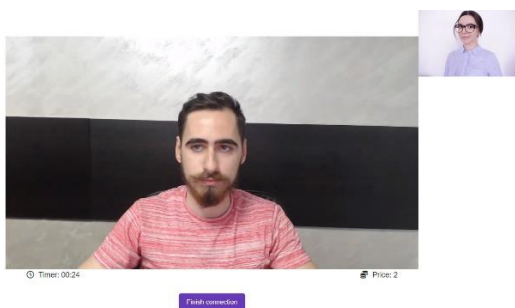


Fig 22 – Cadru client

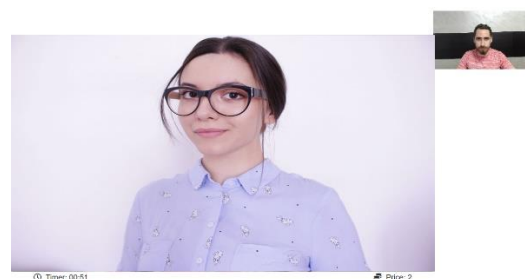


Fig 23 Cadru freelancer

În momentul în care o conversație a fost finalizată, ambii participanți sunt redirecționați către o nouă componentă în care, dacă doresc, pot să acorde o recenzie unul celuilalt. Prin această recenzie se oferă un rating de la 1 la 5 stele, alături de un scurt mesaj pentru a explica alegerea făcută.

Atât recenziile, cât și convorbirea sunt salvate în baza de date și sunt accesibile ambelor părți. Astfel, freelancerul poate vedea cât și de la cine a încasat, iar clientul către cine și cât a plătit. Calculele sunt transparente și sunt furnizate informații exacte legate de momentele plăților, oferindu-se acces permanent la acestea.

Istoric plăți/încasări

Atât clientul, cât și freelancerul au disponibile două secțiuni separate în care pot vedea informații legate de istoricul plăților sau al încasărilor. Prin această secțiune am dorit să pot oferi ambelor tipuri de utilizatori o metodă prin care să poată verifica oricând acțiunile pe care le-au realizat în cadrul aplicației, totodată putându-și administra veniturile/plățile.

În funcție de tipul de cont ales, denumirea și informațiile acestei componente se vor schimba automat. Ambele entități dispun de aceeași interfață grafică pentru această secțiune; diferențele țin de statisticile oferite și de datele primite.

Figura 24 reprezintă plățile accesibile clienților iar figura 25 încasările destinate freelancerilor:

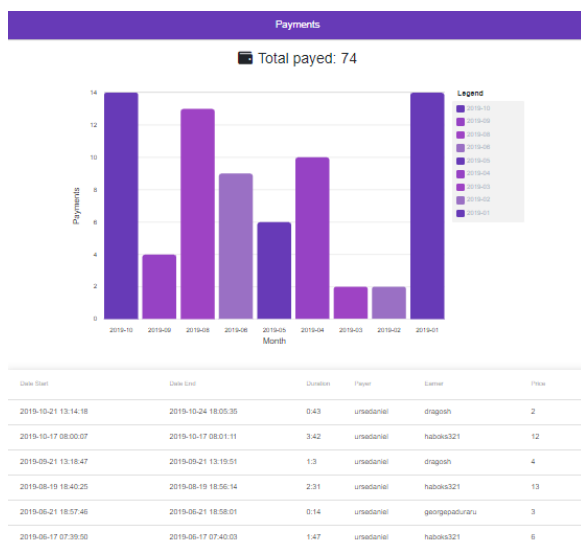


Fig 24 – Plăți client

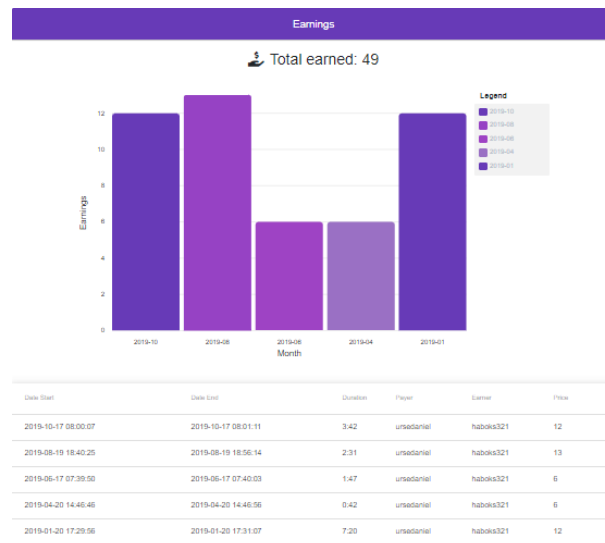


Fig 25 Încasări freelancer

Secțiunea conține 2 zone:

- Grafic ce cuprinde fiecare lună de activitate: fie plată, fie încasare. Fiecare lună are asociat totalul valorilor din cadrul acesteia. Acesta este actualizat automat după fiecare acțiune finalizată cu succes de către utilizator.
- Tabel în care sunt afișate toate informațiile conversațiilor. Ele cuprind numele clientului/freelancerului, data de început și final a conversației, durata acesteia, prețul total.

Designul ales este unul simplist și se dorește a fi un adjuvant în ceea ce privește eficiența oferirii și citirii informațiilor, în scopul de a facilita procesul de observare al tuturor tranzacțiilor făcute.

Statistici vizionări-freelancer

Utilizatorii care au tipul contului setat ca freelancer pot să vadă istoricul apariției acestora în cadrul aplicației. Statisticile sunt monitorizate pe parcursul fiecărei săptămâni. Prin această metodă, fiecare persoană poate vedea dacă profilul său are un impact pozitiv și stârnește interes în rândul clienților.

Se descriu două tipuri de vizionări: de profil și cele din lista utilizatorilor conectați. Fiecare client poate acorda zilnic maxim două vizionări unui utilizator: o dată dacă acest freelancer apare în lista de căutări a clientului și o dată dacă profilul acestuia este accesat și vizionat. Aceste vizionări sunt realizate automat de către aplicație cu WebSockets și nu interferează deloc cu experiența de utilizare a clienților.

Componenta de statistici oferă următoarele informații:

- a) Toate statisticile: atât statisticile de profil cât și de căutări
- b) Statistici de profil
- c) Statistici de căutare

Fiecare dintre aceste trei categorii conține două subsecțiuni: vizualizări din ziua curentă și vizionări din toată săptămâna. Astfel, fiind trei ramificații cu două subcategorii, se poate vedea în detaliu toată activitatea profilului freelancerului. Pe lângă graficele aferente sunt oferite și informații despre data, ora și utilizatorul care a făcut vizionarea respectivă.

Informația primită în această componentă se poate schimba oricând în funcție de interacțiunile profilului freelancerului cu acei clienți care sunt interesați de serviciile oferite de acesta. Această unealtă oferită dorește să aducă o imagine de ansamblu și mai în detaliu fiecărui

prestator de servicii, pentru a ști unde și cum este simțită prezența sa cel mai mult în cadrul aplicației AskME.

În figura 26 este prezentă statistica din ziua curentă, iar în figura 27 cea săptămânală a unui freelancer:

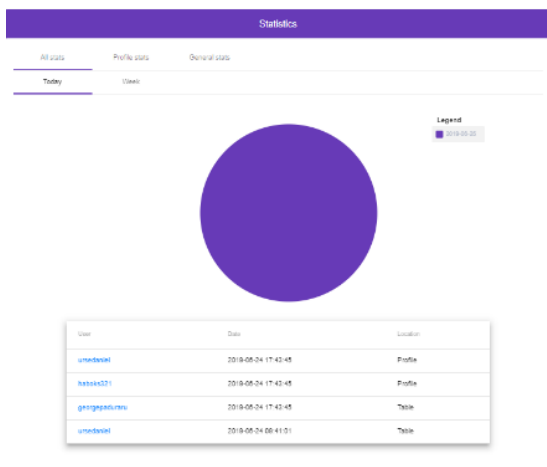


Fig 26 – Statistici zilnice

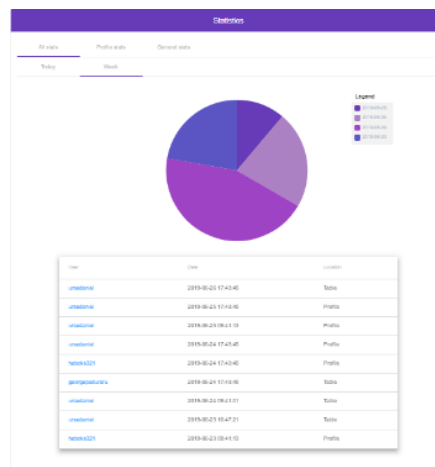


Fig 27 Statistici săptămânale

Concluziile lucrării

Prin aplicația AskME am încercat abordarea unui nou segment în domeniul platformelor online de freelancing, iar acest lucru a constat în introducerea unui sistem de conversație și de rezolvare a problemelor prin stream video.

Pe lângă arhitectura folosită, și anume “*MEAN*”, au fost utilizate anumite module Node.js sau API. Statisticile și integrarea video stream au fost implementate pe baza WebSockets și calitatea de transmitere video este de 1080p (la cea mai înaltă calitate). Integrarea client/server a beneficiat de două tipuri de comunicare – Rest API și WebSockets, ambele lucrând în paralel sau individual în funcție de necesitatea aplicației.

AskME oferă o soluție completă, bazată pe motivația aleasă, iar pentru acest lucru a fost nevoie de studii de caz, tehnologii și cele mai bune metode de implementare. Partea frontend este decuplată în totalitate de backend, iar acest lucru facilitează mentenanța ei în viitor sau a posibilelor îmbunătățiri care pot surveni.

Opinie personală

În opinia mea, din momentul finalizării lucrării pot constata că obiectivele și motivația aleasă au fost respectate, iar rezultatul final, aplicația AskME, oferă toate funcționalitățile care au fost propuse inițial.

Atât designul, cât și metoda de implementare și de arhitectură au fost concepute și finalizate aducând un plus de valoare față de aplicațiile actuale. Cel mai substanțial beneficiu adus industriei freelance este metoda de comunicare prin video stream.

Toate statisticile și informațiile oferite, care sunt ușor de modificat atât UI/UX, cât și ca funcționalitate, garantează buna funcționare a aplicației, de unde rezultă capacitatea acesteia de a începe să ruleze în orice moment, chiar de la finalul lucrării de licență.

Direcții viitoare

Consider că aplicația poate primi îmbunătățiri ce pot aduce mai multe beneficii utilizatorilor și comunității freelance. Câteva dintre acestea sunt:

- a) Implementare aplicație pentru platformele Android și iOS. Deoarece platforma AskME are un design responsive, conceput pentru toate dispozitivele, ea poate fi folosită în prezent și pe smartphones, accesând orice browser care se poate conecta la

aplicația web. Avantajul unei aplicații native reduce timpul de accesare al acesteia, informațiile pot fi oferite într-un mod specific pentru fiecare tip de smartphone și în același timp se pot extinde domeniile de utilizare datorită portabilității mult mai mare al unui telefon/tabletă în comparație cu calculatoare/laptopuri. Această implementare poate fi făcută cu ajutorul frameworkului *"Ionic"*.

- b) Chat live. Introducerea unui chat live între clienți și freelancer ar fi util în cazul în care clientul are o problemă foarte complexă sau multe informații de oferit și dorește să prezinte acest lucru prestatorilor de servicii pentru a ști dacă aceștia îi pot oferi informațiile sau soluțiile necesare pentru rezolvarea problemei sale.
- c) Filtre și sortări avansate pentru lista de freelanceri conectați.
- d) Machine learning folosit în managementul de acțiuni în cadrul întregii aplicații pentru a îmbunătăți timpii de rulare.

Bibliografie

- <https://www.rent-acoder.com/blog/history-of-freelancing/29>
- <https://medium.com/@patterncapturer/the-origin-of-the-word-freelance-and-why-it-should-make-us-happy-84e46a206348>
- <https://www.truelancer.com/blog/history-of-freelancing/>
- <https://codeburst.io/all-about-node-js-you-wanted-to-know-25f3374e0be7>
- <https://nixa.ca/blog/the-history-and-impact-of-nodejs/>
- <https://hackernoon.com/node-js-emerging-as-the-universal-development-framework-for-a-diversity-of-applications-c2e788290f5f>
- <https://medium.com/@LindaVivah/the-beginners-guide-understanding-node-js-express-js-fundamentals-e15493462be1>
- <https://www.impressico.com/2015/10/06/advantages-of-using-express-js/>
- <https://blog.teamtreehouse.com/an-introduction-to-websockets>
- <https://bytescout.com/blog/2014/09/mongodb-history-and-advantages.html>
- <https://www.xplenty.com/blog/the-sql-vs-nosql-difference/>
- <https://jwt.io/introduction/>
- <https://www.altexsoft.com/blog/engineering/the-good-and-the-bad-of-angular-development/>
- <https://www.rishabhsoft.com/blog/reasons-to-choose-angular-for-application-development>
- <https://www.peerbits.com/blog/the-power-of-mean-stack-development.html>
- <https://www.freepik.com/>
- <https://material.angular.io/>
- <https://www.agora.io/en/videocall/>

Anexa 1 – Node.js

Node.js este un runtime JavaScript, construit pe motorul V8 al browserului Chrome. Node.js folosește un model de I/O fără restricții, bazat pe evenimente, aspect care îl determină să fie ușor și eficient. “*npm*” este un pachet Node.js open source, care este cel mai mare din lume. Acesta beneficiază de o utilizare foarte mare din partea developerilor la nivel mondial și are mii de pachete care pot fi implementate în diferite aplicații în funcție de necesități. Aceste pachete pot fi folosite în cadrul oricărei aplicații care folosește Node.js.

Este bazat pe un eveniment asincron, care este conceput pentru a construi aplicații scalabile de rețea. Acesta poate gestiona mai multe conexiuni concurente la un moment dat, în cazul în care atunci când în cererea de conexiune sunt simultan mai multe apeluri pentru fiecare conexiune în parte, o cerere specială denumită “*callback*” este asignată. Dacă nu există nici o sarcină de efectuat, Node.js va sta în așteptare de cereri noi.

Mecanismul de utilizare se poate vedea în figura 3 de mai jos⁴:

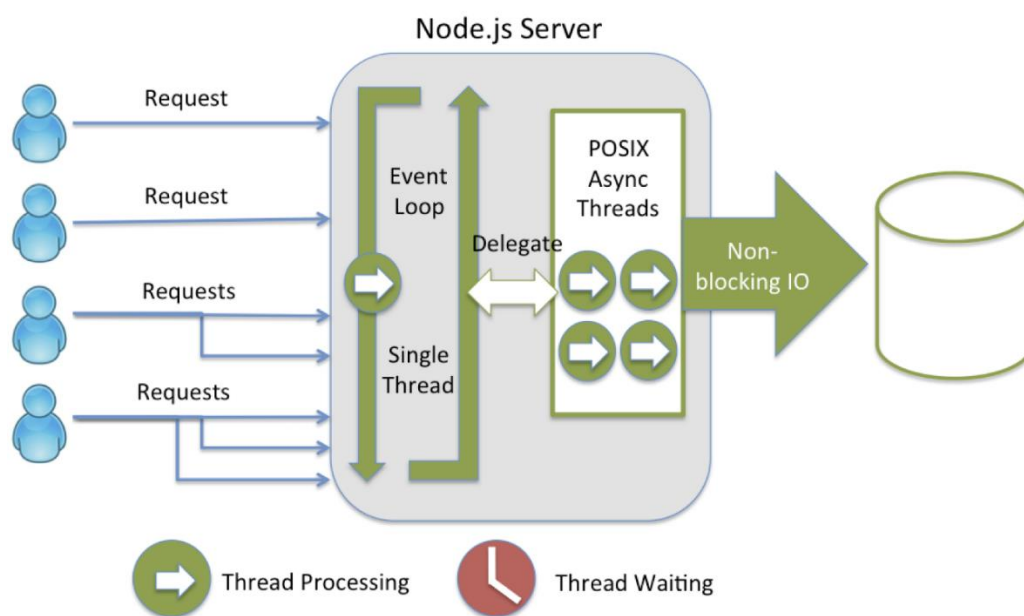


Fig 3

Node.js a contribuit la îmbunătățirea considerabilă a experienței utilizatorilor și a aplicațiilor web care necesită un număr important de intrări și ieșiri. Site-urile web și aplicațiile precum serviciile de streaming, jocurile și serviciile de chat sunt reprezentate în mare măsură de

⁴ <https://codeburst.io/all-about-node-js-you-wanted-to-know-25f3374e0be7>.

capacitățile de intrare / ieșire asincrone.

Asocierea I/O se referă la capacitatea frameworkului de a putea folosi în continuare funcții, chiar dacă există intrări și ieșiri care lipsesc la anumite momente din execuția funcției. Acest tip de administrare I/O este ideal pentru programe și site-uri web care necesită o cantitate foarte mare de interacțiune cu utilizatorul (ca o aplicație chat în care interacțiunea cu utilizatorul include în cea mai mare parte toată logica).

Cel mai mare impact a fost, așa cum a fost prezentat pe scurt, în companiile și comunitățile care execută programe ce necesită un număr mare de operații I/O: Netflix, Paypal, LinkedIn și GoDaddy sunt doar câteva dintre numeroasele corporații care au adoptat Node.js. Concentrat pe reducerea latenței și manipularea protocolului HTTP, acesta a creat un standard pentru creșterea performanțelor site-ului web și cuantificabilitate. Cu prezență virtuală aproape pe toate platformele JavaScript-ului, Node.js a adus îmbunătățiri substanțiale în ceea ce privește îmbunătățirea expertizei utilizatorilor, dar și potențialul general al site-urilor și aplicațiilor pe internet.

Node.js administrează foarte bine toată logica necesară aplicației pentru a funcționa fără probleme. Câteva dintre aceste avantaje sunt:

- rapiditate foarte mare
- trafic de date continuu
- bazat pe Javascript, limbaj care este folosit și pe front-end
- se ocupă de statusul de conectare al utilizatorilor în timp real
- toate cererile către baza de date pot fi manipulate

Figura 4 de mai jos arată impactul Node.js asupra programatorilor⁵:

⁵ <https://hackernoon.com/node-js-emerging-as-the-universal-development-framework-for-a-diversity-of-applications-c2e788290f5f>.

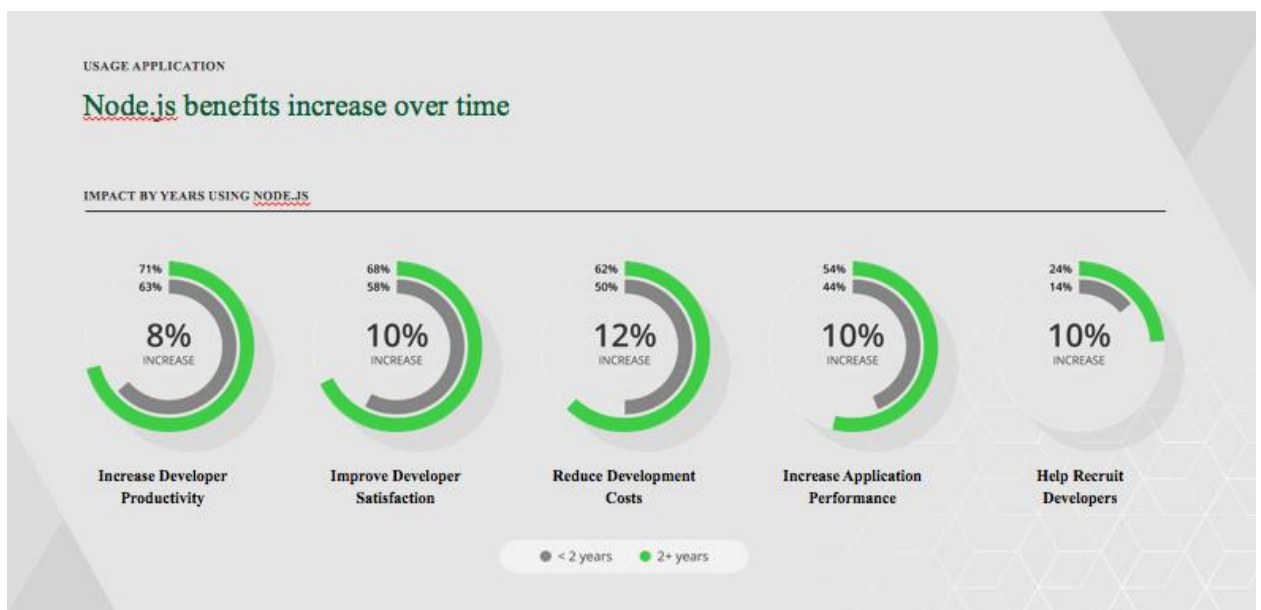


Fig 4

Node.js este prezent și pe frontend; acesta rulează permanent tot proiectul și folosește anumite pachete pentru a face legătura cu backend. Cu toate acestea, modulele utilizate nu sunt duplicate pentru fiecare ramură în parte, ci acestea sunt la comun și au aceleași versiuni pentru a nu se crea probleme în cadrul dependențelor dintre ele.

Pachetele sunt găsite în ramura principală a aplicației sub denumirea de „node_modules”, iar acolo sunt dispuse în ordine alfabetică cu toate fișierele necesare, atât de funcționalitate cât și de design sau expunere pentru alte module.

Având astfel prezent Node.js în cadrul întregii aplicații, acest server rulează atât pentru backend, cât și pentru frontend simultan, dar, cu toate acestea, în cazul în care este o problemă (de exemplu la un serviciu REST), serverul Node.js nu oprește funcționarea întregii aplicații, ci doar a problemei în cauză pentru a nu strica experiența utilizatorilor din mediul online.

Anexa 2 – Express.js

Express.js este un framework web care structurează aplicațiile web pentru a se putea ocupa de diferite cereri HTTP la o adresă specifică. Acesta este construit pe Node.js pentru a avantaja implementarea aplicațiilor web ce folosesc servicii REST.

Puntea de legătură dintre frontend și backend este Express.js, prin expunerea unor Rest API administrate de acesta, astfel încât aplicația web poate fi împărțită în diverse sarcini care au un scop comun sau nu, dar în același timp sunt separate una de cealaltă. Fiind open-source, developerii pot modifica și crea conținut după nevoile platformei.

Toate serviciile sunt regăsite modularizat în fișierele aplicației și au denumiri specifice în funcție de obiectivele și de funcționalitățile acestora. Mai jos, în figura 5, sunt exemplificate metodele de expunere a acestora:

```
app.use( fn: '/api/auth', authRoutes );  
app.use( fn: '/api/posts', postsRoutes );  
app.use( fn: '/api/users', usersRoutes );  
app.use( fn: '/api/notifications', notificationsRoutes );  
app.use( fn: '/api/reviews', reviewRoutes );  
app.use( fn: '/api/logs', logsRoutes );  
app.use( fn: '/api/statistics', statisticsRoutes );
```

Fig 5

Folosind Express.js am putut crea conținut și funcții pentru fiecare dintre ele și, în același timp, acestea reprezentând o punte de legătură dintre ce se expune online și ce se modifică în baza de date. De asemenea, serviciile au o legătură unele cu altele prin expunerea internă de anumite funcționalități necesare individual, în funcție de cerințele acestora.

Anumite API sunt private, adică expuse doar în cazul în care anumite criterii sunt îndeplinite, iar celelalte sunt publice și pot fi utilizate indiferent de restricții. Scopul acestora este de a prezenta sau de a popula anumite locații din aplicație.

Express.js permite folosirea anumitor „middle-ware” pentru a executa anumite funcționalități la fiecare cerere sau răspuns. AskME folosește diferite middleware pentru a manipula sau nu anumite date.

Anexa 3 – Web Sockets

Web-ul a fost construit în jurul ideii că sarcina unui client este de a solicita date de la un server, iar misiunea unui server este de a îndeplini aceste cerințe. Această paradigmă a rămas necontestată de mai mulți ani, dar odată cu introducerea AJAX în jurul anului 2005, mulți oameni au început să exploreze posibilitățile de a face conexiuni între un client și un server bidirecțional.

Problema cu această soluție este că aceasta suportă overhead-ul HTTP. De fiecare dată când se efectuează o solicitare HTTP, o mulțime de anteturi și date cookie sunt transferate pe server. Acest lucru poate adăuga până la o cantitate suficient de mare de date care trebuie transferată, ceea ce la rândul său crește latența. Dacă se construiește ceva de tipul unui joc bazat pe browser, reducerea latenței este esențială pentru a menține lucrurile fără probleme. Cea mai neavantajoasă parte a acestui lucru este că multe dintre aceste antete și cookie-uri nu sunt de fapt necesare pentru a îndeplini cererea clientului.

Clientul stabilește o conexiune WebSocket printr-un proces cunoscut sub numele de handshake WebSocket. Acest proces începe cu trimiterea de către client a unei cereri HTTP obișnuite către server. Un antet de upgrade este inclus în această solicitare care informează serverul că clientul dorește să stabilească o conexiune WebSocket.

Arhitectura AskME permite acestui tip de comunicare să aibă loc și astfel, pe lângă cererile de tip REST, sunt expuse și cele de tip Socket. În cadrul aplicației sunt prezente diferite metode de transfer care sunt asemănătoare ca și concept cu cele API, diferența constând în metoda de comunicare și tipul de expunere al acestora.

În tabelul 1 de mai jos sunt afișate diferențele dintre WebSockets și API Rest pentru a evidenția scopul folosirii acestora în cadrul aplicației AskME:

WebSocket	REST
folosește HTTP pentru a iniția o conexiune	HTTP este protocolul comun de comunicare
comunicare bazată pe socket	comunicare bazată pe resurse mai mult decât pe comenzi
scenariul unei aplicații în timp real	foarte multe cereri și apeluri API
dependența(conexiunea) se face pe baza unei	dependența bazată pe protocolul HTTP și folosește metode HTTP pentru a obține date

adrese IP și al unui port	
costul de comunicare este mic	costul de comunicare este mult mai mare
performanță crescută în cazul numărului mare de cereri	ideal pentru cereri ocazionale

Tabelul 1

Aplicația AskME, datorită necesității de comunicare în timp real dar și de statistici sau de notificări utilizează, pe lângă serviciile REST și web sockets. Acestea sunt prezente și folosite în funcție de tipul de cerere și de funcționalitate. Câteva exemple din cadrul platformei, care sunt servite de către sockets:

- stream video live
- statistici actualizate în funcție de acțiunile fiecărui utilizator
- notificări în timp real
- controlul permanent al prezenței online/offline pentru fiecare persoană
- mesaje de cerere/anulare conexiune în rândul utilizatorilor

Figura 6 de mai jos prezintă un exemplu din aplicație în care este utilizat un socket pentru a administra deconectarea unui utilizator.

Din punct de vedere arhitectural, au fost folosite tehnologii care avantajează expunerea anumitor funcționalități necesare de către AskME. WebSockets, fiind foarte utilizate, dispun de un număr foarte mare de actualizări, ceea ce oferă platformei garanția că își poate continua dezvoltarea în viitor pe această arhitectură care este în strânsă legătură cu toate componentele sale. Acest lucru se poate observa în figura 7 de mai jos⁶:

```

socket.on('disconnectNow', data => {
  User.findOne({username: data}).then( resolve: user => {
    if (user) {
      user.online = false;
      users = users.filter(function (obj) {
        return obj !== data;
      });
      user.save();
    }
  });
  socket.disconnect();
  io.emit('user disconnected');
});

```

Fig 6 – Socket deconectare

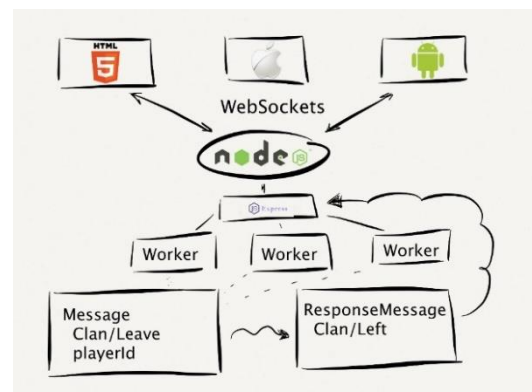


Fig 7 Arhitectură sockets

⁶ <http://www.itechmatics.com/webapps.php>.

Anexa 4 - MongoDB

MongoDB a fost dezvoltat în 2007 de către o organizație din New York, "10gen", care acum este numită MongoDB Inc. MongoDB a fost dezvoltat inițial ca o platformă PAAS (Platform as a service). Acesta este un tip de server de baze de date orientat pe documente, dezvoltat în limbajul de programare C++.

Fiind un sistem non-SQL de gestionare a bazelor de date, este mult mai simplu și complexitățile care vin cu bazele de date relaționale sunt eliminate în MongoDB. Datele de stocare orientate spre tip de colecții/documente sub formă de JSON simplifică sistemele de baze de date.

AskME stochează informațiile necesare sub formă de colecții care sunt administrate de către server. Pentru că aplicația funcționează în mare măsură prin interacțiuni în timp real, este nevoie ca datele să fie preluate și modificate foarte rapid, motiv pentru care MongoDB face acest lucru foarte bine.

Colecțiile stocate în baza de date sunt dispuse într-un mod cât mai intuitiv, care determină facilitarea preluării informațiilor necesare și sunt prezente sub această formă:

- users: conține toți utilizatorii aplicației și informațiile acestora
- statistics: statisticile și acțiunile utilizatorilor
- reviews: recenziile oferite pentru fiecare persoană
- notifications: notificările în timp real
- logs: sunt stocate toate conversațiile video dintre 2 utilizatori

Datorită faptului că NoSQL oferă posibilitatea de a salva colecții sub forme diferite, se poate observa o diferență de informații dintre anumiți utilizatori în funcție de interacțiunile pe care aceștia le au în cadrul aplicației AskME. Serverul știe să interpreteze aceste informații și pune indecși aferenți pentru fiecare acțiune în parte, pentru a mări viteza de procesare de informații atunci când este cazul.

În concluzie, interogările făcute de către WebSockets și REST sunt citite și expuse cu o viteză foarte mare de către Node.js și actualizate sau preluate din MongoDB. Astfel, utilizatorii beneficiază de o acuratețe și o experiență de navigare foarte plăcută, astfel se creează senzația de utilizare a aplicației într-un mod cât mai instantaneu.

Anexa 5 - JWT

JSON Web Token este un standard folosit pentru a crea tokens de acces pentru o aplicație. Serverul generează un token care certifică identitatea utilizatorului și îl trimite clientului. Clientul va trimite tokenul înapoi la server pentru fiecare solicitare ulterioară, astfel încât serverul cunoaște că cererea provine de la o anumită identitate.

Întrucât AskME deține date confidențiale, care nu trebuie expuse către alte entități, am considerat necesară utilizarea unei metode foarte sigure de securizare pentru întreaga arhitectură. Am ales JWT deoarece este cea mai folosită implementare, aceasta fiind utilizată chiar și de către marile companii precum Google, Youtube, ș.a.m.d.

Un JWT este semnat criptografic (dar nu este criptat – prin urmare, utilizarea HTTPS este obligatorie atunci când se stochează datele utilizatorului în JWT), deci există o garanție că putem avea încredere atunci când primim cheia de acces, deoarece niciun agent intermediar nu o poate intercepta și modifica. Astfel, tokenul este generat de către Node.js și trimis către frontend, unde este stocat și utilizat pe baza întregii conexiuni a utilizatorului în cadrul aplicației.

Fiind accesibil când este vorba de personalizări, restricțiile pot fi activate doar pe anumite Rest API; astfel se pot expune anumite informații către public, chiar dacă aceștia nu au un cont activ sau dacă nu sunt conectați la aplicație. Acest lucru este imperios necesar pentru utilizatori; aceștia au posibilitatea de a vedea anumite funcționalități ale aplicației cu toate că încă nu sunt conectați la server.

AskME dispune de 2 roluri, iar acestea sunt securizate cu ajutorul JWT: client și furnizor. Fiecare dintre cele 2 roluri are acces la anumite funcționalități în cadrul aplicației pentru a facilita cât mai mult experiența de utilizare. De asemenea, nu se pot accesa API decât din cadrul aplicației și nu din alte locații externe.

Iată câteva dintre serviciile REST securizate de către JWT:

- /users – toate rutele aferente acestui serviciu sunt securizate
- /statistics – sunt furnizate statistici doar pentru utilizatorul în cauză
- /review – doar funcția de adăugare de recenzie este securizată
- /notifications – sunt oferite notificări doar pentru utilizatorul conectat
- /logs – împarte într-un mod securizat convorbirile dintre 2 persoane
- /auth – doar serviciul de autentificare este securizat în funcție de token, cel de înregistrare nu are restricții decât pentru logica de date trimise de către client

Pe lângă securizarea de bază asupra majorității funcțiilor API ale AskME, JWT este prezent și pentru comunicarea WebSockets, unde acoperă 100% din totalul de emiteri și interceptări ale acestora, expunând informații doar în cazul în care utilizatorul este conectat și are acces asupra lor.

Anexa 6 - Angular

Angular este o parte a ecosistemului JavaScript și unul dintre cele mai populare instrumente de dezvoltare software de astăzi. A fost introdus de Google în 2009 și a primit un feedback numeros și pozitiv din partea comunității de dezvoltare. Conform sondajului StackOverflow din 2018, 36,9% dintre inginerii de software aplică acum AngularJS și noua versiune Angular 2+ pentru a crea interfețe utilizator.

În 2010, principalul beneficiu al AngularJS a fost că permitea transformarea de documente bazate pe HTML în conținut dinamic. Înainte de AngularJS, HTML, limbajul de marcarea web, era întotdeauna static, ceea ce înseamnă că utilizatorii nu puteau interacționa activ cu interfețele pe paginile HTML. Cu toate acestea, Google a considerat că pot fi aduse îmbunătățiri semnificative asupra frameworkului și, de aceea, în septembrie 2016 a fost lansat Angular 2.

Developerii au rescris în întregime codul, potrivit cerințelor în creștere ale web-ului modern. Diferența dintre AngularJS 1.x și noul Angular a fost atât de radicală încât nu era posibilă doar o simplă actualizare de framework, ci era necesară rescrierea platformelor în totalitate.

Datorită numeroaselor actualizări ale frameworkului din 2016 până în prezent și a numărului mare de websiteuri care folosesc această tehnologie, am considerat că aplicarea ei în cadrul AskME este benefică și poate beneficia de actualizări pe viitor în cazul continuării dezvoltării acesteia.

Angular, oferind posibilitatea de interacțiune a supersetului Javascript, și anume Typescript, cu limbajul de tip mark-up HTML și necesitatea acestei funcționalități pentru a implementa stream video, acest lucru a putut fi posibil cu ajutorul modulelor și componentelor furnizate de către framework.

Ajungând în punctul final al prezentării de tehnologii utilizate, AskME este astfel o aplicație “*MEAN*”:

- M – MongoDB, bază de date cu relaționare NoSQL
- E – Express, middleware backend
- A – Angular, framework frontend
- N – NodeJS, server backend

Angular este în strânsă legătură cu celelalte 3 mari arhitecturi de sistem, dar și cu JWT prin care se securizează atât serviciile cât și componentele și modulele din frontend. Folosind Javascript în cadrul tuturor tehnologiilor, aplicația poate fi dezvoltată de programatori care cunosc bine acest limbaj chiar dacă au mai puține cunoștințe de server-side.

Avantajele Angular asupra aplicației AskME:

1. Arhitectură bazată pe componente ce oferă o calitate ridicată a codului scris
2. Refolosire de cod: componentele pot fi afișate și utilizate în mai multe locuri în aplicație – astfel se elimină codul duplicat
3. Informații prezente într-un mod organizat și ușor de înțeles pentru developeri care încep pentru prima dată programarea aplicației; astfel AskME beneficiază de o mentenanță ușoară pe viitor
4. Unit-test prietenos: fiind componente independente, testarea se poate face individual pentru fiecare în parte
5. Durabilitate: componentele, fiind separate una de cealaltă, pot fi ușor reimplementate fără a strica funcționalitățile celorlalte elemente.

Folosind Typescript, aplicația poate fi rescrisă cu ușurință în frameworkul “*Ionic*” pentru a putea deveni aplicație nativă pentru Android și iOS. Astfel, singurele diferențe în cadrul programării ar fi doar adăugarea de elemente mobile: cameră, microfon, mesagerie, notificări native, ș.a.m.d

Datorită lipsei de reîncărcări a fiecărei pagini ale aplicației, AskME este o platformă “*single page*”, adică în momentul în care un utilizator navighează pe website, acesta nu reîncarcă tot conținutul paginilor, ci doar ascunde și aduce în față componentele necesare. Fiind o aplicație care folosește WebSockets și video stream, acest lucru este foarte benefic pentru satisfacția vizuală dar și pentru a mări rapiditatea timpilor de răspuns ale frontendului.

Cu toate că Angular oferă posibilitatea de a folosi un singur modul în cadrul platformelor, acest lucru scade viteza de încărcare al acestora deoarece, inițial, modulele sunt încărcate “*eager*”, însemnând că toate modulele sunt procesate inițial cu toate că sunt folosite sau nu în acel moment. Pentru a crește viteza de încărcare a aplicației am decis utilizarea “*lazy loading*”, iar acest lucru a fost posibil prin împărțirea pe module diferite în funcție de necesitate și pe încărcarea acestora doar când sunt utilizate.

Diferența de navigare a fost crescută în proporție de 70%, iar rutele au fost simplificate datorită acestui lucru.

Astfel, modulele folosite în cadrul aplicației sunt decuplate unele de altele, iar mai jos voi prezenta câteva dintre acestea:

1. Auth – un modul foarte important deoarece se ocupă cu aproape tot ce ține de securitatea aplicației. Acesta conține două componente (login & register), două servicii pentru securizarea rutelor și un serviciu pentru legătura cu serverul. De asemenea este prezent și un *”guard”* pentru permiterea/restricția accesului de vizionare al altor module în funcție de accesul utilizatorilor
2. User – conține tot ce este necesar pentru informațiile utilizatorilor, atât cele publice cât și cele personale. Componentele aferente sunt: Logs & MyProfile. Aceste componente beneficiază de rute private care furnizează informații utilizatorilor conectați. Serviciul acestui modul servește ca o legătură dintre frontend și backend și deține un validator de informații trimise de către client.
3. Ask – modul care oferă persoanelor o interfață plăcută și intuitivă, pentru a găsi un freelancer disponibil să le răspundă la întrebări. Modulul este alcătuit din două componente și anume: Categories & FindConnection. Cele două sunt în legătură și funcționează cu ajutorul unui serviciu care oferă informații disponibile de pe server.

Fiecare modul are ruta lui proprie cu subcomponentele aferente iar unele dintre aceste componente au rute proprii în cazul în care oferă o funcționalitate ridicată și complexă. Unele dintre acestea ar fi:

- StreamComponent – oferă posibilitatea utilizatorilor să interacționeze prin stream video. Componenta folosește atât WebSockets cât și Rest API și tratează un număr mare de cazuri: deconectare, așteptarea conexiunii, validarea utilizatorilor, înregistrarea duratei și a prețului convorbirii, etc.
- StatisticsComponent – destinată freelancerilor, aceasta afișează statistici detaliate sub forma unor grafice și tabele pentru a cunoaște vizibilitatea acestora în cadrul aplicației. Graficele sunt încărcate într-un mod dinamic și plăcut.