# Introducere

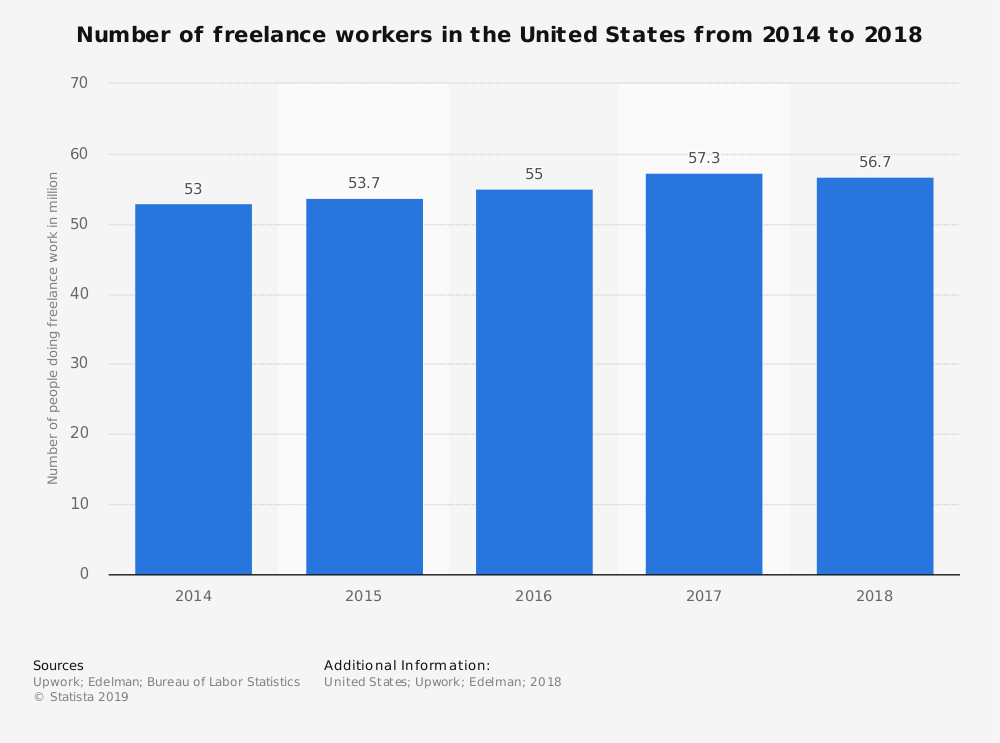
### Istoric

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

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 si 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 crescut substanțial in SUA, aceasta crescând de 3x ori din anul 2014 până în prezent 1, aceasta ajungând până la cifra de 50 milioane de muncitori independenți la acea dată. Acest lucru poate fi observat în graficul de mai jos realizat de *Bureau of Labor Statistics @ 2019.*

.

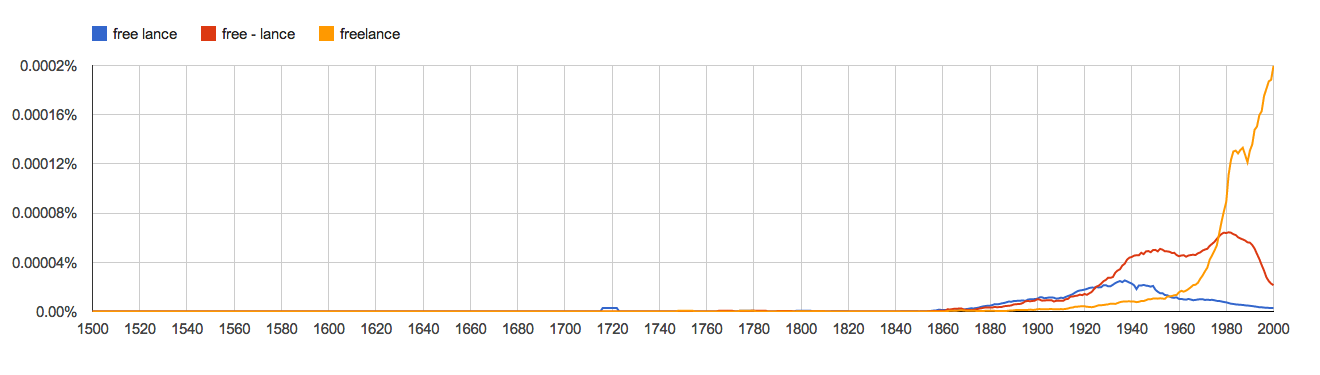
1<https://www.rent-acoder.com/blog/history-of-freelancing/29>

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
* 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 a 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 sau folosite ca mijloc de comunicare.

Cuvântul *‘freelance’* a suferit 3 modificări până a ajunge în forma actuală, formă care este folosită din anul 1970 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.



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

Prima platformă online dedicate 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 online doar că activează sub numele de ‘*Upwork’*.

De la apariția acesteia multe alte website-uri au apărut cel mai mult după anul 2007, odată cu apariția *’Freelance.com’* și *‘Fiverr.com’*. Din acel moment numărul de joburi din mediul online a crescut foarte mult și treptat companiile au început sa colaboreze cu freelanceri.

Astăzi sunt peste 1000 de portaluri de acest gen pe tot internetul, acest lucru facilitând competiția în industrie atât pentru muncitori cât și pentru website-uri, oferind o 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 clienților țintă. Motivul cel mai important care este în interesul unui freelancer este comisionul cel mai mic pe care platforma îl primește pentru munca desfășurată de persoană. De aceea, platforma *‘AskME’* se vrea a fi o platformă având comisioane cât mai mici și fără restricții pentru fiecare individ în materie de preț 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 in permanentă ’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. Atenția se face în egală măsură pentru client cât și pentru furnizor pentru a facilita utilizarea acesteia fără restricții. Această platformă are ca interes comunicarea în timp cât mai rapid dintre cele două entități: client și furnizor.

### Motivaţie

Fiind o persoană care a lucrat ca și freelancer dar și din discuțiile cu diferite persoane de interes am realizat lipsa în domeniul acesta de facilitatea de a oferi anumite servicii prin intermediul ’*video-streamingului’*. De aceea, platforma are ca interes dezvoltarea sa în viitor pentru a putea fi folosită chiar și de persoanele care momentan nu s-au gândit la munca de acest gen, deoarece domeniile regăsite in *‘AskME’* sunt cu mult mai multe față de portalurile actuale de freelance.

Acestea nu sunt doar domenii uzuale, spre exemplu: it, design, video, audio, etc. dar 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.

### Funcționalități propuse

Aplicația dorește a servi toate uneltele necesare unui freelancer și a unui client pentru a face cât mai simplă și utilă comunicarea dintre aceștia printr-o interfață 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 actualizărilor în timp real al notificărilor și ale mesajelor, fără necesitatea reactualizării platformei in cauză.
* pagini statice – lipsa unei interfețe plăcute și dinamice care să-i ofere utilizatorului o experiență plăcută de utilizare, astfel mulți dintre posibilii clienți sau furnizori devin reticenți fată 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. Aceasta problemă este realizată prin prisma unor recenzii date atât de client cât și de furnizor.
* durată de timp mare 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 intra în discuție în acel moment fără a mai aștepta ca freelancerul să se conecteze în aplicație.

În prisma celor enumerate de mai sus, mi-am propus ca aceste probleme să fie rezolvate pentru a oferi o soluție cât mai bună pentru toate persoanele interesate de acest business. Astfel, am prezentat în primul capitol o analiză asupra platformelor existente pe tot internetul. Analiza este făcută pentru a vedea elementele necesare realizării aplicației AskME dar și pentru a aduce îmbunătățiri in domeniul freelance.

În capitolul 2 voi explica tehnologiile folosite pentru realizarea aplicației aducând argumente și explicații pentru a motiva folosirea acestora în detrimentul altor tehnologii.

Capitolul 3 cuprinde prezentarea platformei din perspectiva fiecărui tip de utilizator pentru a vedea rezultatul final și în detaliu asupra fiecărei funcționalități prezente.

# Capitolul 1 – Aplicații web pentru freelance

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

### Upwork.com

În opinia mea personală, după analiza și utilizarea a mai multor aplicații am ajuns la concluzia că aceasta se apropie cel mai mult de nevoile utilizatorilor. Având și o vechime 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 așa de frumos și de simplu, deoarece clienții caută persoane foarte experimentate pentru că majoritatea clienților sunt companii, unele chiar foarte mari care nu sunt de fapt în căutarea unui freelancer, ci mai mult 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 la fel ca 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 20h pe săptămână, dar indicat ar fi 40h.

Funcționalități regăsite:

* Design simplist
* Dificil de aplicat
* Calitatea utilizatorilor datorită experților prezenți în platformă
* Aplicație dedicată mai mult angajărilor remote decât a freelancerilor

### Fiverr.com

Fiverr oferă un sistem puțin diferit față de cele 2 platforme menționate anterior, iar acest sistem este utilizat și în cadrul aplicației AskME. Acest sistem schimbă tipul angajării unui freelancer din perspectiva în care nu clientul pune 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 diversitatea freelancerilor este ridicată datorită nivelului diferit disponibil de experiență a fiecaruia dintre ei.

Funcționalități găsite:

* Mesaje in 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 învechit 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 sistem de freelancing prin intermediul streaming-ului video decât prin metoda clasică, mai specific cea prin mesaje cumparător-furnizor. Majoritatea platformelor nu avantajează deloc freelancerii începători care pot fi capabili în rezolvarea a destul de multe cereri care nu necesită un nivel de cunoștințe foarte ridicat ci poate chiar la nivel de începător.

AskME dorește să diversifice concurența între freelanceri prin a lăsa fiecare utilizator să poată să-și construiască propriul profil în așa fel în cat posibilii clienți să îl aleagă pe el dar și prin setarea unui preț bazat în mod real pe capacitățile sale dar și la nivel concurențial față de ceilalți utilizatori.

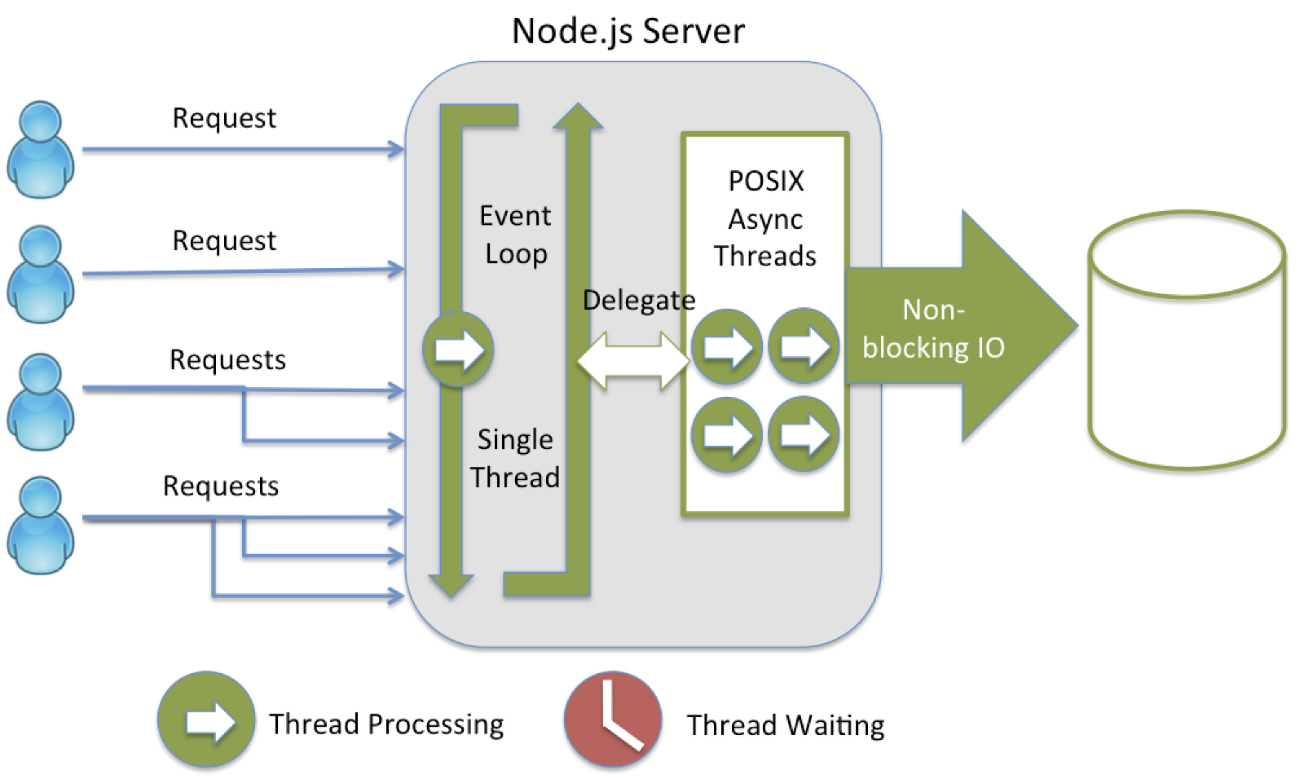
# Capitolul 2 – Tehnologii folosite

### 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 de mai jos 3:



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

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 cum ar fi serviciile de streaming, jocurile și serviciile de chat sunt reprezentate în mare măsură de capacitățile de intrare / ieșire asincrone.

Asocierea I / O se referă la capacitatea frameworkului de 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 este include în cea mai mare parte toată logica).

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

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 permanentă și așteaptă noi cerințe și apeluri din diferite locații, apeluri care pot fi atât din partea utilizatorilor aplicației dar și din alte locații 3rd party cum ar fi: Postman, aplicații offline, baze de date cloud etc.

Node.js administrează foarte bine toată logica necesară aplicației pentru a funcționa fară probleme, câteva din aceste avantaje sunt:

* rapiditate foarte mare
* trafic de date continuu
* bazat pe Javascript, limbaj ce 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
* securitate prin proxy

Figura de mai jos arată impactul Node.js asupra programatorilor 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 in cauză pentru a nu strica experiența utilizatorilor din mediul online.

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

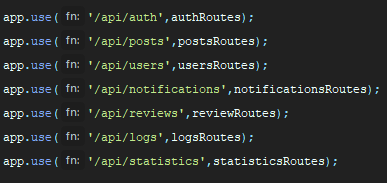
### Express.js

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.

AskME, bazându-se pe servicii REST am considerat că utilizarea acestui framework este necesară pentru administrarea conexiunii dintre frontend si 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.

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



Folosind Express.js am putut crea conținut și funcții pentru fiecare dintre ele și în același timp acestea fiind o punte de legătura 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.

Unul dintre acestea ar fi: pentru fiecare apel care necesită această funcționalitate, middleware-ul verifică dacă utilizatorul este conectat sau nu, iar dacă nu este acesta întoarce unul sau mai multe mesaje de eroare pentru a nu furniza date anumitor persoane care sunt neautorizate.

Cu ajutorul Express.js serviciile REST pot accesa/modifica/șterge înregistrări din baza de date atât în cazul unei cereri din partea clienților dar și atunci când serverul consideră că este necesar acest lucru.

Această expunere se face din locul principal al serverului iar conexiunea se face din momentul în care acesta este pus în funcțiune. În cazul în care conexiunea nu este făcută cu succes, serverul rămâne în funcțiune pentru a furniza în continuare date disponibile fără o legătura la baza de date.

### 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 acestă 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 genul unui joc bazat pe browser, reducerea latenței este esențială pentru a menține lucrurile fără probleme. Cea mai proastă parte a acestui lucru este că multe dintre aceste antete și cookie-uri nu sunt de fapt necesare pentru a îndeplini cererea clientului.

Ceea ce este adevărat nevoie este o modalitate de a crea o conexiune persistentă, având latență scăzută și care poate suporta tranzacțiile inițiate fie de client, fie de server. Acesta este exact ceea ce oferă Web Sockets și tocmai de aceea ele sunt prezente în cadrul aplicației AskME. Ele oferă o conexiune permanentă între un client și server pe care ambele părți le pot utiliza pentru a începe să trimită date în orice moment.

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 sunt expuse pe lângă cererile de tip REST și cele de tip Socket. În cadrul aplicației sunt prezente diferite metode de transfer care sunt asemănâtoare ca și concept cu cele API, diferența ar fi 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 in cadrul aplicației AskME:

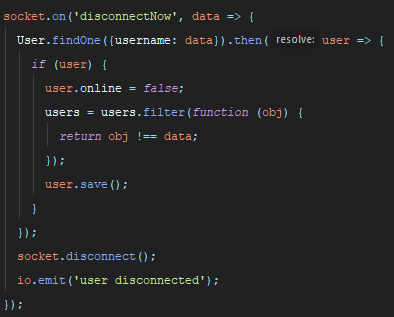
|  |  |  |
| --- | --- | --- |
| WebSocket | REST | |
| folosește HTTP pentru a iniția o conexiune | HTTP este protocolul comun de comunicare | |
| comunicare bazată pe socket | comunicare bazată pe resurse mai mult decât pe comenzi | |
| scenariul unei aplicații în timp real | foarte multe cereri și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

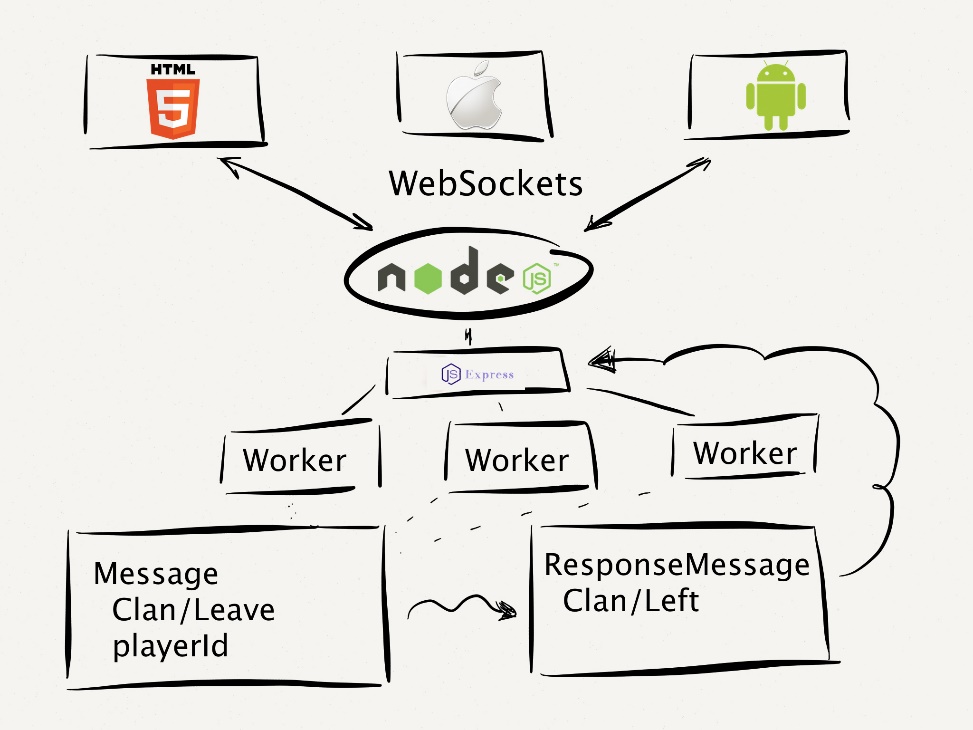
Aplicația AskME, datorită necesității de comunicare în timp real dar și de statistici sau de notificări utilizează, pe lângă serviciile REST și web sockets. Acestea sunt prezente și folosite în funcție de tipul de cerere și de funcționalitate. Câteva exemple din cadrul platformei care sunt servite de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

Figura 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 de mai jos5:



În concluzie, datorită temei alese și de funcționalitățile în timp real ale aplicației, fără utilizarea de WebSockets acestea nu ar fi fost posibile. Cu toate acestea, exista și posibilitatea ca toată platforma să fie structurată pe sockets dar din cauza anumitor elemente care pot fi implementate cu ajutorul serviciilor REST, am decis ca și acestea să fie prezente în arhitectura AskME.

### 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.

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

Datorită metodologiei de administrare si de operarea 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 trebuiesc stocate în baza de date sunt diferite unele față de altele iar din acest motiv este necesară utilizarea MongoDB.

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 și mai ușor de preluat informațiile 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

NoSQL oferind posibilitatea de a salva colecții sub forme diferite se poate observa o diferență de informații dintre anumiți utilizatori în funcție de interacțiunile care aceștia le au în cadrul aplicației AskME. Serverul știe să interpreteze aceste informații și pune indexi 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 MariaDB. 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.

### 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.

AskME având datele confidențiale care nu trebuiesc expuse către alte identități 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.

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 ale utilizatorului în cadrul aplicației.

Fiind accesibil când este vorba de customizări, 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 securitzată
* /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 lânga securizarea de bază asupra majorității funcțiilor API ale AskME, JWT este prezent și pentru comunicarea WebSockets unde acoperă 100% din totalul de emiteri si interceptări ale acestora, expunând informații doar în cazul în care utilizatorul este conectat și are acces asupra lor.

### 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 mare 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 permite transformarea de documentele bazate pe HTML în conținut dinamic. Înainte de AngularJS, HTML, limbajul de marcare 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 celor multe 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 reîncărcării 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 plăcerea 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, doar că 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 făcut 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 de asemenea rutele au fost simplificate datorită acestui lucru.

Astfel, modulele folosite în cadrul aplicației sunt decuplate unele de altele iar mai jos voi prezenta câteva din 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 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 intrebări. 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

### Concluzie

Aplicația AskME este o aplicație de tip MEAN (MongoDB, Express, Angular, Node) folosind astfel ultimele tehnologii dar în același timp și printre cele mai benefice pentru scopul și tema aplicației. Dinamica informațiilor si rapiditatea acestora sunt administrate foarte bine de fiecare componentă a arhitecturii și sunt afișate ușor si securizat.