ACADEMIA DE STUDII ECONOMICE BUCUREŞTI

FACULTATEA DE CIBERNETICĂ STATISTICĂ ȘI INFORMATICĂ ECONOMICĂ

LUCRARE DE LICENȚĂ

Aplicație pentru administrarea unui sistem de votare în cadrul unei organizații

Coordonator științific:

Prof. Univ. Dr. Diaconiță Vlad

Student:

Dragomirescu Oana-Alexandra

2021

CUPRINS

INTRODUCERE

În general, se consideră faptul că voluntariatul reprezintă o activitate realizată în timpul liber din proprie inițiativă cu scopul de a-i ajuta pe alții fără a primi ceva în schimb.

Din punctul meu de vedere, acesta nu este doar o simplă activitate, ci, mai degrabă, o nevoie complexă pe care toți oamenii o au. Chiar dacă nu există o recompensă materială, sunt de părere că sunt oferite foarte multe lucruri care ajută la dezvoltatea personală. Spre exemplu, prin a face voluntariat, se pot învăța sau îmbunătăți o mulțime de *soft skill*-uri, cum ar fi abilitatea de a vorbi în public, de a-ți exprima părerea, de a lucra într-o echipă, de a-ți face prieteni și legături. O persoană introvertită are posibilitatea de a ieși din zona de confort și de a deveni mai sociabilă, ajungând chiar să începă ea discuție cu persoane necunoscute.

Din fericire, am avut oportunitatea de a-mi împlini această nevoie în cadrul celor trei ani de facultate prin participarea activă în cadrul Sindicatului Studenților din Facultatea de Cibernetică, Statistică și Informatică Economică (SiSC) și în cadrul Microsoft Learn Student Ambassadors Romania ASE (MLSA).

Organizația non-profit Sindicatul Studenților din Facultatea de Cibernetică, Statistică și Informatică Economică a fost fondată în anul 1996 și este axată activității de reprezentare a studenților, fiind singura organizație studențească care s-a dedicat în principal acestui scop din cadru Academiei de Studii Economice. SiSC nu luptă doar pentru respectarea drepturilor studenților, dar și pentru asigurarea unui cadru de învățare calitativ superior tuturor tinerilor care au dorința de a se dezvolta atât pe plan profesional, cât și pe plan personal.

Microsoft Learn Student Ambassadors Romania ASE(MLSA) este un program anual ce reunește studenții pasionați de tehnologie din cadrul Academiei de Studii Economice. Acesta este disponibil la nivel global, însă are trăsături locale accentuate. MLSA este centrat pe accesul la tehnologie și inovație. De asemenea, programul este axat pe schimbul de experientă între mediul academic și industrie.

Prin participarea la diferite ședințe de alegeri organizate de cele două organizații, am simțit pentru prima dată **nevoia unei aplicații de vot**. Desigur, odată cu apariția situației pandemice, a început să se simtă din ce în ce mai mult **necesitatea de a migra spre online** și necesitatea acestei aplicații.

Astfel, am decis să realizez pentru licență ceva ce știu că va fi folosit, ceva ce pot lăsa în urmă – o **aplicație de vot**, numită **VV**. Deși scopul ei principal va fi utilizarea în cadrul unei asociații, aceasta poate fi extinsă și pe plan general.

1. IMPORTANȚA SISTEMELOR DE VOTARE ÎN SOCIETATEA MODERNĂ

- ## Ce este votul?

Din punctul meu de vedere, votul este o alegere, un drept, prin intermediul căruia cetățenii selectează persoane, partide sau coaliții ce vor urma să îi reprezinte în procesul de conducere al statului respectiv. Acest lucru este folosit și în cadrul unor organizații pentru a lua mai multe decizii ce implică mai mulți participanți.

Acțiunea de a vota este un proces fundamental într-un sistem democratic. Pentru cetățenii unei țări, este o șansă de a avea un cuvânt de spus legat de oamenii care îi reprezintă sau legat de o problemă care îi afectează. De aceea, informarea despre vot și participarea la alegeri este o responsabilitate importantă a celor care votează.

Așadar, prin intermediul votului, un grup de oameni își exprimă opinia în legătură cu o propunere sau o candidatură [1].

- ## Ce este un sistem de vot?

Ca o definiție generală, sistemele de vot sunt niște mecanisme cu ajutorul cărora se aleg reprezentanții unei grupări. Acești reprezentanți sunt însărcinați cu puterea să își reprezinte și să facă alegeri pentru cei care îi aleg. Sistemele electorale sau sistemele de vot din întreaga lume au o legătură puternică cu sistemele de partide [2].

Din punctul de vedere al analizei democratice, există două tipuri de sisteme de vot: **sistemul electoral majoritar** și **sistemul electoral cu reprezentare proporțională**. Primul fel este întâmpinat, de obicei, în țări precum Marea Britanie sau Statele Unite ale Americii, țări în care este întâlnită democrația majoristă, pe când al doilea tip de sistem este caracteristic țărilor cu democrație consensualistă (de exemplu: Italia, Belgia, Germania) [3].

Deosebirile dintre sistemul electoral majoritar și sistemul electoral cu reprezentare proporțională sunt în strânsă legătură cu două posibile interpretări ale procesului democratic. Pe când una dintre ele se fundamentează pe logica deciziei eficiente, punând accentul pe exprimarea cerinței majorității, cealaltă are complet altă abordare, utilizând principiul proporționalității. Prioritatea nu este ca rezultatul alegerilor să exprime prezența unei majorități, ci evitarea dezavantajării minorităților semnificative.

- **Care sunt tipurile unui sistem de vot?**

În cadru asociațiilor, sistemele de vot sunt o necesitate. Prin intermediul acestora se poate vota într-un mod democratic respectând normele legale de transparență decizională, iar rezultatele pot fi afișate în timp real.

Pe lângă clasificarea anterioară, sistemele de vot se pot împărți în: sisteme de vot portabile, sisteme de discuții cu vot integrat, sisteme de votare la distanță [4].

Sistemele de vot portabile asigură mobilitate, întrucât pot fi tarnsportate cu ușurință de la o locație la alta și prezintă varietate opțiunilor de vot care răspund necesităților organizațiilor.

Sistemele de discuții cu vot integrat au funcționalitatea de a răspunde diferitelor cerințe care apar în timpul unei întâlniri: votarea propriu-zisă într-un interval temporar dorit, exportarea rezultatelor, stocarea și întocmirea de statistici pe baza acestora.

Cel de-al treilea fel, **sistemele de votare la distanță**, devin din ce în ce mai utile în cadrul videoconferințelor și situațiilor în care participanții se află în locații diferite. Este vorba despre folosirea votului electronic pentru a facilita exprimarea democratică a alegerii, dar și pentru a menține confidențialitatea informațiilor.

1.1. Generalități despre votare și impactul economic

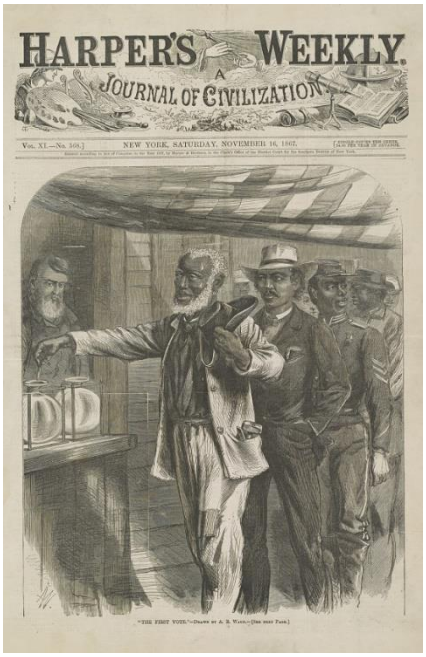
HARPER'S WEEKLY
A JOURNAL OF CIVILIZATION
NEW YORK, SATURDAY, NOVEMBER 16, 1867.

Figure 1 - Harper's Weekly

- **Când a apărut primul vot?**

În data de 16 Noiembrie 1867, coperta revistei *Harper's Weekly* arăta o coadă de bărbați afro-americani îmbrăcați în funcție de profesiile lor, așteptându-și rândul să voteze. Primul bărbat este îmbrăcat în muncitor, cel de-al doilea ca un om de afaceri, următorul poartă o uniforma de armată, iar ultimul dintre votanți un costum de fermier. Al 15-lea amendament al Constituției Statelor Unite declara faptul că indiferent de rasă, culoare sau condiția anterioară de servitute, dreptul cetățenilor la vot nu va fi refuzat sau restrâns atât de Statele Unite, cât și de niciun al stat [5].

Sursă imagine: [6]

Statele din sud au limitat dreptul la vot prin aplicarea taxelor, a testelor de alfabetizare și prin alte mijloace. Astfel, promisiunea amendamentului 15 nu a fost realizată pe deplin decât un secol mai târziu, chiar dacă legea fusese omologată pe 3 februarie 1870.

- ## Ce este, de fapt, dreptul la vot?

Cu ocazia unui scrutin, cetățenii își expun doleațele folosindu-se de dreptul la vot. Persoanele au posibilitatea de a-și alege reprezentații care îi vor guverna sau de a da un răspuns unei solicitări date de guvern printr-un referendum. În democrațiile moderne, dreptul la vot este văzut ca un drept civic fundamental. Deși acesta este privit ca o alegere a unor reprezentanți, sufragiul acordă aceeași importanță și referendumurilor și inițiativelor.

Pentru a-i fi acordat dreptul de a vota, un cetățean trebuie să aibă minim 18 ani împliniți și poate consulta celelalte criterii de eligibilitate stabilite de guvern.

De menționat este faptul că femeile nu au avut ințial dreptul la vot, ci a fost nevoie să îl câștige. În Romania, acestea au putut vota pentru prima data în anul 1929, doar la alegerile locale și având diferite restricții. Zece ani mai târziu, femeilor li s-a ofertit împuternicirea de a vota în condiții egale cu ale bărbaților, însă, în practică, restricțiile impuse pentru ambele sexe au afectat mai mult genul feminine decât cel masculin.

- ## Cat costă un vot?

Conform *mediafax.ro*, pe timp de pandemie costul alegerilor locale a crescut, ajungând la suma de 200 milioane de euro, ceea ce este peste suma cheltuită pentru alegerile europarlamentare.

Totodată, din lecturarea articolului scris de Bianca Chirilă în noiembrie 2016 se poate observa costul alegerilor pentru Senat si Camera Deputatilor. Este vorba de suma de 227,7 milioane de lei care a fost investită pentru a le organiza [7].

Pentru a reduce costurile organizării unui vot, o soluție este utilizarea unei aplicații. Astfel, numărul secțiilor de votare se va diminua, deci și necesitatea de a plăti membrii. Nu va mai fi nevoie de imprimarea buletinelor de vot, utilizarea ștampilelor, montarea cabinelor și amenajarea secțiilor.

Din punct de vedere sanitar, votul online ar conduce la scăderea riscurilor infectării cu SARS-CoV-2. Introducerea unui sistem de vot electronic ar reduce numarul de persoane prezente în sectiile de votare, deoarece mulți vor prefera sa voteze online, fără a se deplasa la secțiile unde

exista risc de răspândire a virusului. Desigur, acest beneficiu al votării online poate fi generalizat pentru orice tip de virus.

De asemenea, diaspora nu va mai fi nevoită să călătorească o distanță considerabilă pentru a vota, ci va putea face acest lucru de acasă, economisind bani și timp. Acest lucru este facil și pentru persoanele cu deficiențe locomotorii.

Generația tânără este deja obișnuită cu sistemele electronice de plată, serviciile de curierat și faptul că nu este obligatoriu să se deplaseze pentru achiziționarea de bunuri, așadar, vor considera oportun acest nou sistem de vot.

Problema care se pune ar fi în legătură cu populația îmbătrânită pentru care vor fi necesare secții de vot și asistență pentru a folosi aplicația, însă aceste lucruri nu vor avea un impact sesizabil asupra costurilor totale pentru organizarea sesiunii de vot.

1.2. Modalități de votare

De-a lungul timpului, procesul de votare s-a îmbunătățit și adaptat în funcție de nevoie și astfel au apărut multiple modalități de a vota.

Printre cele mai cunoscute se numără următoarele [8]:

- **Vot pe hârtie**

Votul pe hârtie este cea mai obișnuită metodă de vot și folosește buletinele de vot. Prin acestea, alegătorii își marchează preferințele. Un sistem alternativ pe hârtie, cunoscut sub numele de scrisori de vot, este utilizat în Israel, unde secțiile de votare conțin o tavă cu buletinele de vot pentru fiecare partid care contestă alegerile; buletinele de vot sunt marcate cu litera (literele) atribuită acelei părți. Alegătorilor li se dă un plic în care pun votul partidului pentru care doresc să voteze, înainte de a plasa plicul în urna de vot. Același sistem este implementat și în Letonia.

- **Vot online**

Există deja țări în care oamenii au posibilitatea de a vota online, însă acestea sunt într-un număr redus. Prima dintre acestea a fost Estonia care a folosit acest tip de vot în anul 2005.

- **Vot prin poștă**

Multe țări permit votul prin poștă. Acesta funcționează în felul următor: alegătorilor li se trimite un buletin de vot acasă, votează și apoi îl trimit înapoi prin poștă.

- **Vot prin mașini de vot**

Pentru acest tip de votare, se folosesc mașini de vot care pot fi manuale sau electronice. Este un proces implementat deja în diferite țări, precum Brazilia, și se desfășoară astfel: alegătorii introduc numărul candidatului preferat și apoi confirmă votul în momentul în care fotografia candidatului apare pe ecran.

- **Vot deschis**

Această tehnică de a vota are loc în public și se face de obicei prin utilizarea membrelor superioare. De exemplu, pentru a vota un candidat, alegătorul trebuie să ridice mâna dreaptă sus. Un sistem de votare public încă folosit este Landsgemeinde, în Elveția.

- **Vot în persoană**

Dacă sunt prezente toate persoanele eligibile pentru vot, atunci acesta se poate desfășura personal.

- **Alte modalități de a vota**

Din cauza numărului mare de analfabeți, în Gambia se utilizează o modalitate de votare folosind marmură, adică în secțiile de votare există fotografii ale candidaților legate de tobe metalice pictate în diferite culori. Persoanei care urmează sa voteze îi este înmânată o bucată de marmură pe care acesta trebuie să o așeze în dreptul candidatului ales, într-un tambur. În acel moment, un clopot sună pentru a înscrie votul. Pentru a fi considerat invalid votul, marmura nu este introdusă în tambur, ci este lăsată deasupra acestuia.

Un alt sistem similiar care a fost utilizat și în România utilizează bile albe și negre. Cea albă înseamnă sprijinul, iar cea neagră opoziția. De aici a apărut termenul de blackballing.

- **Votul în Romania**

În țara noastră, conform Constituției României, votul poate fi **universal**, adică toți cetățenii au posibilitatea de a vota, indiferent de etnie, sex, religie, apartenență socială sau politică. De asemenea, votul poate fi **egal**, implicând o egalitate în voturi (toate au aceeași valoare).

În cadrul votului **direct**, alegerea se exercită în mod nemijlocit de către persoana care va vota. Votul **secret** implică obligativitatea de a-l menține ascuns, iar opusul acestuia este cel **liber exprimat,** în care decizia de sau pe cine a vota îi revine cetățeanului [9] [10].

În cadrul aplicației VV, votul este universal și liber, iar nimeni nu are accesul să schimbe alegerea cuiva, odată înregistrată în baza de date.

1.3. Desfășurarea unui vot

În cadul procesului de votare actual este vital să fie îndeplinite anumite proceduri și reguli pentru ca totul să se desfășoare eficace. Toate aceste aspecte privind organizarea și desfășurarea alegerilor sunt menționate riguros în lege.

Alegerile au loc doar în zile de duminică, nu au voie să depășească intervalul de o zi, iar rezultatul este comunicat prin intermediul *mass-media*.

Alegătorii sunt repartizați în funcție de domiciliu la cea mai apropiată secție de vot. În momentul în care se prezintă acolo, sunt legitimați pe baza actului de identitate și sunt trecuți pe lista de prezență unde semnează ca au primit buletinul de vot.

Buletinul de vot este un formular imprimat prin intermediul căruia se exercită dreptul la vot. Tipărirea buletinelor de vot se face în mod diferit, fiind marcate în mod distinct pentru fiecare tur, obiectiv și nivel pentru a se evita eventuale sabotaje. Ulterior, acestea sunt transmise comisiei electorale cu minim o oră înainte de momentul deschiderii secției de votare.

Votantul se duce în cabina de vot cu buletinul și pune ștampila pe candidatul dorit. În acea cabină accesul se face individual, nu este permis să fie două sau mai multe persoane.

Pentru a fi valid votul, ștampila trebuie să fie vizibilă, pusă într-un singur loc, pentru un singur candidat.

În cazul în care pe buletinul de vot nu există nicio marcație de ștampilă sau aceasta se află în exteriorul casetelor, este anulat. De asemenea, dacă a fost ștampilat de mai multe ori sau există mai multe semne de ștampilă în aceeași căsuță precum și orice alte semne noi, buletinul de vot nu va fi luat în calcul la numărarea voturilor, după cum este menționat în legea nr. 33/2007, publicată în Monitorul Oficial nr. 627 din data de 31 august 2012 [11].

Următoarea etapă este introducerea acelui buletin de vor în urna corespunzătoare. Pot exista mai multe urne de vot, în funcție de scopul alegerilor. Acestea sunt sigilate corespunzător de către doi membri și desigilate după închiderea secției de votare pentru a număra voturile.

În cadrul aplicației VV, nu este nevoie de acest proces complex. Mare parte din desfășurarea activității de vot este automată, voturile sunt numărate direct în program, iar alegerea candidatului dorit este intuitivă. Rezultatul este calculat de *soft*, evitând astfel eventualele erori umane și afișat odată cu finalizarea timpului destinat sesiunii respective.

Va fi nevoie, totuși, de instruirea persoanelor mai în vârstă care nu știu să folosească aplicația sau de existența unor secții de vot pentru cei care nu au posibilitatea de a utiliza aplicația, ceea ce nu reprezintă o problemă din punctul de vedere al costrurilor, întrucât se va reduce semnificativ numărul membrilor din comisiile de vot.

1.4. Importanța procesului

Votul asigură oportunitatea cetățenilor de a face parte din procesul decizional care le afectează viața. Dacă aceștia nu votează, înseamnă că le oferă puterea celorlalți să ia decizii în locul lor. Votul fiecărui individ transmite un mesaj în legătură cu problemele considerate importante.

În cadrul unui stat de tip democratic, votul oferă posibilitatea exprimării dorințelor politice și controlarea puterii. Ba chiar permite participarea sau schimbarea puterii într-o manieră pașnică, organizată și controlată. Fără drept de vot nu există democrație.

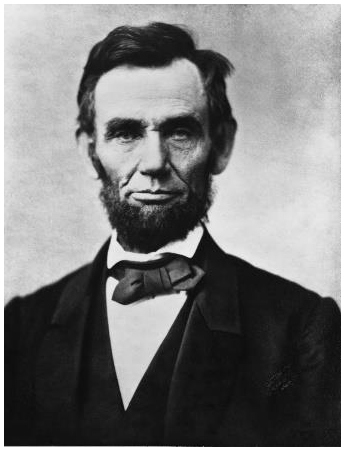

Figure 2 - Portret Abraham Lincoln

Abraham Lincoln a denumit democrația „*guvernarea poporului de către popor și pentru popor*” [12]. Această frază accentuează faptul că guvernul are datoria de a servi cetățenii, nu invers. Așadar, cetățenii au unul dintre cele mai mari drepturi pe care oamenii liberi îl pot avea. Este vorba despre dreptul la vot, un principiu fondator într-o democrație. Acesta plasează puterea în oameni și le oferă posibilitatea de a juca un rol în guvernare.

Sursa imagine: [13]

Cel mai bine pentru o țară este modul în care majoritatea populației decide. În caz contrar există și alte tipuri de conducere, precum dictaturile sau orice alt guvern tiranic. Însă, acestea sunt opuse democrației și tind să impună voința asupra oamenilor, în loc ca oamenii să își impună voința asupra modului de conducere.

Pe parcursul timpului, o mulțime de oameni s-au luptat și sacrificat pentru a-și câștiga dreptul la vot. Acest lucru continuă până în prezent, deoarece există milioane de oameni în întreaga lume care încă nu au dobândit acest drept, lucru care îi oferă o importanță aparte.

2. ANALIZA CERINȚELOR FUNCȚIONALE ALE VV

2.1. Prezentare

Aplicația VV propune o cale inovativă de a desfășura un proces de vot într-un mod sigur și eficient. Aceasta va fi alcătuită din doua părți principale: o parte pentru gestionarea conturilor utilizatorilor și o parte pentru procesul de votare propriu-zis.

- **Gestionarea conturilor utilizatorilor**

Este foarte important ca votanții să facă parte din organizația respectivă și ca accesul la aplicație să fie controlat. Așadar, este nevoie de o verificare suplimentară în momentul în care un cont este creat.

Fiecare utilizator va avea nevoie de un cont cu nume de utilizator și parolă pentru a putea folosi aplicația. Acest cont va fi validat de către administrator pentru a se asigura că doar membrii organizației au acces la aplicație. La rândul său, administratorul va face o cerere către un alt utilizator(*superuser*-ul) pentru a-și crea contul.

Pentru a primi dreptul de administrator, acesta trebuie să încarce o poză cu dovada că deține o poziție de conducere în cadrul organizației în momentul în care își face contul. În schimb, nu este necesar și pentru ceilalți utilizatori să facă pași suplimentari pentru validarea contului, întrucât administratorul(spre exemplu, cenzorul/cenzorii) va știi ce membrii are.

- **Procesul de votare**

O altă componentă foarte importantă este reprezentată de procesul de votare propiu-zis. Interfața aplicației trebuie să fie intuitivă și ușor de folosit, deoarece utilizatorii provin din medii și categorii de vârstă diferite. Aplicația nu este destinată doar organizațiilor orientate pe cunoașterea de tehnologie, ci îi este adresată oricui. Așadar, funcționalitățile sunt poziționate în așa fel încât să fie accesibile.

Pentru a vota, se deschide o sesiune pe o perioadă determinată, iar accesul la aceasta se face prin contul personal, printr-un cod generat aleator pe care administratorul îl va transmite. Utilizatorul introduce codul și astfel îi este afișată întrebarea, însoțită de posibilitățile de răspuns. Fiecare utilizator poate vota o singură dată. Răspunsul poate fi unic sau multiplu, în funcție de problema pusă în discuție și necesitățile organizației fată de sesiunea de vot respectivă.

În momentul în care utilizatorul deschide aplicația, acesta accesează **pagina de autentificare** și își poate introduce creditențialele pentru a intra în cont. De asemenea, are disponibil meniul, din care poate alege să își creeze un cont nou, să intre în cont sau să vizualizeze **pagina principală**. Fără a avea cont, accesul este foarte limitat: pagina principală, pagina pentru înregistrare și pagina pentru autentificare. Mai jos, în capturile de ecran sunt prezentate formularele de autentificare și înregistrare din cadrul aplicației.

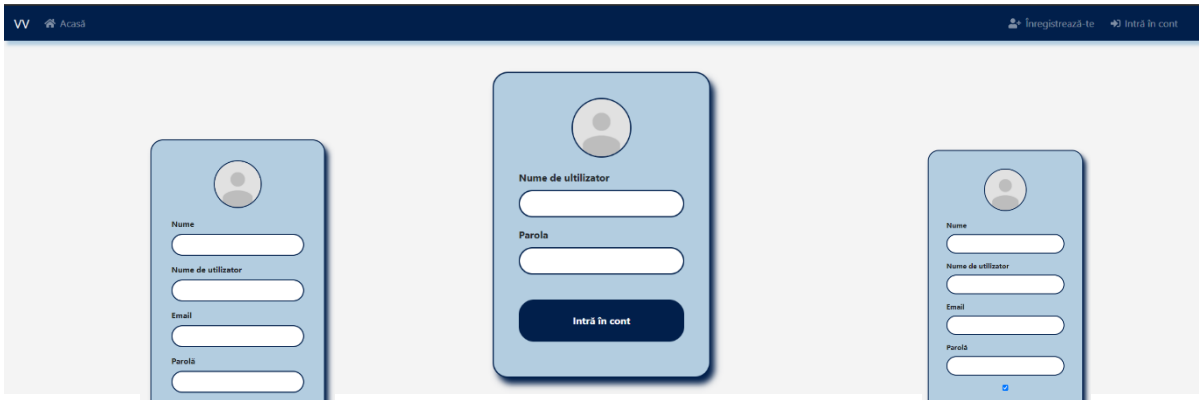

Figure 3 - Pagina de autentificare

Figure 4 - Formular pentru înregistrare votant

Figure 5 - Formular pentru înregistrare candidat

În figura 4 și figura 5 este prezentat **formularul de înregistrare**. Aici, pentru a-și crea un cont de administrator, utilizatorul este obligat să introducă o poză cu dovada că face parte din organizația respectivă și postul de conducere pe care îl ocupă. Acest lucru îi apare după bifarea căsuței din dreptul „Doresc să devin administrator”. Pentru a crea un cont de votant nu este necesară trimiterea vreunei dovezi.

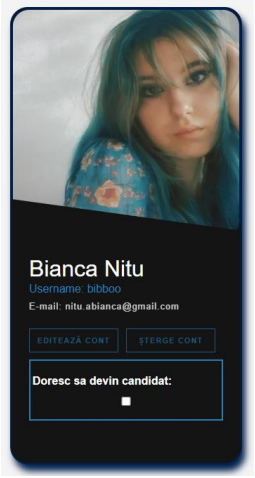

Figure 6 - Card profilul meu

Pagina „**profilul meu**”, se concentrează exclusiv pe utilizator. Acesta poate vizualiza datele actuale ale profilului, le poate modifica și poate alege dacă dorește să candideze. Pentru vizualizarea acestor informații am utilizat niște carduri asemănătoare cu cele pentru candidaturi. Un astfel de exemplu se poate observa în figura 6. În cazul în care este bifat *checkbox*-ul din secțiunea „doresc să devin candidat”, este necesar să trimită o cerere care va fi validată sau nu de administrator. În cazul în care votantul devine candidat, are accesul la o nouă pagină, intitulată „**Candidatură**” prin intermediul căreia își va crea și încărca o candidatură cât mai atractivă. Acest lucru este ilustrat în figura 7.

Figure 7 - Formular pentru adăugarea candidaturii

Există o pagină destinată tuturor candidaturilor. Aceasta poate fi vizualizată de orice tip de utilizator. În figura următoare avem un exemplu în care doi candidați și-au creat candidatura. Aceștia au putut modifica stilul scrisului, culoarea, dimensiunea, alinierea și au putut adăuga poze. Candidatura va fi afișată pe un cartonas, iar poza candidatului este preluată de pe site-ul *https://en.gravatar.com/*, de unde acesta o poate modifica.

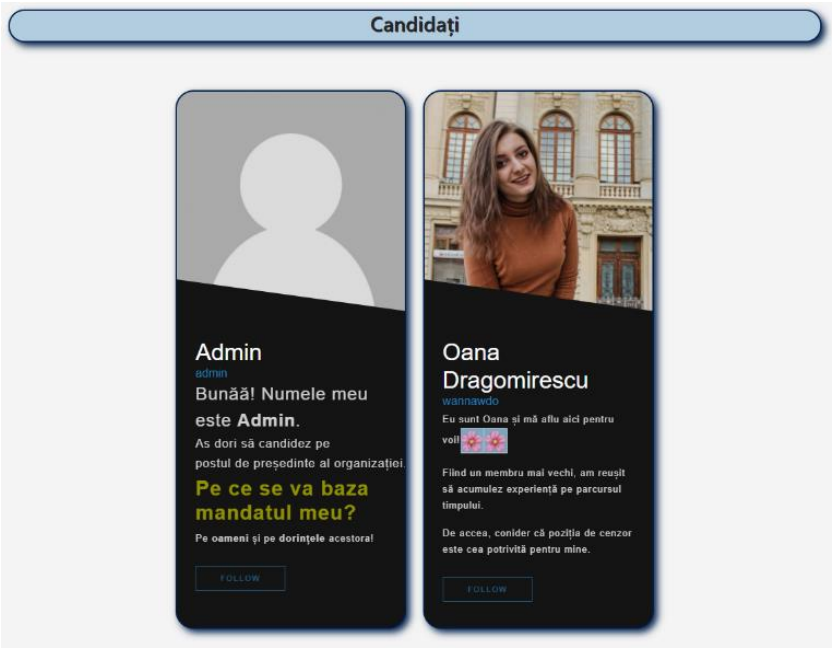

Figure 8 - Exemple de candidaturi

Din punctul de vedere al administratorului, există suplimentar o pagină pentru **gestionarea conturilor** și una pentru **gestionarea sesiunilor de vot**.

Prima dintre acestea este reprezentată în Figura 9. În lista din partea stângă a paginii, administratorul vizualizează conturile și statusul acestora, iar în partea dreaptă are butoanele care îi ofera posibilitatea de a crea sau șterge contul, de a activa sau de a șterge toate conturile. De asemenea, poate căuta un cont după numele persoanei.

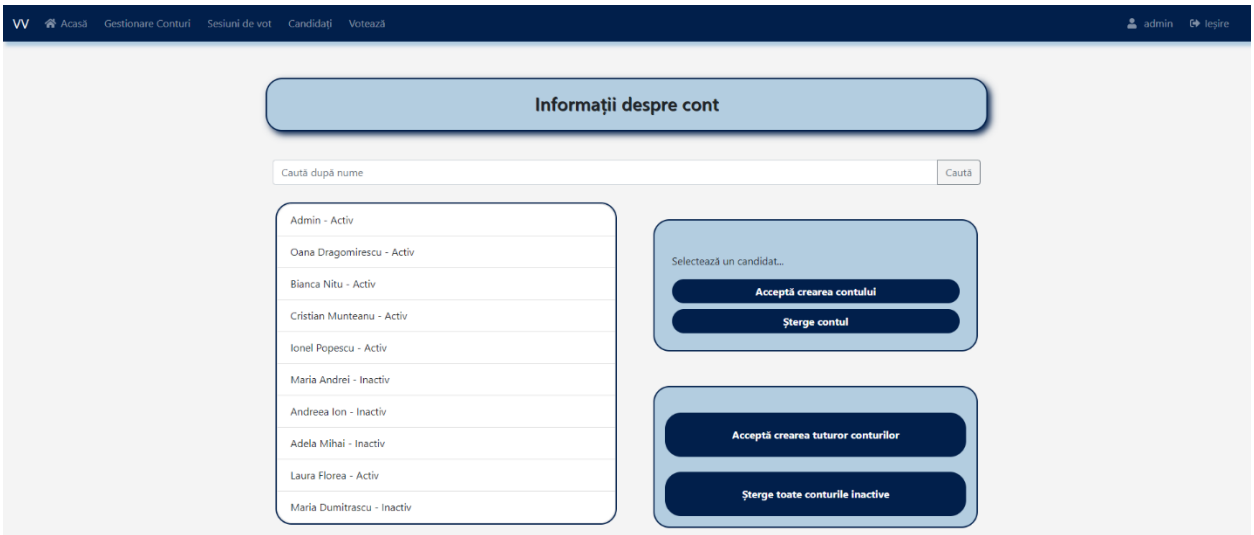

Figure 9 - Pagină pentru gestionarea conturilor

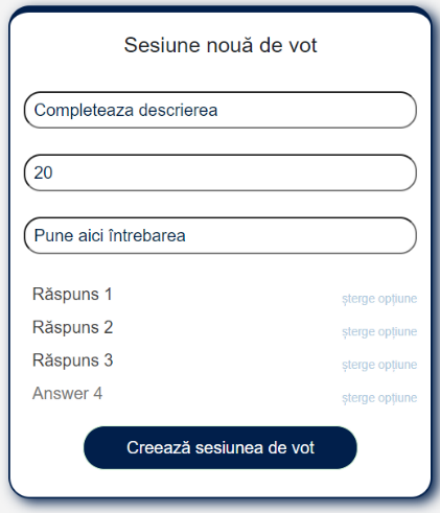

Figure 10 - Formular pentru crearea unei sesiuni de vot

Crearea unei sesiuni de vot îi este destinată tot administratorului, iar formularul specific acestei acțiuni este înfățișat în figura 9. Pentru a fi validă, sesiunea trebuie aibă o descriere, un timp în care să fie disponibilă și întrebarea, însoțită de răspunsuri.

Această sesiune va fi vizibilă într-o listă pe pagina cu sesiunile de vot.

Prin selectarea sesiunii dorite, vor apărea pe ecran informații referitoare la aceasta împreună cu un *input* pentru a adăuga codul de acces. Dacă este valid codul, se va afișa întrebarea, iar utilizatorul va avea posibilitatea de a vota.

Candidații si votanții pot doar vizualiza întrebarea și pot vota. La finalul sesiunii, rezultatul va fi afișat pe ecran, precum în figura 11.

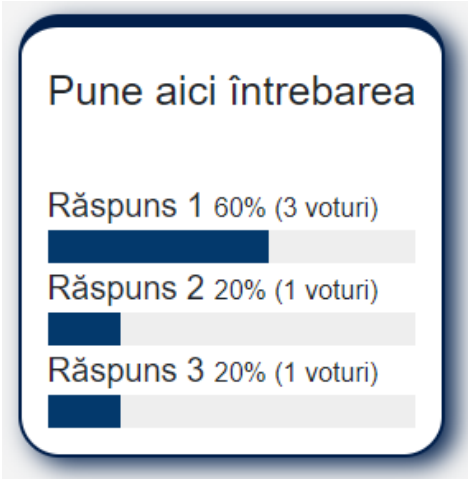

Figure 11 - Exemplu rezultate sesiune de vot

La realizarea interfeței aplicației au fost folosite ca reper următoarele *mock-up*-uri create în programul *Balsamiq*:

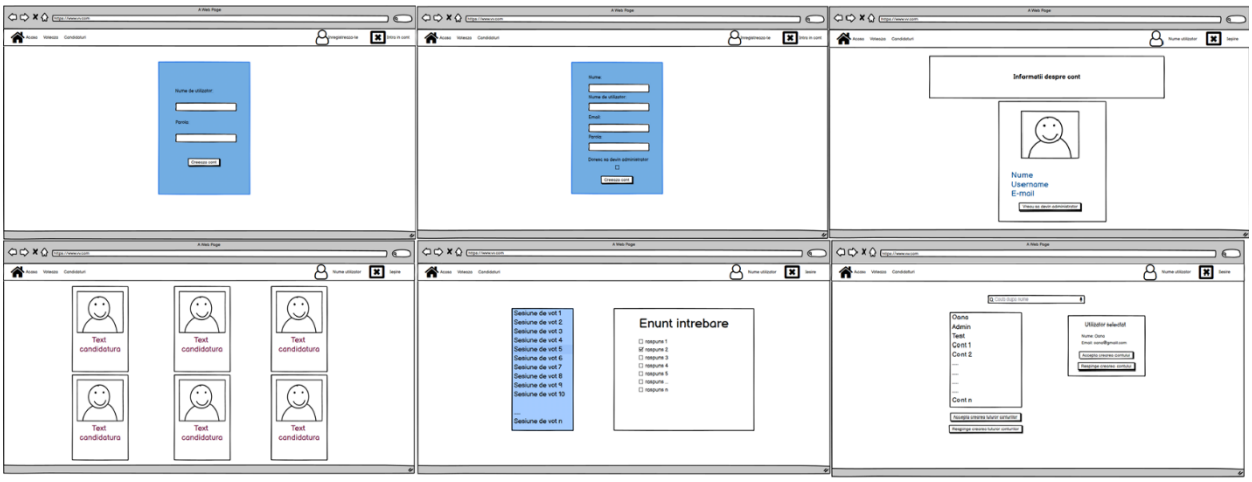

Figure 13 - Interfața utilizatorului în Balsamiq

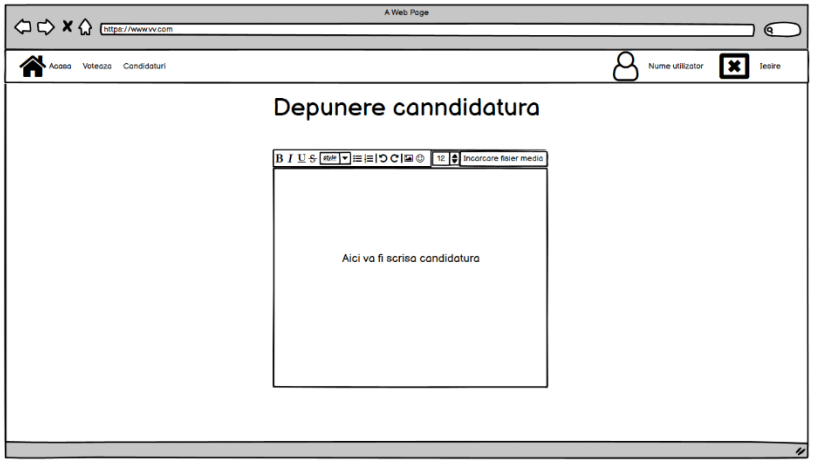

Figure 12 - Interfața utilizatorului pentru Depunere Candidatură realizată în Balsamiq

Acestea înfățișează paginile pentru înregistrare, autentificare, profil, depunerea candidaturii, vizualizarea candidaturilor, vizualizarea sesiunilor de vot și gestionarea conturilor.

2.2. Obiective

Aplicația VV își propune **modernizarea modului actual de votare** prin facilitarea unui stil ordonat și modern în cadrul organizațiilor. Scopul este de a crea sesiuni de vot pe un timp determinat, iar concluzia să fie vizibilă cât mai rapid printr-o aplicație intuitivă, ușor de folosit de către toate categoriile de vârstă. Prin utilizarea aplicației, centralizarea rezultatelor se va face într-un timp scurt și automat, deci cu mai puține șanse de a se greși.

De asemenea, acest mod modern de a vota va **ușura procesul în cadrul organizațiilor**. Oamenii pot vota oricând, de oriunde se află, atât timp cât au conexiune la internet. Acest lucru va favoriza luarea deciziilor și va încuraja mai mulți membrii participanți să își exprime părerea, ceea ce va duce la o reprezentare mai bună a opiniei grupului. Este și o completare pentru videoconferințele din viața de zi cu zi, mai ales că odată cu apariția situației pandemice, s-a dezvoltat destul de mult ideea de a lucra de acasă.

Desigur, nu este nevoie de o pandemie pentru a se observa faptul că o astfel de aplicație este necesară. Există deja o mulțime de firme care au angajați sau clienți situați în diferite țări sau chiar pe diferite continente. Toate acestea folosesc resurse online pentru a comunica, iar prin utilizarea VV, procesul de luare a deciziilor în cadrul ședințelor va fi îmbunătățit.

Totodată, aplicația are potențial și **poate fi dusă la un nivel mai complex**, ajungând chiar să fie utilizată în cadrul alegerilor electorale. Există deja tări în care este posibil. Acest lucru ar aduce o mulțime de avantaje. Întreg procesul de votare ar fi mai puțin costisitor decât cel actual, oamenii în vârstă sau cei cu deficiențe locomotorii ar putea alege de acasă, fără a mai fi necesară deplasarea la secțiile de vot. Modalitatea ar ajuta și diaspora, adică oamenii care locuiesc în afara granițelor țării de origine. Unii dintre aceștia sunt nevoiți să călătorească și zeci de kilometri pentru a ajunge în locul special amenajat pentru vot, lucru care nu ar mai fi necesar odată cu utilizarea unei aplicații.

Pe lângă toate acestea, VV dorește să asigure o cale securizată de votare. Doar un singur utilizator are dreptul de a schimba permisiunile în legătură cu tabela în care se stochează voturile și rezultatul, iar imutabilitatea este asigurată. Acesta este *superuser*-ul. Administratorul va putea vizualiza datele, deoarece are acces la baza de date, însă nu le va putea modifica, pe când votanților și candidaților le va fi afișat doar rezultatul. Aceștia din urmă nu pot vedea alte voturi sau informații, ci doar rezultatul final afișat în locul sesiunii pentru care a votat.

2.3. Aplicații similare

În producție există deja diferite aplicații care au funcționalități similare cu VV.

Una dintre cele mai cunoscute aplicații de pe piață care permite crearea de sondaje este ***doodle.com*** [14]. Acest site este folosit lunar de mai mult de 30 de milioane de utilizatori [15]. Doodle permite programarea ședințelor sau întâlnirilor pentru un grup numeros de oameni, dar și întâlniri 1 la 1 în funcție de disponibilitatea celeilalte persoane. Există și un calendar, prin care utilizatorul își împărtășește disponibilitatea prin trimiterea unui link. Acesta permite rezervarea unei întâlniri cu persoana care deține calendarul. Pentru a putea crea sondaje, este nevoie de un cont, iar pentru a vota este necesar ca persoana să își introducă numele.

În figura din dreapta se poate observa rezultatul unei sesiuni de vot realizat în doodle.com. În acest exemplu, s-au votat ideile favorite pentru un giveaway. Votanții au avut posibilitatea de a alege mai multe idei, iar cea care a câștigat este ideea 5, care are 6 voturi.

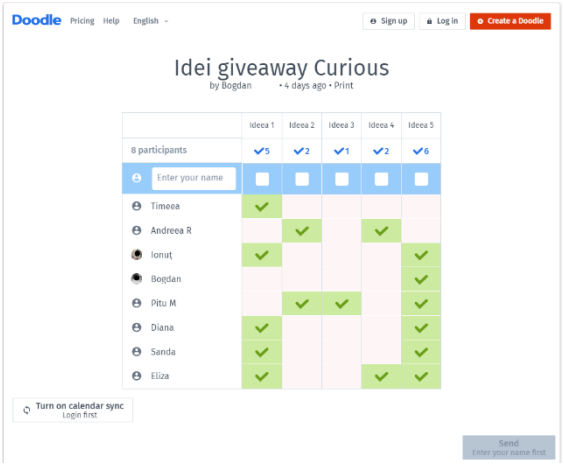

Figure 14 - Exemplu rezultat vot doodle.com

O altă aplicație populară, asemănătoare cu cea menționată anterior este ***StrawPoll*** [16]**.** Această aplicație are scopul de a face realizarea de sondaje cât mai ușoară, indiferent de numărul de votanți. Accentul se pune pe rezultat. Este important să se afle ce crede o majoritate în legătură cu un anumit subiect. Numele este unul sugestiv – *straw* și *poll*, adică sondaj de paie. Acestă expresie face aluzie la un obicei de a ridica în aer un fir de iarbă pentru a vedea în ce direcție bate vântul, o metaforă pentru direcția părerii grupului vizat.

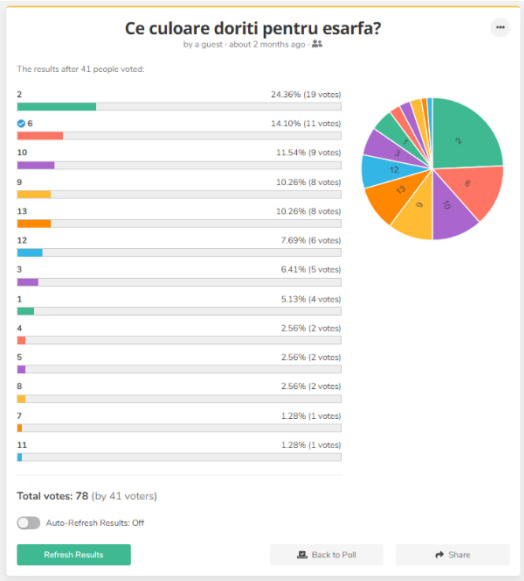

Figure 15 - Exemplu rezultat vot strawpoll.com

În figura din stânga este prezentat rezultatul unui sondaj realizat folosind aplicația *StrawPoll*, prin intermediul căruia s-a votat culoarea preferată pentru eșarfe. Votul a fost multimplu, votanții având posibilitatea de a alege mai multe rezultate.

Aplicații care merg pe aceeași idee sunt și ***Fast Poll*** [17], ***SurveyMonkey*** [18] sau ***Poll Maker*** [19]. Prima verifică securitatea cookie-urilor și a sesiunii, dă posibilitatea de a crea sondaje publice, dar și private, iar transmiterea lor se face prin intermediul unui link. ***SurveyMonkey*** are ca obiectiv țintă contextul business, operând pe un mediu de afaceri „*freemium*", adică este permisă folosirea instrumentelor de bază gratuit, însă utilizarea tuturor funcționalităților este contra cost. Cea din urmă, ***Poll Maker*** permite pe lângă realizarea de sondaje și realizarea de chestionare.

De asemenea, există site-uri oficiale precum ***Votes PA*** [20]. Acesta aparține guvernului din Pennsylvania și oferă o varietate de informații despre alegeri și despre drepturile omului. Utilizatorul poate să își creeze un cont și să voteze online prin intermediul acestuia. Este un site securizat. ***VotesPA*** oferă oportunitatea de deveni un contribuitor la procesul de realizare a sondajelor, prin completarea unui formular. Site-ul prezintă toate cerințele necesare, pozițiile și fișa postului.

Vote.gov [21] este un site care permite înregistrarea pentru a participa la alegeri. Acesta este destinat persoanelor din Statele Unite ale Americii. Utilizatorul alege statul din care face parte, și se înregistrează cu 21 de zile înainte de ziua alegerilor fie online, fie prin mail, fie personal.

Similar, ***Vote.org*** [22] permite înregistrarea pentru a vota și menționarea modului în care votantul va participa la acest proces. Din nou, este un site pentru Statele Unite ale Americii.

2.4. Tipuri de utilizatori

Aplicația are la bază trei tipuri de utilizatori: **administratorul**, **candidații** și **votanții**. Există și un **superuser** care are responsabilitatea de a accorda dreptul de administrator persoanei care solicită acestă permisiune, dacă dovada este validă și autentică.

Drepturile pe care le are fiecare clasă de utilizator sunt descrise în cele ce urmează.

2.4.1. Administrator

Administratorul își poate crea un cont însă este necesar să trimită o dovadă că face parte din organizație și ce post ocupă, poate crea o sesiune de vot, poate accesa sesiunile active de votare și poate face modificări în enunțul acestora. El are acces la lista cu utilizatorii înscriși în organizația din care face parte și are dreptul de a accepta sau respinge crearea conturilor pentru candidații și votanții care aparțin organizației respective. De asemenea, în cazul în care o persoană părăsește organizația, administratorul îi poate șterge contul sau în cazul în care un votant dorește să devină

candidat, administratorul trebuie să îi ofere accesul la pagina de candidatură după verficarea cererii. Acesta ***nu*** are dreptul de a schimba răspunsurile vontanților sau de a modifica candidaturile concurenților, doar le poate vizualiza.

2.4.2. Candidat

Candidatul este un votant care are dreptul să își depună candidatura și să o modifice. Pentru a deveni candidat, este necesar să îi fie confirmat accesul de către administrator. Aceasta se va face din pagina de profil, la apăsarea butonului „Doresc să candidez” care va trimite o cerere administratorului. Dacă cererea este acceptată, în locul „Doresc să candidez” va apărea „Modifică și vizualizează candidatura” și va avea acces la pagina aferentă.

Candidatul poate vizualiza propria candidatură, celelalte candidaturi, sesiunile de vot disponibile și poate vota. Candidatura constă într-un text însoțit sau nu, la alegerea candidatului, de poze și un videoclip. În momentul în care votează, acesta poate alege orice variantă din cele date, inclusiv se poate vota pe sine.

2.4.3. Votant

Votantul este utilizatorul standard. Acesta își poate crea un cont, are dreptul de a vizualiza pagina cu informații despre candidaturi, sesiunile curente de vot și poate vota. Alegerea este anonimă, iar votul va fi salvat astfel încât administratorul nu are accesul de a-l modifica.

2.5. Analiza sistemului informatic

- **Diagrame ale cazurilor de utilizare**

Reprezentarea schematică a principalelor funcționalități ale unui sistem informatic este realizată prin intermediul diagramei generale a cazurilor de utilizare. Programatorii au datoria de a identifica cererile în legătură cu soluția pe care o vor dezvolta și implementa pe baza acestei diagrame. Astfel, pentru cerințele identificate în aplicația VV, soluția informatică care a fost elaborată conține urmatoarele funcționalități:

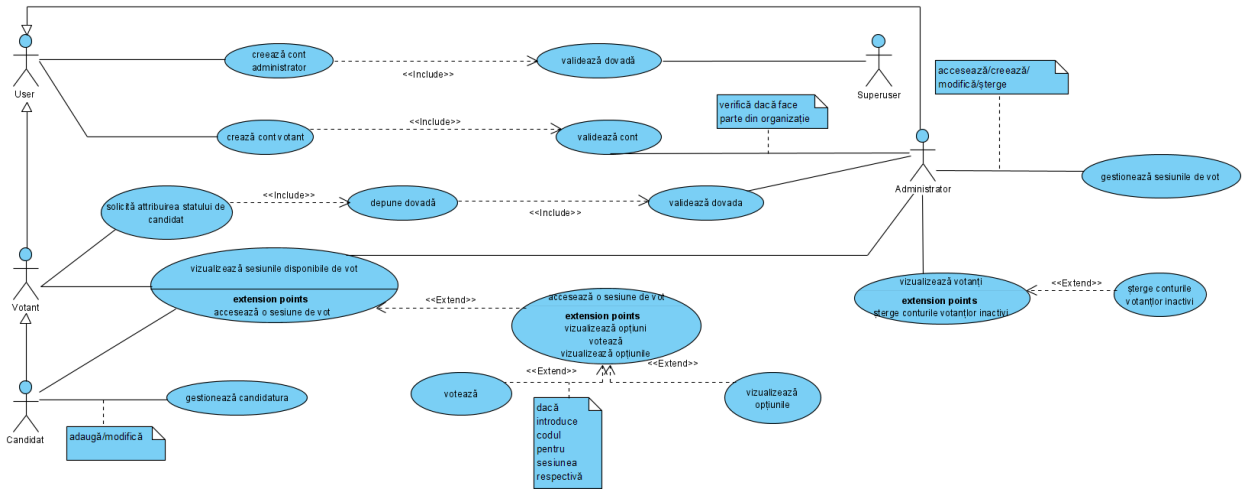

Figure 16 - Diagrama generală a cazurilor de utilizare

În diagrama de mai sus sunt prezentate principalele cazuri de utilizare a soluției informatice, iar în următoarele două diagrame sunt detaliate cazurile de utilizare pentru vizualizarea sesiunilor disponibile de vot(Figura 16) și, respectiv, pentru gestionarea sesiunilor de vot(Figura 14). **Diagrama generală a cazurilor de utilizare** modelează principalele cerințe funcționale pentru realizare aplicației și se pot identifica cele trei tipuri de utilizatori principali: **votantul**, **candidatul** și **administratorul**, dar și **superuserul** care are rolul de a acorda drepturl de administrator userilor.

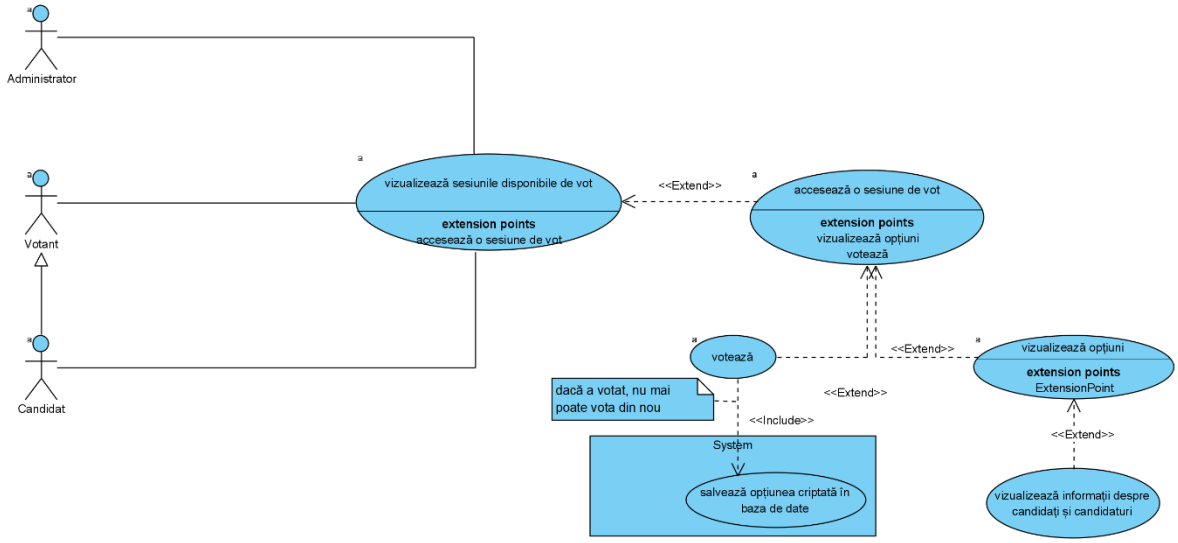

Figure 17 - Diagrama cazului de utilizare ,,vizualizează sesiunile disponibile de vot

În figura 17 este evidențiat faptul că toate cele trei tipuri de utilizatori pot vizualiza sesiunile de vot, le pot accesa și pot vota. De asemenea, aceștia pot vizualiza informații despre candidați, iar salvarea opțiunii alese se face de către sistem.

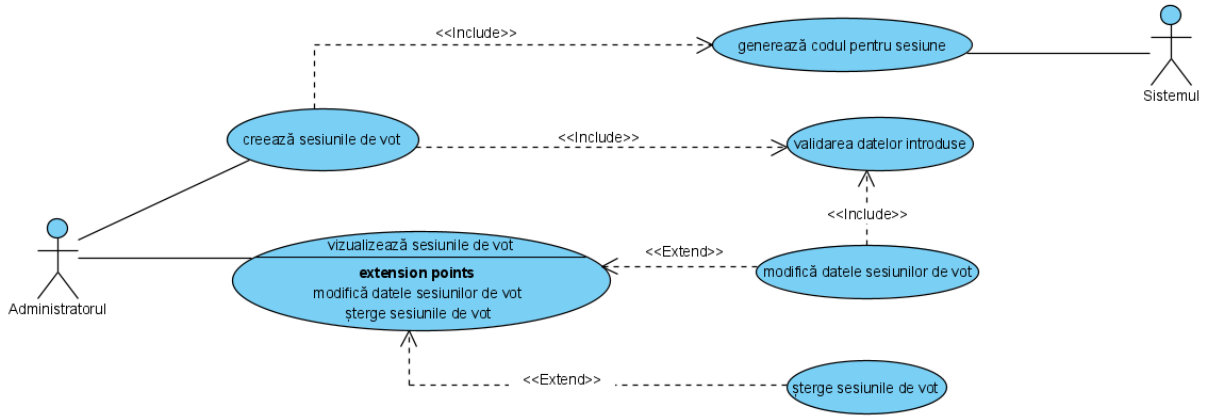

Figure 18 - Diagrama cazului de utilizare ,,gestionează sesiunile de vot"

În figura de mai sus este prezentată, din perspectiva administratorului, gestionarea sesiunilor de vot. Pe lângă cele menționate în diagrama anterioară, aici există și opțiunea de a crea și modifica o sesiune de vot. Validarea datelor se face automat, de către sistem, precum și generarea codului pentru acces.

- **Descrierea textuală a cazurilor de utilizare**

Table 1 - Descrierea cazului de utilizare ,,solicită atribuirea statutului de candidat"

Element al cazului de utilizare	Descriere
Cod	CU01
Stare	Schiță
Scop	Candidare
Nume	Solicită atribuirea statutului de candidat
Actor principal	Votant, Candidat
Descriere	Votantul dorește să își depună candidatura
Preconditii	Votantul are dreptul de a candida, adică a devenit candidat
Postconditii	Candidatura a fost depusă
Declansator	Votantul solicită să devină candidat
Posibile erori	Dovada nu este validă
Starea sistemului în caz de eroare	Candidatura nu a putut fi depusă

Flux de baza	1. Votantul se autentifică. [Curs alternativ A: votantul nu este autentificat] 2. Votantul solicită atribuirea statutului de candidat 3. Votantul încarcă dovada 4. Administratorul validează dovada 5. Votantul devine candidat 6. Candidatul își poate depune candidatura
Fluxuri alternative	A: 1. Utilizatorul este redirecționat către pagina de autentificare. 2. Utilizatorul se autentifică.
Relatii	-
Frecventa utilizarii	Medie

Table 2 - Descrierea cazului de utilizare ,,creează cont de administrator”

Element al cazului de utilizare	Descriere
Cod	CU02
Stare	Schiță
Scop	Crearea unui cont de administrator
Nume	Creează cont administrator
Actor principal	User, Superuser
Descriere	Userul dorește să își creeze un cont de administrator
Preconditii	Userul trebuie să își creeze cont și să introducă dovada că face parte din asociația respectiăe, împreună cu postul pe care îl ocupă
Postconditii	Userul devine administrator
Declansator	Userul solicită să devină administrator
Posibile erori	Dovada nu este validă
Starea sistemului în caz de eroare	Contul nu a putut fi creat
Flux de baza	1. Userul introduce dovada că aparține organizației și informațiile despre cont și postul pe care îl ocupă. 2. Superuserul analizează dovada și aprobă crearea contului. [Curs alternativ A: Dovada nu este validă] 3. Userul devine administrator
Fluxuri alternative	A: 1. Userul este înștiințat că nu i s-a putut acorda rolul de administrator.

Relatii	-
Frecventa utilizarii	Scăzută

Table 3 - Descrierea cazului de utilizare ,,accesează sesiune vot”

Element al cazului de utilizare	Descriere
Cod	CU03
Stare	Schiță
Scop	Votare
Nume	Accesează o sesiune de vot
Actor principal	Votant, Candidat, Administrator
Descriere	Utilizatorul dorește să acceseze o sesiune de vot
Preconditii	Utilizatorul este înregistrat
Postconditii	Utilizatorul a votat
Declansator	Utilizatorul accesează una dintre sesiunile de vot disponibile
Posibile erori	Utilizatorul nu este autentificat
Starea sistemului în caz de eroare	Votul nu a putut fi înregistrat
Flux de baza	1. Utilizatorul se autentifică. 2. Utilizatorul accesează o sesiune de vot prin introducerea unui cod. [Curs alternativ A: Utilizatorul nu este autentificat] 3. Utilizatorul vizualizează opțiunile. 4. Utilizatorul votează. [Curs alternativ B: Sesiunea aleasă de vot nu mai este disponibilă]
Fluxuri alternative	A: 1. Utilizatorul este redirecționat către pagina de autentificare. 2. Utilizatorul se autentifică. B: Utilizatorul este redirecționat către pagina de vizualizare a sesiunilor disponibile de vot.
Relatii	-
Frecventa utilizarii	Ridicată

- **Diagrama de clase**

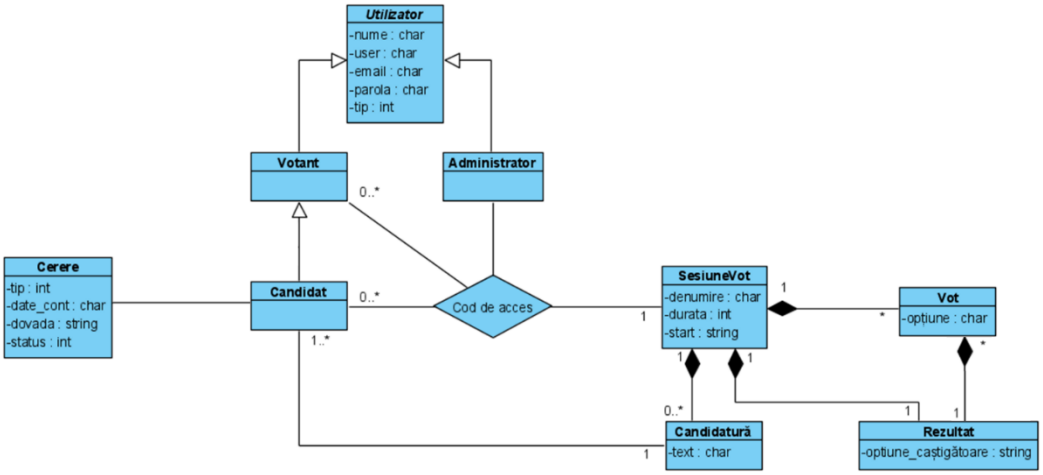

Figure 19 - Diagrama claselor

Pornind de la diagrama de clase din figura 19 va fi realizată ulterior baza de date, întrucât sunt prezente aici clasele împreună cu atributele lor. De asemenea, există și legăturile dintre tabele marcate prin multiplicitățile din figură.

- **Diagrama de stare**

De asemenea, tot pornind de la diagrama claselor, am putut realiza diagrama de stare. Aceasta este utilizată pentru a observa mai ușor stările prin care trece un obiect și pentru a putea analiza comportamentul entităților.

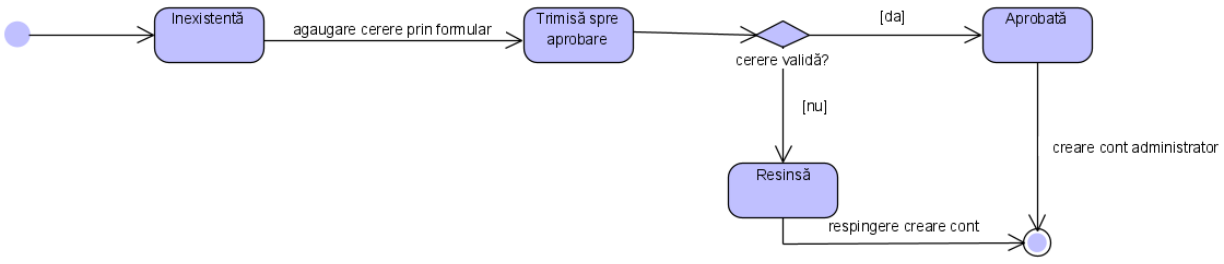

Figure 20 - Diagramă de stare pentru clasa Cerere

Prin figura 20 sunt evidențiate stările prin care poate trece o cerere: inexistentă, trimisă, aprobată/respinsă. Pe baza acestei cereri este creat sau nu un cont de administrator.

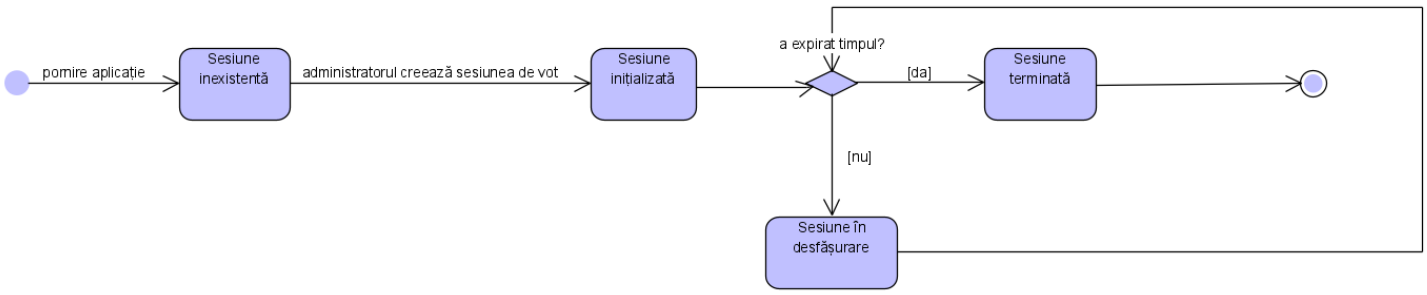

Figure 21 - Diagramă de stare pentru clasa SesiuneVot

De asemenea, din figura 21 se poate observa că o sesiune de vot trece prin următoarele stări: inițial, sesiune **inexistentă**, apoi după ce administratorul o creează devine **inițializată**, iar în funcție de timp poate fi **în desfășurare** sau **terminată**.

- **Diagrama de colaborare în BPMN**

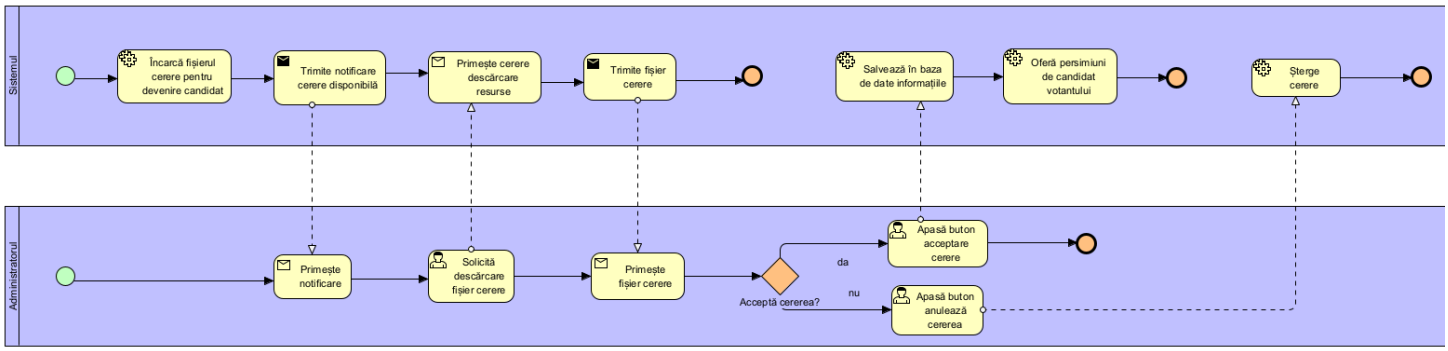

Figure 22 - Diagramă de colaborare devenire candidat

În figura 22 este reprezentată **diagrama de colaborare pentru a deveni candidat**. În cazul în care un votant dorește să candideze, are nevoie de acordul administratorului. Acesta decide dacă cererea trimisă este sau nu eligibilă.

- **Diagrama de procese în BPMN**

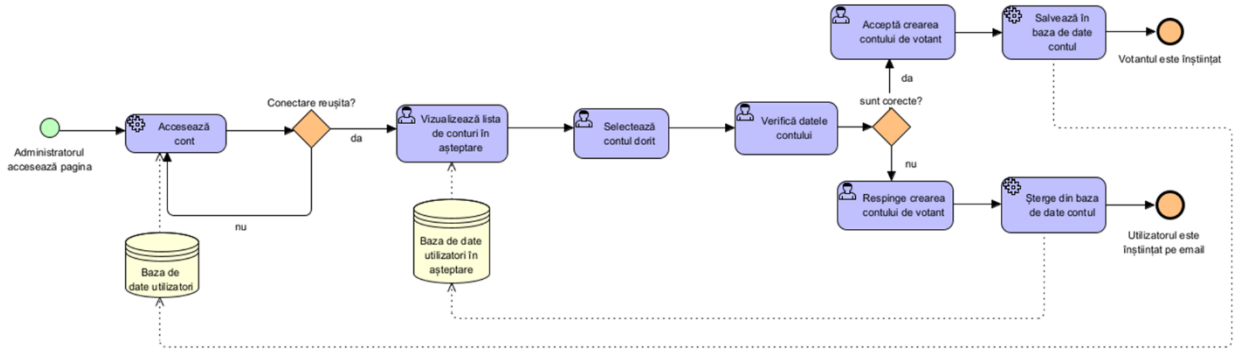

Figure 23 - Diagramă de procese creare cont de votant

Pentru a evidenția etapele care se urmează în cearea unui cont de votant, am construit diagrama de procese din figura de mai sus.

Administratorul are, printre altele, și atribuția de a gestiona conturile: a accepta sau a respinge crearea lor, a vizualiza conturile, a șterge conturile inactive.

- **Proiectarea bazei de date**

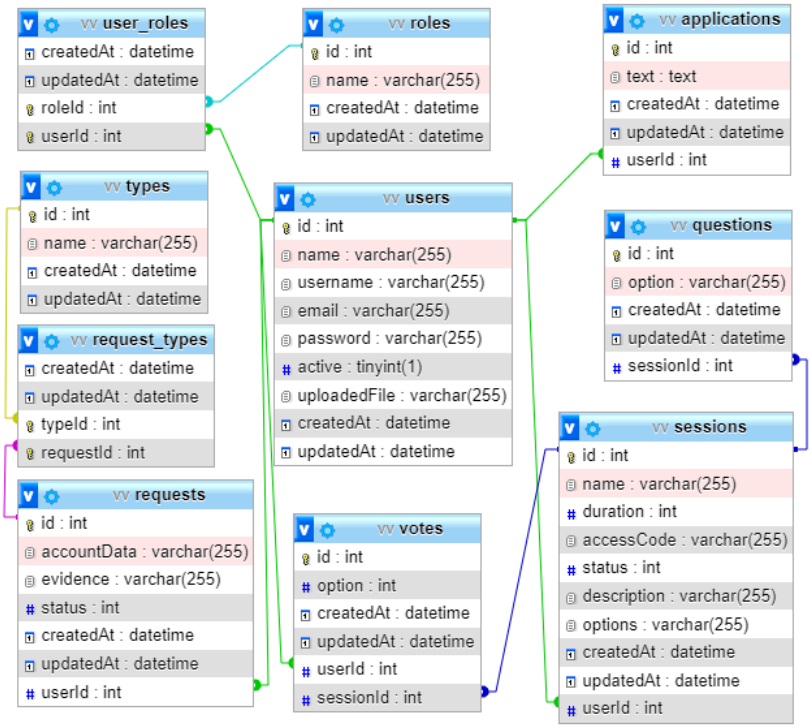

Figure 24 - Schema bazei de date generatî în phpMyAdmin

În figura anterioară, este reprezentată schema bazei de date generată în phpMyAdmin.

Aceasta este alcătuită din următoarele tabele:

- ***users***, corespondenta pentru UTILIZATOR în care vor fi stocate informațiile cu privire la numele, parola, e-mail-ul, numele de utilizator al acestuia, dar și atributul *active* cu ajutorul căruia, prin confirmarea administratorului, se va activa sau nu un cont;
- ***roles,*** tabela ROL, prin care sunt reținute denumirile tipurilor de utilizatori ai aplicației, alături de id-ul acestora;
- ***user_roles*** are rolul de a rezolva relația de tipul *many to many* dintre tabela utilizator și tabela rol. Aceasta va stoca ca și cheie străină id-ul utilizatorului și id-ul rolului;
- ***requests,*** CERERE**,** este creată pentru a gestiona cererile utilizatorilor: fie de a deveni candidat, fie de a deveni administrator. De accea, această tabelă se află în strânsă legătură cu tabela utilizator printr-o cheie străină;
- ***types,*** TIP**,** este o tabelă asemănătoare cu roles și gestionează cele două tipuri de cereri, având un id și o denumire;
- pentru a rezolva relația m:m dintre cerere și tip, am construit tabela ***request_types*;**
- deoarece utilizatorii au posibilitatea de a candida, am creat tabela CANDIDATURĂ (***applications***) pentru care va fi stocat textul și id-ul;
- ***sessions***, tabela SESIUNE aferentă sesiunii de vot, conține atributele acesteia (id, nume, durată, cod de acces, status, descriere) și este legată de tabela utilizator printr-o cheie străină, deoarce este necesară stocarea id-ului administratorlului care a creat sesiunea respectivă;
- fiecare sesiune are o singură întrebare reținută la nivelul atributului description din tabela sessiune, însă are mai multe răspunsuri pentru care a fost realizată tabela ***questions*** - ÎNTREBARE**.** Cu ajutorul acesteia, vom reține în baza de date fiecare răspuns și vom putea gestiona votul.
- ***votes***, tabela VOT pe baza căreia este calculat rezultatul final; un utilizator are un sigur vot pentru o sesiune.

Pe lângă atributele introduse manual, *Sequelize* salvează pentru fiecare tabelă și informații referitoare la momentul în care a fost creată instanța respectivă, dar și la momentul actualizării acesteia.

Am creat modelele pe care apoi le-am conectat între ele în fișierul *index.js* prin comenzi specifice *Sequelize*.

3. VALORIFICAREA TEHNOLOGIILOR UTILIZATE ÎN DEZVOLTAREA SOLUȚIEI

3.1. Tehnologii informatice utilizate

La început, paginile web erau realizate folosind **HTML**, un limbaj de marcare pentru crearea de pagini web care este folosit pentru a afișa informații în browser și **CSS**, un standard pentru înfrumusețarea intefeței prin formatarea elementelor scrise în fișierul HTML. Însă aceste două unelte nu au fost de ajuns, întrucât, pe parcurs ce tehnologia evolua, și cerințele utilizatorilor avansau. În timp, au început să apară paginile web dinamice folosind limbaje complexe precum PHP sau *Python*. Astfel, a apărut și ***JavaScript*** care avea ca scop principal să ofere dinamism paginilor, să le facă mai interactive într-un limbaj de programare accesibil. Acest lucru se întâmpla în data de 4 Decembrie 1995 prin ajutorul lui Brendan Eich, creatorul limbajului de programare *JavaScript*.

Chiar dacă nu a fost cunoscut și adoptat de la început, limbajul a ajuns să fie foarte folosit, fiind singura opțiune pe care *browserele* o oferă pentru a realiza interacțiunea dintre partea clientului și utilizatorului, acesta reprezentând standardul în domeniu. De asemenea, având la bază acest limbaj de programare au apărut ulterior unele dintre cele mai cunoscute și folosite *framework*-uri precum: *Angular, React.js, Vue.js, Node.js*. Așadar, operează atât pe partea de *front-end*, realizând interactivitatea elementelor HTML și schimbarea dinamică a conținutului paginilor, cât și pe partea de *back-end* fiind responsabil de operațiunile de tipul *server-side*. *JavaScript* este cel mai cunoscut limbaj de *scripting*, utilizat chiar și de Google și Facebook pentru a realiza site-uri interactive. Se mai folosește și pentru realizarea de jocuri și implementarea de aplicații *desktop,* dar și pentru consultarea și interogarea bazelor de date (*CouchDB, MongoDB*).

De-a lungul timpului, limbajul s-a aflat în continuă ascensiune și a fost supus unor modificări. O schimbare notabilă care a apărut destul de recent a fost în 2015, odată cu publicarea celei de a șasea versiuni, când au apărut cuvintele „*let*” și „*const*”. Aceste două cuvinte cheie ajută la declararea variabilelor - primul la nivel de bloc, iar cel de-al doilea, *const*, declară o variabilă constantă. Totodată, *JavaScript* 6 introduce și funcțiile de tip „*arrow functions*” și mai multe metode predefinite, precum: *find, findIndex*. S-a optimizat și modul în care se definește constructorul în cadrul unei clase, moștenirea, accesul la clasa de bază și la membrii statici.

JavaScript este, cu siguranță unul dintre cele mai utile limbaje de programare care evoluează continuu și se adaptează la cerințele pieței, fiind apreciat ca un limbaj multi-paradigmă.

3.2. Baza de date

Pentru a asigura persistența datelor într-o aplicație se utilizează bazele de date, iar în cadru VV am folosit o bază de date relațională, întrucât are ca avantaj accesul rapid la informațiile pe care le stochează într-un mod organizat. Pentru a accesa și manipula datele se folosește limbajul SQL(*Structured Query Language*), limbaj specific pentru bazele de date relaționale scris în C, C++.

Un sistem de gestiune a bazelor de date (SGBD) relațional *open-source* este *MySQL*, care a devenit popular prin utilizarea sa în cadrul aplicațiilor web, însă are și alte întrebuințări importante, întrucât este folosit în comerț electonic, depozitarea datelor și în cadrul aplicațiilor de înregistrare.

- **De ce MySQL?**

Cel mai popular SGBD, MySQL a fost produs de compania suedeză MySQL AB. Versiunea inițială a apărut pe 23 mai 1995, iar printre avantajele sale se numără: securitatea datelor, performanțe ridicate, scalabilitate, reducerea costurilor de proprietate, flexibilitatea de a fi *open-source* (documentație elaborată și suport rapid) și control complet asupra fluxului de lucru.

Pentru gestionarea unei baze de date SQL, am utilizat în aplicația VV librăria *Sequelize*.

- **Ce este Sequelize?**

Sequelize este un ORM (*Object-Relational Mapping*), adică mapează sintaxa obiectelor pe schema bazei de date, care se bazează pe promisiuni(*promises*) pentru *MySQL*, MariaDB, *Postgres* sau *Microsoft SQL Server.* Acest ORM dispune de un suport solid pentru tranzacții, relații, *lazi-loading* și *eager-loading,* etc. Pentru a instala *Sequelize* se folosește comanda `npm install –save sequelize` din cadrul utilitarului *Node Package Manager*.

3.3. Client

Pentru că ne aflăm într-o eră în care totul evoluează extrem de rapid, iar tehnicile de *front-end* (acea parte a unui site care îi este afișată utilizatorului) țin pasul cu succes, programatorii sunt obligați să își îmbunătățească constant metodele de lucru și cunoștințele. Așadar, familiarizarea rapidă cu noile unelte apărute este o necesitate.

Chiar dacă există deja posibilitatea de a proiecta și integra diferite elemente vizuale într-o pagină web folosind limbajul HTML(*Hyper Text Markup Language*) [23] și stilizarea acestora în mod independent prin CSS(*Cascading Style Sheet*) [23], se observă o nevoie de a face lucrurile mai complexe, deci necesitatea de a utiliza un *framework.*

- **Ce este un *framework*?**

Un *framework* este o structură conceptuală. Acesta reprezintă o arhitectură de *software* care are misiunea de a modela relațiile generale ale unui site, adică reprezintă o colecție de biblioteci și diferite instrumente care pot transforma sarcinile obișnuite în niște module reutilizabile. *Framework*-urile ajută la realizarea de aplicații interactive fără a fi nevoie de solicitarea unui server.

În cadrul VV este utilizat ***Vue.js***, un framework *open-source* creat de Evan You în perioada în care lucra pentru Google și lansat în luna februarie a anului 2014 [24] care poate crea aplicații *single-page* și care a ajuns popular într-un timp foarte scurt. Sintaxa este intuitivă și realizată într-o manieră în care îi permite să fie înțeleasă chiar și de programatorii începători, dar familiarizați cu conceptele de bază de *JavaScript*. Deși Evan You lucra cu Angular, a dorit să realizeze ceva cu dificultate mai redusă, dar cel puțin la fel de eficient. Pentru primele șase luni, *View* se numea *Seed* și inițial se concentra pe stratul *View* al MVC. Arhitectura s-a bazat pe componente care au fost introduse de *React*, iar sintaxa a urmat un stil similar cu AngularJS, creând astfel o legătură între cele două. Conform *State of Vue* [25], *framework-ul* este „un copil care a primit cele mai bune părți ale părinților săi”.

O aplicație de tipul *single-page,* prescurtat ca **SPA** este o aplicație web sau un site web care funcționează într-un *browser* și interacționează cu utilizatorul prin a rescrie dinamic pagina curentă cu informații noi de la server, adică nu se folosește metoda clasică prin care se încarcă pagini noi, întregi.

Aplicațiile *single-page* au scopul de a crea tranziții rapide astfel încât să facă site-ul să se comporte ca o aplicație nativă, iar utilizatorul nu trebuie să aștepte pentru încărcarea componentelor. De aici reiese principalul avantaj: viteza. Majoritatea resurselor necesare sunt încărcate în momentul lansării aplicației, deci nu mai este necesar să fie reîncărcate în timpul utilizării, iar singurul lucru care suferă modificări sunt datele trasmise de și către server. Astfel, nu se așteaptă comunicarea client-server tot timpul, ceea ce face tot procesul mai rapid.

Prima versiune de *Vue.js* a fost 0.8.0, iar cea care i-a marcat începutul măreț a fost 1.0, fiind un pas inovator pentru această bibliotecă. Transformarea lui *Vue.js* dintr-o bibliotecă *View-layer* în ceea ce numim în prezent cadru progresiv a constat în lanasare bibliotecilor precum: *vue-*

router (18.08.2015), *vuex* (28.11.2015), *vue-cli* (27.12.2015). O rescriere completă a cadrului a fost lansarea versiunii 2.0 în anul 2016, introducând concepte noi precum *Virtual DOM* și capacități *Server-Side Render-ing*. În septembrie 2020, după patru ani, apare versiunea 3.0 ce aduce funcții noi subliniate de *State of Vue* 2021 [26] : dimensiune mai mică cu 50% față de 2.6; rescrierea DOM-lui a dus la îmbunătățirea vitezei, făcându-l mai rapid; noua compoziție API duce la un cod mai curat și ușor de citit; permiterea redării de cod de la o componentă la alta prin funcția Tender.

- **De ce Vue.js?**

Chiar dacă este destul de nou, Vue.js a prins popularitate repede și a ajus să fie printre primele 3 cele mai populare *framework*-uri. Este creat într-un limbaj ușor de învățat, cu o sintaxă simplă, este flexibil și de dimensiuni reduse, oferă posibilitatea de a crea ușor o aplicație și de a integra diferite funcționalități. Vue.js este versatil și ajută la multiple sarcini: de la realizarea unei aplicații web și mobile, până la aplicații progresive, are puterea de a gestiona atât procese simple, cât și procese dinamice.

Totodată, *framework*-ul vine cu o documentație detaliată și extinsă pe care am consultat-o în procesul de realizare al aplicației.

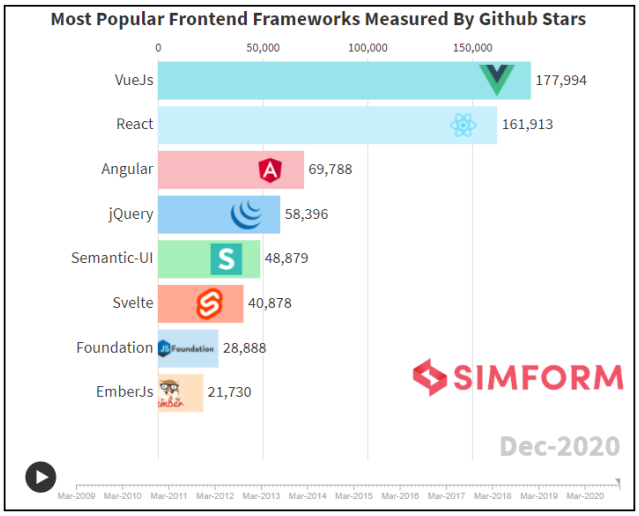

Figure 25 - Clasament GitHub

Sursa imagine: [27]

În figura anterioară este reprezentat topul celor mai populare *framework*-uri în funcție de recenziile de pe GitHub din Decembrie 2020. Se poate vizualiza [28] evoluția acestora de la începutul anului 2009 până la finalul anului 2020. În august 2016, *Vue.js* a ajuns pe locul 3 în acest top, unde s-a situat până în martie 2017. Din luna iulie, acesta a ajuns pe primul loc și s-a menținut până în decembrie 2020 inclusiv.

Din punctul meu de vedere, *Vue* este un *framework* potrivit pentru aplicația VV, întrucât nu ocupă mult spațiu, se randează ușor, având un start mai rapid decât celelalte *framework*-uri, iar curba de învățare este mai mică. De asemenea, trecerea de la elemente mai simple, precum utilizarea sintaxei de HTML sau CSS, la elemente mai complexe (funcții, funcționalități) se face destul de ușor.

Creatorul Redux-ului, Dan Abramov, are un citat *„bibliotecile de flux sunt ca niște ochelari, vei știi când vei avea nevoie de ele*" [29]. Un astfel de instrument care este folosit în aplicația de vot VV este *Vuex*.

Vuex este o bibliotecă care este recomandată pentru aplicațiile cu un flux de date mediu sau mare și are rolul de a administra starea aplicațiilor dezvoltate folosind *Vue.js*. Acesta servește ca un magazin centralizat pentru toate componentele din care este alcătuită o aplicație, dispunând de niște reguli care se asigură că starea poate fi schimbată doar într-un mod predictibil.

Vuex este alcătuit după principiul clasic MVC (*Model, View, Controller*) din 3 părți: *state, view* și *action,* însă odată cu apariția mai multor elemente care împărtășesc o stare comună, simplitatea dispare, întrucât există posibilitatea ca mai multe *views* să depindă de aceeași stare (*state*) sau diferite acțiuni(*actions*) să solicite modificarea aceleași părți de stare.

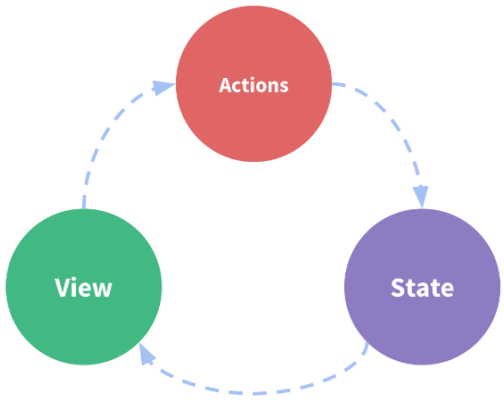

Figure 26 - Parți componente Vuex

Sursa: [30]

Desigur, pentru a obține produsul dorit, avem nevoie și de alte unelte care să vină în ajutorul framework-ului. Un astfel de exemplu este **Bootstrap**.

- **Ce este Bootstrap?**

Bootstrap este o colecție de unelte HTML, CSS și JavaScript pentru a crea aplicații și pagini web. A fost creat inițial de Mark Otto și Jacob Thornton pentru Twitter ca un proiect *open-*

source găzduit pe GitHub pentru a încuraja coerența între uneltele interne și lansat pe data de 19 August 2011 sub numele *Twitter Blueprint*. Începând cu aprilie 2021, *Bootstrap* a devenit al zecelea cel mai plăcut proiect de pe GitHub, având peste 150000 de stele.

3.4. Server

Pentru a implementa complet o aplicație nu este suficientă doar partea de client, ci este nevoie și de cea de server. Cele două se completează și au același scop comun.

Node.js este o multiplatformă *open-source* pentru dezvoltarea de aplicații de server și rețea, care oferă posibilitatea de a rula și testa codul de *JavaScript* în afara *browser*-ului de internet. A fost dezvoltată de Ryan Dahl în 2009 și este compatibilă cu Windows, Linux și OS X.

Un server de tip *Node.js* este de tip **REST** și se bazează pe API. Arhitectura REST este compusă dintr-un set de constrângeri cu scopul de a realiza servicii web, cunoscute sub numele de servicii web RESTful. Interacțiunea între sisteme se realizează prin protocolul HTTP, fără a necesita implementarea de funcții adiționale.

Arhitectura oferă sistemelor dreptul să acceseze și manipuleze date conform unui set reguli prestabilite. Într-un sistem *RESTful* este necesar un Client și un Server. **Clientul** este independent și are rolul de a solicita informații (resurse) pe care **Server**-ul le deține. Cele două componente sunt separate și se pot dezvolta diferit, fără a se influența una pe cealaltă. Acest lucru oferă simplitate. Este ușor de implementat de către programatori, iar programele *software* interpretează convenabil rezultatul.

API este o prescurtare de la *Application Programming Interface*, adică un intermediar *software* care permite comunicarea între mai multe aplicații. Acest intermediar este folosit în cadrul aplicațiilor pentru a simplifica programarea prin abstractizarea implementării de bază. Se expun doar acele obiecte sau acțiuni care sunt folositoare dezvoltatorului. Atfel, livrarea serviciilor și transmiterea informațiilor este flexibilă.

- **De ce *Node.js*?**

Node.js are multiple caracteristici care îl fac prima alegere a arhitecților *software*, precum:

- un server bazat pe *Node.js* nu așteaptă un API să returneze datele, ci trece la următorul după ce îl apelează. Există un mecanism de notificare pentru evenimente care ajută serverul să obțină răspunsul apelului anterior. Astfel, API-urile sunt **asincone**;

- este integrat în motorul de *JavaScript* al *Google Chrome* și este **foarte rapid** la execuția codului;
- chiar dacă folosește un singur fir de execuție, este **scalabil**. Oprează diferit de serverele tradiționale care creează fire de execuție limitate pentru gestionarea cereilor. Faptul că dispune de un mecanism de evenimente, ajută serverul să răspundă rapid, fără să se blocheze.
- aplicațiile Node.js **nu memorează datele**, ci le reproduc pur și simplu, în bucăți
- este sub **licența MIT** (https://raw.githubusercontent.com/joyent/node/v0.12.0/LICENSE)

De asemenea, printre avantajele librăriei *Node.js* se numără și nivelul înalt de performanță, ușor de înțeles și ușor de învățat, dispune de un număr larg de biblioteci existente, este disponibil pentru multiple platforme și este un mediu *open-source*.

Odată cu utilizarea Node.js, se va folosi implicit și ***node package manager(npm)*** pentru a instala, folosind o linie de comanda care poate interacționa cu un registru la distanță, dependințele locale sau instrumentele la nivel global pentru orice tip de proiect. De asemenea, *npm* este privită și ca o platofrmă care oferă posibilitatea de a căuta instrumente deja implementate în *JavaScript* sau chiar publicarea unor implementări personale.

Cel mai popular framework de Node.js, ***Express.js*** [31], se instalează cu ajutorul *node package manager,* folosind comanda `npm install express –save`. După rularea acesteia, va fi introdusă în fișierul `package.json` dependința *express* alături de versiunea care permite funcționarea serverului existent cu *express*.

Express.js oferă un set de caracteristici pentru dezvoltarea aplicațiilor, permițându-le dezvoltatorilor să creeze și administreze servere. Scopul principal pentru care este utilizat este de a oferi logică pe partea de server pentru aplicații *web* sau mobile și vine cu multiple funcționalități incorporate. Are, totodată, o suită de programe terțe care pot fi folosite spre a îmbunătăți funcționalitățile, securitatea și viteza.

Majoritatea dezvoltatorilor preferă să folosească *Node.js* pentru viteza sa, iar Express.js este ca o completare care se suprapune fără a scădea performanța. Acest framework este o soluție facilă pentru aplicațiile de tip *single-page,* site-uri web sau API-uri HTTP publice.

Un alt *framework* regulat folosit împreună cu *Node.js* este *Vue.js*. Cele două folosesc același limbaj de programare - *JavaScript* și au o structură ușor de înțeles.

Utilizând toate tehnologiile menționate anterior, am reușit să creez o aplicație de vot - aplicația VV.

4. IMPLEMENTAREA SOLUȚIEI INFORMATICE

Soluția informatică este destinată uzului în cadrul organizațiilor și asigură un cadru ordonat și sigur pentru procesul de votare, fie că este vorba de o simplă organizare a unui eveniment, fie că este vorba de alegerea biroului de conducere. Poate fi considerat ca o unealtă internă utilă în procesul de luare a deciziilor.

Aplicația are la bază principiile arhitecturii REST, fiind astfel realizată pe două planuri separate. Este vorba de back-end și de front-end. Acestea sunt testate separat: **planul clientului** poate fi vizualizat și probat direct prin intermediul unei extensii (*Vue.js devtools*) în *browserul* web, vizual, pe când pentru **planul serverului** se folosește *Postman*, un instrument prin intermediul căruia pot fi trimise solicitări.

REST este o prescurtare de la termenul din limba engleza, *Representational state transfer*, care are ca și caracteristică principală faptul că facilitează comunicarea sistemelor între ele, fiind un stil arhitectural prin intermediul căruia sunt furnizate diferite standarde între sistemele informatice *web*. Această arhitectură separă cele două planuri: cel al clientului de cel al serverului fără a fi nevoie ca unul dintre acestea să aibă informații despre starea în care se află celălalt.

Prin arhitectura REST sunt impuse un set de constrângeri pe care sistemele *RESTful* (acele sisteme care utilizează arhitectura REST) trebuie să le respecte. La baza acestei arhitecturi stau operațiile **CRUD** care sunt efectuate pe baza de date, fiind mapate prin multiple apeluri ale metodelor HTTP. În tabelul de mai jos sunt reprezentate cele patru operații corelate cu metoda HTTP specifică.

Table 4 - Operații CRUD și metodele HTTP corespondente

Operația CRUD	Metoda HTTP
CREATE – pentru a crea o resursă	POST
READ – pentru a selecta o resursă	GET
UPDATE – pentru a actualiza o resursă	PUT
DELETE – pentru a șterge o resursă	DELETE

Sursa: adaptare folosind informațiile din [32]

4.1. Implementarea aplicației client

Partea de client a aplicației este reprezentată de partea vizuală a aplicației care îi dă utlizatorului puterea de a modifica modul în care îi sunt afișate informațiile, precum a fost prezentat și în capitolul 3.3 al lucrării, și este creată după cele trei tipuri de utilizatori principali: **votant**, **candidat** și **administrator**. Astfel, aplicația VV va fi prezentată pe rând din perspectiva celor trei.

Inițial, am creat aplicația *vue.js* folosind comanda **`vue create my-app`** în terminal. Am folosit configurările implicite, deoarece este o cale rapidă în prototiparea unui proiect nou.

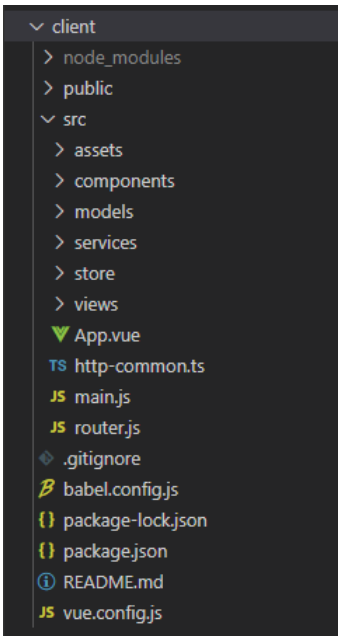

Figure 27 - Structura dosarului „client"

Structura dosarului „**client**" arată precum în figura 27.

Fișierul ***main.js*** este folosit pentru a importa toate bibliotecile și unelete necesare în cadrul aplicației, precum *Axios, Bootstrap, Fontawesome, VeeValidate*.

În **router.js** există rutele care sunt folosite pentru a redirecționa cererile către funcțiile corespunzătoare ale *controlerului*. În fișierul *App.vue* este păstrată componenta rădăcină a aplicației.

De asemenea, există și fișiere care se formează automat, precum *babel.config.js* (implicit, este directorul curent aflat în lucru și are rol de director *root)*, dar și fișiere care apar după utilizarea comenzii `npm install` în terminal în dosarul corespunzător clientului, precum *package-lock.json* sau *package.loc* în care sunt scrise dependințele proiectului pentru a ușura reinstalarea.

Urmează apoi dosarele intitulate „components" , „models" și „views" care împreuna cu baza de date și rutele reușesc să asigure funcționalitatea și vizibilitatea părții clientului. Acestea comunică precum în figura 28: se trimite o cerere de tip HTTP către rută care o direcționează controlerului potrivit, apoi controalerele afișează cu ajutorul *views* datele preluate din baza de date de către modele sau se folosesc de modele pentru a transmite informații în baza de date, iar, ulterior, trimit răspunsul HTTP.

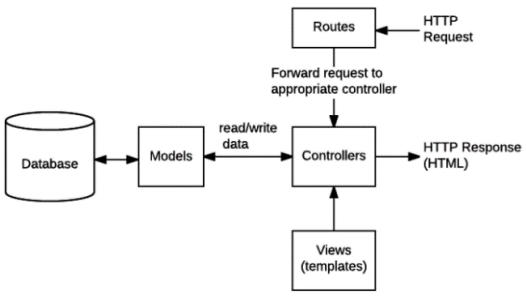

Figure 28 - Cerere - răspuns HTTP

Sursă imagine: [33]

- **Votant**

Pentru procesul de autentificare există pagina de înregistrare și pagina prin care utilizatorul poate intra în cont.

În figura 29 este reprezentată funcția handleRegister, utilizată în pagina de înregistrare pentru a prelua datele din formular și a le verifica. În cazul în care apare vreo problemă, s*erverul* va trimite eroarea printr-un mesaj. Dacă datele introduse sunt valide, prin *axios* se vor transmite informațiile referitoare la utilizator către baza de date împreună cu mesajul de succes și codul 200.

```
handleRegister() {
  const formData = new FormData();
  formData.append("file", this.file);
  formData.append("username", this.user.username);
  this.message = "";
  this.submitted = true;
  this.$validator.validate().then((isValid) => {
    if (isValid) {
      this.$store.dispatch("auth/register", this.user).then(
        (data) => {
          this.message = data.message;
          axios.post(
            "http://" +
              window.location.hostname +
              ":8080/api/auth/register",
            formData
          );
          this.successful = true;
        },
        () => {
          (error) => {
            this.message =
              (error.response && error.response.data) ||
              error.message ||
              error.toString();
            this.successful = false;
          };
        }
      );
    }
  });
},
```

Figure 29 - Cod sursă pentru funcția handleRegister

În variabila *formData* va fi reținută cererea în format .pdf, .jpg, .jpeg introdusa de utilizator pentru crearea unui cont de tip administrator.

Funcția *handleRegister*() este apelată în partea de HTML prin *@submit.prevent* la nivelul formularului.

- **Candidat**

Candidatul este un votant care are ca și permisiune suplimentară accesul la pagina de candidatură. Această pagină este construită dintr-un editor de text, iar codul HTML aferent acestuia poate fi vizualizat în figura 30.

Prin *ref* se face o referință la elementul-copil „*myQuillEditor*” din cadrul aplicației.

```
<template>
  <div class="container">
    <header class="jumbotronn">
      <h3 class="info">
        Adaugă candidatura
      </h3>
    </header>
    <quill-editor
      ref="myQuillEditor"
      v-model="content"
      :options="editorOption"
      @blur="onEditorBlur($event)"
      @focus="onEditorFocus($event)"
      @ready="onEditorReady($event)"
    />

    <div class="butonas">
      <button class="btnnn btn-primary btn-block">
        <span><strong>Trimite candidatură</strong></span>
      </button>
    </div>
  </div>
</template>
```

Figure 30 - Cod HTML editor text

Se poate observa faptul că este folosită directiva *v-model* pentru a asigura legarea datelor în ambele sensuri între informațiile referitoare la candidatură pe care le introduce utilizatorul și editorul de text.

Totodată, sunt utilizatele și evenimentele *blur* (executat în momentul în care se pierde concentrarea pe editor), *focus* (afișează in consolă mesajul „*editor focus!*” de fiecare dată în care se apasă cu mouse-ul pe editor) și *ready* (prin care se anunță în consolă ca editorul de text este pregătit pentru a fi folosit).

- **Administrator**

Pe lângă celelalte funcționalități prezente pentru votant, interfața administratorului dispune de o pagină de **gestionare a conturilor** și o pagină pentru **gestionarea sesiunilor de vot**.

```
<ul class="list-group">
  <li
    class="list-group-item"
    :class="{ active: index == currentIndex }"
    v-for="(user, index) in users"
    :key="index"
    @click="setActiveUser(user, index)"
  >
    {{ user.name }} - {{ user.active ? "Activ" : "Inactiv" }}
  </li>
</ul>
```

Figure 31 - Crearea listei pentru vizualizarea conturilor

În figura alăturată se poate vizualiza codul HTML aferent al unei liste de obiecte – lista conturilor. Prin evenimentul de *click*, se apelează funcția *setActiveUser(user, index)* care preia datele și indexul pentru utilizatorul selectat din listă.

Pentru a gestiona mai rapid **conturile**, am implementat și funcționalitatea de căutare în listă, iar în imaginea alăturată este afișat codul sursă pentru aceasta. Se poate observa faptul că este compusă dintr-un *input* și un buton, ambele având atributele *type* și *class*. Directiva *v-model* este utilizată pentru a crea o legătură bidirecțională și alege automat modul corect în care elementul se actualizează în funcție de tipul de intrare.

```
<div class="list row">
  <div class="input-group mb-3">
    <input
      type="text"
      class="form-control"
      placeholder="Caută după nume"
      v-model="name"
    />
    <div class="input-group-append">
      <button
        class="btn btn-outline-secondary"
        type="button"
        @click="searchName"
      >
        Caută
      </button>
    </div>
  </div>
```

Figure 32 - Codul pentru căutarea în lista de conturi

La *click*-ul pe buton este apelată automat funcția searchName. Aceasta caută utilizatorul dorit după nume. Dacă îl găsește, îl retunează (îl afișează în listă). Dacă nu îl găsește, afișează eroarea la consolă.

Pentru a crea o **sesiune de vot**, administratorul are la dispoziție un formular special.

4.2. Implementarea soluției la nivel de server

Pentru partea de server, structura de fișiere este următoarea:

În dosarul *uploads* sunt stocate cererile pentru a deveni administrator sau candidat.

Models conține fișiere cu fiecare model, adică o abstractizare care reprezintă o tabelă din baza de date, împreună cu legăturile dintre acestea.

Toate acestea componente interacționează între ele așa cum a fost menționat la începutul capitolului 4 al lucrării.

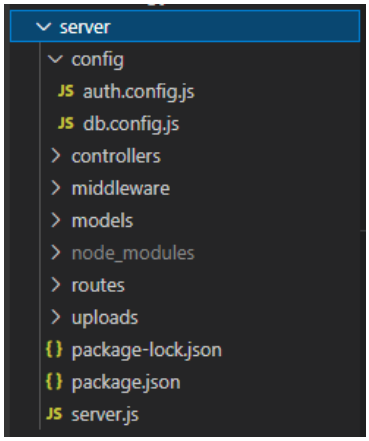

Figure 33 - Structura dosarului „server"

- **Serverul**

Pentru a avea tot ceea ce este necesar pentru buna funcționare a aplicației, am început prin crearea serverului de NodeJS, care a fost rulat cu succes pe portul 8080.

Inițial, am importat toate depenințele necesare, precum *express*, *bodyParser*, *md5*, *fileUpload*, am importat fișierele din dosarul cu rutele, am setat portul astfel încât aplicația să ruleze cu succes pe protul 8080 și am făcut o funcție pentru a inițializa tipurile de utilizatori din aplicație: votant, candidat, administrator.

După crearea serverului, am realizat baza de date întrucât va fi utilizată ulterior în cadrul celorlalte controalere. Pentru acest lucru, am avut nevoie de *Sequelize*.

În figura 34 este reprezentat fișierul *db.config.js*. Aici am introdus contul de SQL, hostul pe care va rula aplicația, numele bazei de date, dialectul și informații referitoare la *pooling*.

Utilizând datele din acest fișier serverul se va putea conecta la baza de date pentru a face ulterior diferite operații.

server > config > db.config.js > ...

```
1  module.exports = {
2      HOST: "localhost",
3      USER: "root",
4      PASSWORD: "root",
5      DB: "vv",
6      dialect: "mysql",
7      pool: {
8        max: 5,
9        min: 0,
10       acquire: 30000,
11       idle: 10000
12     }
13   };
```

Figure 34 - Codul fișierului db.config.js

- **Autentificare**

```
exports.signin = (req, res) => {
  User.findOne({
    where: {
      username: req.body.username,
    },
  })
    .then((user) => {
      if (!user) {
        return res.status(404).send({ message: "User Not found." });
      }

      var passwordIsValid = bcrypt.compareSync(
        req.body.password,
        user.password
      );

      if (!passwordIsValid) {
        return res.status(401).send({
          accessToken: null,
          message: "Invalid Password!",
        });
      }

      var token = jwt.sign({ id: user.id }, config.secret, {
        expiresIn: 86400, // 24 hours
      });

      var authorities = [];
      user.getRoles().then((roles) => {
        for (let i = 0; i < roles.length; i++) {
          authorities.push("ROLE = " + roles[i].name.toUpperCase());
        }
        res.status(200).send({
          id: user.id,
          name: user.name,
          username: user.username,
          email: user.email,
          roles: authorities,
          accessToken: token,
        });
      });
    })
    .catch((err) => {
      res.status(500).send({ message: err.message });
    });
};
```

Figure 35 - Cod sursă signIn

După ce și-a creat contul, utilizatorul se poate autentifica prin interfața pentru introducerea datelor contului, însă, toată acțiunea se petrece în spate de fapt.

În figura 35 este reprezentată funcția destinată autentificării, împreună cu validările pentru datele introduse.

Dacă nu a fost găsit utilizatorul în baza de date, serverul va trimite o eroare având codul 404 și mesajul „*User not found*”. Daca parola nu este validă, eroarea va fi 401 - „*Invalid Password*!”.

Dacă nu există erori, utilizatorul va accesa contul, iar serverul va transmite codul 200 împreună cu datele contului. Se primește un token valabil 24 de ore. La expirarea acestui interval, utilizatorul nu va mai avea acces la pagini și nu va mai putea vota.

- **Înregistrare**

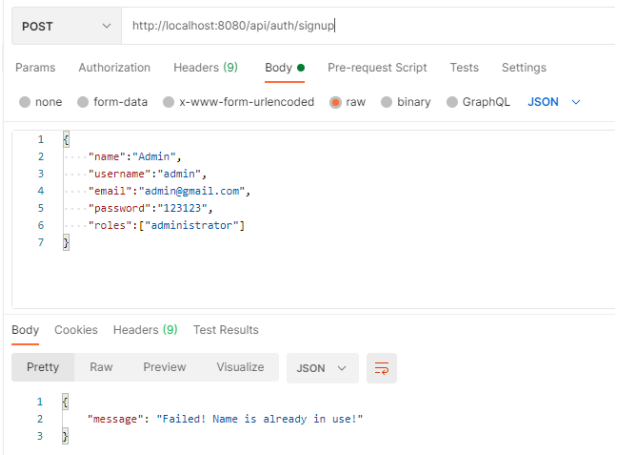

Figure 37 - Creare cont administrator Postman

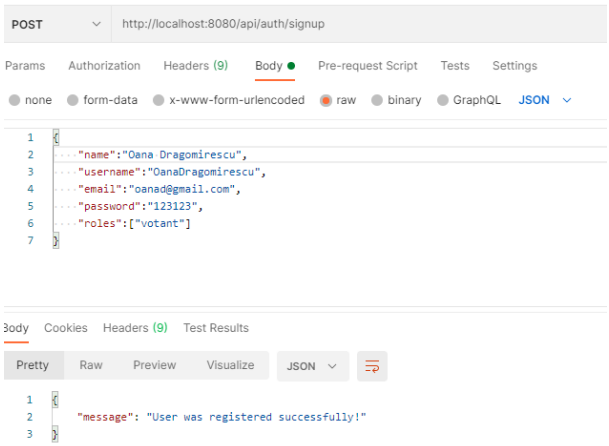

Figure 36 - Creare cont votant Postman

Pentru a testa faptul că totul este funcțional fără a utiliza partea de client a aplicației, am optat pentru *Postman*. În cele două imagini anterioare am folosit *Postman* pentru a testa crearea unui

cont de administrator, respectiv, crearea unui cont de votant. În primul caz, utilizatorul „*admin*” există deja în baza de date, așa că serverul returnează eroare și mesajul „*Failed! Name is already in use*”, pe când, în cel de-al doilea caz, datele introduse respectă toate criteriile, iar contul este creat cu succes.

```
[
    {
        "id": 1,
        "name": "Admin",
        "username": "admin",
        "email": "admin@gmail.com",
        "active": true,
        "uploadedFile": null
    },
    {
        "id": 2,
        "name": "Oana Dragomirescu",
        "username": "wannawdo",
        "email": "wannawdo@gmail.com",
        "active": true,
        "uploadedFile": null
    },
    {
        "id": 3,
        "name": "Bianca Nitu",
        "username": "bibboo",
        "email": "nitu.abianca@gmail.com",
        "active": true,
        "uploadedFile": null
    },
    {
        "id": 4,
        "name": "Cristian Munteanu",
```


Figure 38 - Vizualizare conturi în Postman

Folosind *Postman*, în figura 38 am făcut o operație de tip GET către server pentru a vizualiza toți utilizatorii din baza de date. Aceștia sunt afișati după id, fără a le putea vizualiza parola criptată pentru a asigura o securitate sporită aplicației.

Atributul „*active*” este folosit pentru a arăta stadiul în care se află contul: dacă este *true*, atunci contul este valid, dacă este *false*, atunci contul nu este valid.

Totodată, cu ajutorul atributului „*uploadedFile*” se va reține denumirea cererii în cazul în care utilizatorul dorește să își creeze cont de administrator.

- **Sesiuni vot**

Deoarece un candidat nu poate candidat simultan decât pe un singur post de conducere, crearea și actualizarea candidaturii are loc în același *controller*, foslosind o operație de tipul **PUT**.

```
exports.upsert = async (req, res) => {
  try {
    const { text, accessToken } = req.body;
    const userId = jwt.verify(accessToken, config.secret).id;
    const candidatura = await Application.findOne({ where: { userId } });
    if (!candidatura) {
      await Application.create({ text, userId });
      res.status(201).send({ message: "Candidatura a fost creata" });
    } else {
      await candidatura.update({ ...candidatura, text });
      res.status(200).send({ message: "Candidatura a fost actualizata" });
    }
  } catch (err) {
    res.status(500).send({
      message: err.message || "Some error occurred while retrieving users.",
    });
  }
};
```

Figure 39 - Cod sursa metoda upsert

Astfel, se va căuta candidatura în baza de date în funcție de id-ul utilizatorului. Daca nu a fost găsită, aceasta va fi creată și se va returna codul 201, împreună cu mesajul „Candidatura a fost creată”. Dacă utilizatorul cu id-ul dorit are deja o candidatură, o va găsi în baza de date și o va actualiza, apoi serverul va returna codul 200 și mesajul „Candidatura a fost actualizată”.

- **Vot**

Pentru a calcula rezultatele, am folosit o funcție JavaScript care folosește un *forEach* pentru a parcurge un *array* în timp ce numără într-o variabilă fiecare tip de vot(de câte ori se repetă fiecare element din vector).

```
289     validate() {
290       const votes = this.poll.answers
291         .filter((answer) => answer.voted == true)
292         .map((answer) => answer.id).length;
293       if (votes > 0) {
294         if (votes > 1) {
295           if (this.poll.multipleVotes == true) {
296             this.isValid = true;
297           } else {
298             this.isValid = false;
299           }
300         } else {
301           this.isValid = true;
302         }
303       } else {
304         this.isValid = false;
305       }
306     },
```

Figure 40 - Codul sursă pentru funcția validate

Funcția *validate* din figură se asigură că nu sunt bifate simultan mai multe răspunsuri, întrucât fiecare utilizator poate alege doar o singură opțiune.

Constanta *votes* reține dimensiunea *array*-ului cu opțiunile selectate. Dacă este 0, nu va putea vota. Daca este mai mare decât 1, înseamnă că ne aflăm în cazul în care avem vot multiplu, deci la apăsarea butonului „Votează” operațiunea nu va avea succes, iar utilizatorul va fi anunțat. Altfel, daca avem o singură opțiune bifată, atributul *isValid* devine true și va permite transmiterea votului către server și stocarea în baza de date.

- **Candidatura**

Deoarece este nevoie să poată fi creată, acualizată și ștearsă, pentru pagina destinată candidaturii au fost definite următoarele rute:

```
1   const { authJwt } = require("../middleware");
2   const user = require("../controllers/user.controller.js");
3   const candidatura = require("../controllers/candidatura.controller.js");
4
5   module.exports = function (app) {
6     app.use(function (req, res, next) {
7       res.header(
8         "Access-Control-Allow-Headers",
9         "x-access-token, Origin, Content-Type, Accept"
10      );
11      next();
12    });
13
14    app.get("/candidaturi", candidatura.findAll);
15
16    app.get("/candidaturi/:accessToken", candidatura.findOne);
17
18    app.delete("/candidaturi/:id", candidatura.deleteOne);
19
20    app.delete("/candidaturi/all", candidatura.deleteAll);
21
22    app.put("/candidaturi", candidatura.upsert);
23  };
```

Figure 41 - Rute pentru pagina ,,Candidatură"

Axios interacționează cu metodele HTTP *get, post, delete, put* precum a fost menționat și la începutul capitolului care pot fi observate și în imaginea de mai sus. Informațiile esențiale necesare

efectuării operațiunilor cerute se transmit prin parametrii URL, iar datele extinse (conținutul pentru o candidatură nouă/actualizată) se transmit în corpul *request*-ului ca *form data*.

- **Profilul meu**

Pentru a afișa imaginile pe cartonașul cu profilul am folosit **md5**, care este o funcție criptografică care funcționează unidirecțional. Aceasta acceptă un mesaj de orice lungime și returnează un rezultat de lungime fixă care va fi utilizat în autentificarea mesajului original . În figura 42 se poate observa că în variabila *imageurl* se stochează link-ul către pagina *gravatar*, alături de e-mail-ul utilizatorului. Cifra 600 reprezintă dimensiunea pozei în pixeli. Astfel, în momentul în care se accesează pagina „Profilul meu" va fi preluată poza pe care utilizatorul trebuie să o încarce pe site-ul gravatar.

```
165   mounted() {
166     if (!this.currentUser) {
167       this.$router.push("/login");
168     } else {
169       this.imageurl =
170         "https://www.gravatar.com/avatar/" +
171         md5(this.currentUser.email) +
172         "?s=600&d=mp";
173     }
```

Figure 42 - Cod sursă pentru afișarea imaginii profilului

Am folosit această metodă pentru evitarea stocării imaginilor în cadrul aplicației. De asemenea, pentru a menține aspectul ordonat al paginii, imaginile trebuie să fie de același tip, la aceeași rezoluției, iar *Gravatar* este o unealtă care asigură acest lucru. În plus, oferă o interfață ușor de folosit pentru a face modificări imaginii sau pentru a schimba rapid fotografia.

4.3. Implementarea și asigurarea securității aplicației

- **Validarea conturilor**

Pentru a utiliza aplicația VV, este nevoie de cont, iar prin restricționarea procesului de creare a conturilor, se va asigura o securitate sporită. Astfel, nu este suficientă doar crearea unui cont, ci este necesară și validarea acestuia de către administrator. Se presupune că acesta cunoaște care sunt membrii organizației sau numărul acestora, ceea ce înseamnă ca nu va permite crearea a mai multor conturi decât este necesar.

Fără un cont activ, un utilizator poate vizualiza doar pagina principală, pagina de autentificare și cea de înregistrare, deci nu poate accesa pagina destinată candidaților, nu poate adăuga o candidatură, nu poate accesa sesiunile de vot și nu poate vota.

În plus, conturile sunt ușor de gestionat din pagina „Gestionare conturi”, iar prin funcția de căutare după nume se pot identifica ușor persoanele care nu fac parte din organizația respectivă.

- ***JSON Web Token* (JWT)**

JWT este un standard care definește un mod compact și autonom pentru transmiterea informațiilor care sunt semnate digital între părți. Această operațiune este realizată în siguranță, iar informațiile sunt transmise ca obiect JSON.

În cadrul aplicației VV, JWT este utilizat pe fiecare pagină, împiedicând accesarea acesteia în lipsa *tokenului* sau la expirarea lui (la 24 de ore de la autentificare). De asemenea, protejează și *endpoint*-urile API, deoarece *token*-ul este constant transmis între client și server pentru verificarea permisiunilor astfel încât pagina să poată fi accesată doar de utilizatorii care au rol autorizat pentru asta.

Totodată, nu este doar o modalitate de verificare a identității utilizatorului autentificat, ci și de a stoca o cantitate mică de date în mod criptat.

Spre exemplu, am folosit acest lucru în cadrul paginii de profil. Profilul utilizatorului este transmis clientului prin JWT la autentificare, într-o manieră *read-only*, iar apoi este folosit pentru afișarea numelui, *email*-ului și numelui de utilizator în formularul de editare al contului fără a mai face alte *request*-uri dacă se iese din cont.

- **Baza de date**

Următorul pas pentru a asigura securitatea aplicației a fost restricționarea accesului la baza de date din alte surse.

Am configurat folosind ***Xampp*** fișierul ***my.ini*** ca să permită accesul la baza de date doar local. Astfel, singurul mod de accesare al tabelelor este de pe mașina locală. Chiar dacă ar afla cineva IP-ul calculatorului pe care se află aplicația VV și ar încerca să se conecteze, nu va avea posibilitatea.

Un alt aspect important care ar împiedica fraudarea este limitarea accesului pe tabela cu voturile astfel încât să nu permită comenzile ***update*** sau ***delete***. Astfel, odată înregistrat votul, nu va mai putea fi modificat.

Pentru a face acest lucru, m-am conectat la *mysql* în terminal și am introdus următoarea comandă pentru a crea un utilizator nou:

```
mysql> CREATE USER 'appuser'@'localhost' IDENTIFIED BY 'apppassword1234';
```

Figure 43 - Comanda SQL pentru crearea unui utilizator

După aceea, am rulat următoarele comenzi:

```
mysql> SELECT CONCAT('GRANT SELECT, INSERT, UPDATE, DELETE SHOW VIEW ON vv.``, TABLE_NAME, `` to ''appuser''@`localhost`;')
    -> FROM INFORMATION_SCHEMA.TABLES
    -> WHERE TABLE_SCHEMA = 'vv';
+-----------------------------------------------------------------------------------------------------------+
| CONCAT('GRANT SELECT, INSERT, UPDATE, DELETE SHOW VIEW ON vv.``, TABLE_NAME, `` to ''appuser''@`localhost`;') |
+-----------------------------------------------------------------------------------------------------------+
| GRANT SELECT, INSERT, UPDATE, DELETE SHOW VIEW ON vv.`applications` to 'appuser'@`localhost`;             |
| GRANT SELECT, INSERT, UPDATE, DELETE SHOW VIEW ON vv.`questions` to 'appuser'@`localhost`;                |
| GRANT SELECT, INSERT, UPDATE, DELETE SHOW VIEW ON vv.`request_types` to 'appuser'@`localhost`;            |
| GRANT SELECT, INSERT, UPDATE, DELETE SHOW VIEW ON vv.`requests` to 'appuser'@`localhost`;                 |
| GRANT SELECT, INSERT, UPDATE, DELETE SHOW VIEW ON vv.`results` to 'appuser'@`localhost`;                  |
| GRANT SELECT, INSERT, UPDATE, DELETE SHOW VIEW ON vv.`roles` to 'appuser'@`localhost`;                    |
| GRANT SELECT, INSERT, UPDATE, DELETE SHOW VIEW ON vv.`sessions` to 'appuser'@`localhost`;                 |
| GRANT SELECT, INSERT, UPDATE, DELETE SHOW VIEW ON vv.`types` to 'appuser'@`localhost`;                    |
| GRANT SELECT, INSERT, UPDATE, DELETE SHOW VIEW ON vv.`user_roles` to 'appuser'@`localhost`;               |
| GRANT SELECT, INSERT, UPDATE, DELETE SHOW VIEW ON vv.`users` to 'appuser'@`localhost`;                    |
| GRANT SELECT, INSERT, UPDATE, DELETE SHOW VIEW ON vv.`votes` to 'appuser'@`localhost`;                    |
+-----------------------------------------------------------------------------------------------------------+
11 rows in set (0.00 sec)
```

Figure 44 - Comenzi SQL pentru a limita accesul la baza de date VV

Am introdus pe rând comenzile din tabel, însă pentru tabela *votes*, am permis **doar SELECT** și **INSERT**.

Am configurat fișierul *db.config.js* astfel încât să conțină credențialele noului utilizator, precum în figura de mai jos.

```
1  module.exports = {
2      HOST: "localhost",
3      USER: "appuser",
4      PASSWORD: "apppassword1234",
5      DB: "vv",
6      dialect: "mysql",
7      pool: {
8        max: 5,
9        min: 0,
10       acquire: 30000,
11       idle: 10000
12     }
13   };
```

Figure 45 - Fișierul db.config.js actualizat

CONCLUZII ȘI DIRECȚII VIITOARE

Lucrarea aceasta are ca și obiectiv descrierea avantajelor, tehnologiilor și utilizării aplicației dezvoltate. Subiectul ales are un impact vital asupra economiei, iar acțiunea de a vota este un drept care a fost câștigat cu greu de-a lungul timpului și pentru care au luptat foarte mulți oameni.

Așadar, aplicația VV oferă o modalitate modernă de votare care are numeroase avantaje prezentate în cele 4 capitole ale lucrării:

- **Capitolul 1**: „Importanța sistemelor de votare în societatea modernă” cuprinde informații generale referioare la vot și la procesul de votare, precum și motivele pentru care este necesară o aplicație;
- **Capitolul 2**: „Analiza cerințelor funcționale ale VV” se concentrează pe expunerea aplicației de vot VV și obiectivelor acesteia;
- **Capitolul 3**: „Valorificarea tehnologiilor utilizate în dezvoltarea soluției” prezintă cerințele principale și funcționalitățile pe care se bazează aplicația, precum și capturi de ecran cu aspectul acesteia;
- **Capitolul 4**: „Implementarea soluției informatice” se axează pe codul aplicației și faza de dezvoltare a acesteia, alături de metodele de securizare.

Poate fi considerat că aplicația contribuie la inovarea procesului de votare, rezolvând probleme existente legate de costuri, de răspândirea de virusuri, necesitatea de a avea personal care să se ocupe de organizare, chiar oferind posibilitatea mai multor oameni de a-și exprima ideile, fiind accesibilă oricând, de oriunde.

Procesul de proiectare și dezvoltare al aplicației de vot VV a presupus însușirea de noi cunoștinte din domeniul *web* prin folosirea *Vue.js*, un *framework* modern de *JavaScript*, utilizarea unui sistem de gestiune a bazelor de date relațional popular și realizarea sincronizării între partea clientului și cea a serverului, între *back-end* și *front-end*.

Întrucât tehnologiile folosite în cadrul VV sunt versatile și ușor adaptabile, aplicația modernă permite să fie oricând actualizată sau modificată după schimbările continue a cerințelor pieței, ba chiar și integrată în cadrul altor aplicații, precum *Jira, Confluence, Trello*.

De asemenea, pe viitor, pentru a spori securitatea aplicației se poate implementa un mecanism de tip *blockchain*. Lucrul care este benefic la *blockchain* este faptul că în momentul în care cineva votează, rezultatul nu este transmis doar la *server*, ci și la toți clienții. Astfel, dacă administratorul ar schimba ceva, s-ar observa ușor discrepanța.

BIBLIOGRAFIE

[1] D. Online, „Definiția votului,” [Interactiv]. Available: https://dexonline.ro/definitie/vot. [Accesat 12 Ianuarie 2021].

[2] „Wikipedia - Sistem de vot,” [Interactiv]. Available: https://ro.wikipedia.org/wiki/Sistem_de_vot. [Accesat 12 Ianuarie 2021].

[3] A. Lijphart, „Modele ale democrației. Forme de guvernare și funcționare in treizeci și șase de țări,” în *Modele ale democrației. Forme de guvernare și funcționare in treizeci și șase de țări*, Iasi, Polirom, 2000.

[4] „Sisteme de votare,” GBC, [Interactiv]. Available: https://gbc.ro/solutii/audio/sisteme_de_votare?gclid=CjwKCAiAg8OBBhA8EiwAlKw3ktC2_mccmXTch-8hw6G89MD37-c8OkByobe77phfjTjoHt1OrGVuKRoC_psQAvD_BwE. [Accesat 12 Ianuarie 2021].

[5] „"The First Vote," November 16, 1867,” *Harper's Weekly*, p. 1, 1867.

[6] „iowa.gov,” [Interactiv]. Available: https://iowaculture.gov/. [Accesat 14 Noiembrie 2020].

[7] „Cât a costat organizarea alegerilor parlamentare din 2016,” Bianca Chirilă, 26 11 2020. [Interactiv]. Available: https://stirileprotv.ro/stiri/alegeri-parlamentare-2020/cat-a-costat-organizarea-alegerilor-parlamentare-din-2016-cati-bani-au-fost-alocati-in-urma-cu-patru-ani.html. [Accesat 9 Ianuarie 2021].

[8] „Voting,” [Interactiv]. Available: https://en.wikipedia.org/wiki/Voting. [Accesat 9 Ianuarie 2021].

[9] G.-M. C. D. C. G.-C. D. Cristina Ipate-Toma, „Cetățenia activă. Participarea cetățenilor la luarea deciziilor publice și la controlul aplicării acestora.,” în *EDUCAȚIE SOCIALĂ. Manual pentru clasa a VII-a*, Costești, Argeș, Ars Libri, 2019, pp. 68-75.

[10] CONSTITUŢIA ROMÂNIEI - Art. 36 – Dreptul de vot, Andreas, 2019.

[11] „Legea nr. 33/2007 privind organizarea și desfășurarea alegerilor pentru,” 16 Noiembrie 2018. [Interactiv]. Available: https://www.europarl.europa.eu/external/appendix/ee2019/legea-nr-33-2007-privind-

organizarea-si-desfasurarea-alegerilor-pentru-parlamentul-european.pdf. [Accesat 4 Ianuarie 2021].

[12] „Abraham Lincoln,” [Interactiv]. Available: https://ro.wikiquote.org/wiki/Abraham_Lincoln. [Accesat 5 Ianuarie 2021].

[13] „Abraham Lincoln,” [Interactiv]. Available: https://en.wikipedia.org/wiki/Electoral_history_of_Abraham_Lincoln#/media/File:Abraham_Lincoln_head_on_shoulders_photo_portrait.jpg. [Accesat Noiembrie 10 2020].

[14] „Doodle,” [Interactiv]. Available: https://doodle.com/en/. [Accesat 20 Februarie 2021].

[15] „Get started with Doodle,” Ianuarie - ultima actualizare 2021. [Interactiv]. Available: https://help.doodle.com/hc/en-us/articles/360012149713-Get-started-with-Doodle. [Accesat 6 Martie 2021].

[16] „StrawPoll,” [Interactiv]. Available: https://strawpoll.com/. [Accesat 17 Februarie 2021].

[17] „Fast Poll,” [Interactiv]. Available: https://fast-poll.com/. [Accesat 3 Martie 2021].

[18] „SurveyMonkey,” [Interactiv]. Available: https://www.surveymonkey.com/. [Accesat 14 Martie 2021].

[19] „Pool Maker,” [Interactiv]. Available: https://www.poll-maker.com/. [Accesat 10 Martie 2021].

[20] „Votes PA,” [Interactiv]. Available: https://www.votespa.com/. [Accesat 10 Octombrie 2020].

[21] „Vote.gov,” [Interactiv]. Available: https://vote.gov/. [Accesat 2 November 2021].

[22] „Vote.org,” [Interactiv]. Available: https://www.vote.org/. [Accesat 16 Octombrie 2021].

[23] „Primii pași în programare. Frontend, Backend, Baze de date,” Jademy, [Interactiv]. Available: https://www.jademy.ro/blog/primii-pasi-programare-concepte-si-definitii/. [Accesat 6 Aprilie 2021].

[24] „Vue.js,” [Interactiv]. Available: https://en.wikipedia.org/wiki/Vue.js. [Accesat 6 Aprilie 2021].

[25] Monterail, State of Vue, 2017.

[26] Monterail, State of Vue.js 2021, 2021.

[27] „Best Frontend Frameworks of 2021 for Web Development," [Interactiv]. Available: https://www.simform.com/best-frontend-frameworks/. [Accesat 18 Aprilie 2021].

[28] „Best Frontend Frameworks of 2021 for Web Development," SIMFORM, 5 Ianuarie 2021. [Interactiv]. Available: https://www.simform.com/best-frontend-frameworks/. [Accesat 16 Aprilie 2021].

[29] D. Abramov, „Twitter - citat Dan Abramov," 29 Februarie 2016. [Interactiv]. Available: https://twitter.com/dan_abramov/status/704304462739939328?lang=en. [Accesat 7 Mai 2021].

[30] „What is Vuex?," [Interactiv]. Available: https://vuex.vuejs.org/#what-is-a-state-management-pattern. [Accesat 19 Aprilie 2021].

[31] I. StrongLoop, „Express," 2017. [Interactiv]. Available: https://expressjs.com/. [Accesat 3 Mai 2021].

[32] „What is CRUD?," codecademy, [Interactiv]. Available: https://www.codecademy.com/articles/what-is-crud. [Accesat 30 Mai 2021].

[33] „Express Tutorial Part 4: Routes and controllers," Mozilla, [Interactiv]. Available: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/routes. [Accesat 10 Iunie 2021].

[34] „DEXONLINE," [Interactiv]. Available: https://dexonline.ro/definitie/vot. [Accesat 12 Ianuarie 2021].

ANEXE

Anexa 1. Lista de figuri

Anexa 2. Lista de tabele