

ACADEMIA DE STUDII ECONOMICE DIN BUCUREȘTI
FACULTATEA DE CIBERNETICĂ, STATISTICĂ ȘI
INFORMATICĂ ECONOMICĂ

Lucrare de licență

Coordonator științific,
Prof. univ. dr. Ion Smeureanu

Student,
Andrei-Marius Tecșor

București
Anul 2021

ACADEMIA DE STUDII ECONOMICE DIN BUCUREȘTI
FACULTATEA DE CIBERNETICĂ, STATISTICĂ ȘI
INFORMATICĂ ECONOMICĂ

Lucrare de licență

Aplicație web pentru promovarea și
integrarea unui stil de viață sustenabil

Coordonator științific,
Prof. univ. dr. Ion Smeureanu

Student,
Andrei-Marius Tecșor

București
Anul 2021

Cuprins

Introducere	4
Capitolul 1. Problematika sustenabilității în secolul al XXI-lea	6
1.1 Prezentarea domeniilor abordare.....	6
1.1.1. Amprenta ecologică	6
1.1.2. Consumerismul	8
1.1.3. Influențarea în mediul online	8
1.2 Prezentarea ideilor de reducere a amprente ecologice	10
1.3 Comparație cu alte soluții deja existente.....	12
Capitolul 2. Analiza și proiectarea sistemului informatic propus	15
2.1 Prezentarea cerințelor sistemului ce soluționează problema sustenabilității	15
2.2 Analiza sistemului	20
2.3 Proiectarea sistemului	23
Capitolul 3. Implementarea sistemului informatic propus	27
3.1 Prezentarea tehnologiilor folosite în cadrul aplicației.....	27
3.2 Implementarea soluției	28
3.3 Prezentarea unei soluții pentru managementul stilului de viață - Collectio	37
Concluzii	42
Bibliografie	43
Anexe	46

Introducere

De-a lungul istoriei, ființa umană a căutat noi metode de a supraviețui cât mai facil, pornind de la prima sa realizare existențială și anume crearea și folosirea de unelte, până în prezent când toate nevoile sale se pot îndeplini în fața unui ecran, prin apăsarea unor simple butoane.

În evoluția umană, persistă totuși o constantă pe care, în timp, individul a depreciat-o, consumându-i resursele într-un mod haotic și poluând-o, fără să țină cont și de posibilele consecințe. Mediul natural este principalul pilon al supraviețuirii și al evoluției noastre ca specie, pilon ale cărui reacții la acțiunile egoiste ale omului încep să devină din ce în ce mai îngrijorătoare, cele mai întâlnite repercusiuni la nivel global fiind:

- Încălzirea globală, care presupune încălzirea graduală a suprafeței, a oceanelor și a atmosferei Pământului, fapt cauzat în principal de arderea combustibilului fosil, ardere care pompează dioxid de carbon, metan și alte gaze de seră în atmosferă;
- Poluarea cu plastic, care duce la omorârea a milioane de animale anual, pe lângă multe alte efecte negative;
- Poluarea aerului, principalul factor al bolilor de natură respiratorie.

Odată cu creșterea riscurilor asupra vieții, protejarea mediului a captat atenția multor specialiști din diferite domenii, devenind un subiect principal atât în discuțiile guvernelor statelor, cât și în mediul privat. Cu toate acestea, procentul persoanelor conștiente de reacțiile actuale ale mediului natural la activitățile omului este îngrijorător. Conform unui articol publicat în jurnalul *Nature*, mai mult de o treime din populația adultă de pe suprafața Terrei nu a auzit niciodată de conceptul de schimbări climatice. [\[1\]](#)

Din cauza faptului că trăim într-un moment în care internetul și rețelele de socializare fac posibilă transmiterea informației rapid, individul este copleșit de o cantitate considerabilă de conținut. Chiar dacă există un număr mare de persoane și asociații care promovează și abordează în mediul online diferite subiecte referitoare la protejarea mediului și integrarea unui stil de viață durabil, acest topic nu se bucură de aceeași popularitate ca unul bazat pe divertisment.

Pe de altă parte, înțelegerea acestor noțiuni poate deveni complexă și neatractivă, mai ales pentru persoanele care nu pot percepe și nu pot anticipa consecințele severe pe termen lung, în cazul în care nu vor face o schimbare în traiul de zi cu zi.

Ideea temei mele de licență se conturează în jurul întrebării „De ce să încep un stil de viață sustenabil?”. Pornind la această întrebare, am reușit să descopăr două răspunsuri pentru întrebarea „De ce să nu o fac?”.

Primul răspuns susține faptul că este dificil pentru o persoană să îndeplinească anumite sarcini fără a se putea bucura vizibil de roadele muncii sale într-un viitor apropiat. O schimbare de asemenea anvergură necesită timp, susținere și implicare din partea celorlalți, întrucât doar o comunitate care înțelege importanța tranziției poate reuși.

Al doilea răspuns are la bază ierarhia lui Abraham Maslow, cunoscută și sub numele de „Piramida lui Maslow” ([Anexa 1](#)), fiind vorba în special de nivelurile patru și cinci, respectiv nevoia de stimă și nevoia de auto-actualizare. Pe lângă respectul de sine, individul are nevoie de apreciere și admirație din partea celorlalți pentru a continua procesul de integrare al unui stil de viață durabil. După îndeplinirea acestei nevoi, persoana poate aspira spre ultimul nivel al piramidei ce presupune „fructificare la maximum a capacităților proprii, pentru a deveni din ce în ce mai bun”. [\[2\]](#)

Înțelegând aceste aspecte, motivația de a dezvolta aplicația izvorăște din dorința de a crea un mediu propice în care persoanele să facă tranziția către un stil de viață sustenabil într-o comunitate unită printr-un scop comun, fiecare membru având posibilitatea de a juca un rol important în evoluția aproapelui său.

Capitolul 1. Problematica sustenabilității în secolul al XXI-lea

1.1 Prezentarea domeniilor abordare

1.1.1. Amprenta ecologică

În momentul de față, problema principală la nivel global, din punct de vedere al mediului înconjurător, este reducerea amprentei ecologice pe cât de mult posibil. Nucleul problemei este în speță faptul că omenirea consumă mai multe resurse naturale decât poate regenera planeta.

Amprenta ecologică (în limba engleză *Ecological Footprint*) este o metrică dezvoltată la începutul anilor 1990 de către Mathis Wackernagel în cadrul lucrării sale de doctorat la Universitatea din Columbia Britanică, sub îndrumarea profesorului William Rees. [3] Aceasta măsoară cererea omenirii de capital natural în raport cu oferta naturii, concret, cantitatea resurselor naturale necesare pentru a susține economia și activitatea oamenilor prin intermediul unui sistem complex numit contabilitate ecologică.

Contabilitatea ecologică este un subset al contabilității propriu-zise, obiectivul principal fiind integrarea informațiilor referitoare la mediu în cadrul economic. În calculul amprentei ecologice, aceasta ajută la compararea ariei biologice productive pe care societatea o folosește pentru consum cu biocapacitatea zonei respective, biocapacitatea însemnând puterea de regenerare a resurselor necesare omului dintr-o zonă anume. [4][5]

Biocapacitatea reprezintă diferența dintre cantitatea de teren biologic productiv (inclusiv suprafața maritimă) pentru a furniza resursele necesare consumului populației și, totodată, pentru a absorbi deșeurile generate de acesta. Pentru a face biocapacitatea comparabilă în timp și spațiu, zonele sunt ajustate proporțional cu productivitatea lor biologică, aceste suprafețe fiind exprimate în „hectare globale” (gha). [7][8]

Din perspectiva cererii, amprenta ecologică colectează toate zonele productive pentru care concurează o populație, o persoană sau un produs. În mod uzual, aceste zone sunt terenuri arabile, zone piscicole, suprafețe forestiere ș.a.m.d. Sunt evaluate activele ecologice necesare realizării unei activități sau a unui bun pentru a produce resursele naturale pe care le consumă și pentru a absorbi deșeurile rezultate în urma procesului. De menționat este faptul că resursele consumate fac inclusiv referire la: produsele alimentare de origine animală sau vegetală, produse din lemn, dar și spațiul necesar pentru infrastructura urbană.

În ceea ce privește oferta, accentul cade pe biocapacitate, ilustrând astfel productivitatea activelor ecologice ale zonelor menționate anterior în cadrul cererii. Totodată, neexploatarea acestor zone poate juca un rol esențial în absorbirea deșeurilor generate în urma activităților omului, cum ar fi emisiile de carbon generate de arderea combustibililor fosili. [4][6]

Din punct de vedere al productivității ecosistemelor, fiecare țară este diferită; acest aspect este evidențiat în conturile ecologice ale țării respective, formula care ilustrează impactul ecologic al unei țări fiind următoarea:

$$EF_c = EF_p + (EF_i - EF_e) \text{ (Anexa 2) , unde:}$$

- EF_c este amprenta ecologică de consum și ilustrează consumul biocapacității de către locuitorii unei țări; aceasta este cea mai susceptibilă la îmbunătățire prin schimbarea modalităților de consum ale individului.
- EF_p este amprenta ecologică de producție; aceasta indică consumul biocapacității rezultat din procese de producție într-o zonă geografică specifică, de pildă: o regiune sau o țară; măsura aceasta oglindește produsul intern brut al țării respective.
- $(EF_i - EF_e)$ reprezintă amprenta ecologică netă de comerț, fiind compusă din diferența dintre amprenta ecologică a importului (EF_i) și amprenta ecologică a exportului (EF_e); aceasta consemnă utilizarea biocapacității la nivel internațional. Contrar unei intuiții economice, o țară a cărei amprentă a importului este mai mare decât cea a exportului este dependentă de resursele regenerabile și servicii ecologice din afara graniței acesteia.

Pe baza analizei referitoare la impactul ecologic, țările lumii sunt împărțite în două categorii: creditori ecologici, țările ale căror amprentă este mai mică decât biocapacitatea lor, și debitori ecologici, țările care se află în stare contrară față de prima categorie. [\[6\]\[7\]\[9\]](#)

Amprenta ecologică este principalul factor în evaluarea sustenabilității unui cadru, atât la nivel național sau global, cât și la nivel privat sau personal, oferind posibilitatea individului să integreze un stil de viață durabil în toate cadrele în care acesta activează.

Actualmente, peste 85% dintre țările lumii se află în deficit ecologic (conform Global Footprint Network), acest deficit ecologic mondial purtând numele de depășire ecologică globală (în engleză *global ecological overshoot*). [\[7\]\[10\]](#)

Conform ultimelor date oferite de Global Footprint Network, România înregistra în anul 2017 un deficit de 0,3 gha, cu o biocapacitate de 3,1 gha și o amprentă ecologică de 3,4 gha, cele din urmă fiind raportate per individ. Făcând comparație cu alte țări ale lumii, precum Statele Unite ale Americii, China, Germania etc., țara noastră deține un scor cu un real potențial de a înclina balanța spre a-și plăti datoria, aceasta reușind în anul 2014 să atingă echilibrul. ([Anexa 3](#), [Anexa 4](#))

Totuși, nu este de îndeajuns să ne raportăm la o singură țară, întrucât problema este una de natură mondială, astfel o mare parte dintre guvernele lumii începe să devină conștientă de pericolul și gravitatea situației unui deficit ecologic global prelungit.

1.1.2. Consumerismul

Cum am afirmat și anterior, componenta cea mai pasibilă de optimizare este amprenta ecologică de consum care pune accent pe individ și pe modalitatea de consum a acestuia. O problemă cu care societatea actuală se confruntă începând de la revoluția industrială până în prezent este consumerismul.

Conceptul de „consumerism” a primit de-a lungul istoriei mai multe definiții, multe dintre ele aflându-se chiar în antiteză. Pentru subiectul abordat în această lucrare, vom asocia termenul cu o serie de conotații negative care fac referire la nivelurile ridicate de consum ale individului și la achizițiile egoiste în scopul colectării superficiale de produse, cunoscută și sub numele de materialism. [\[10\]](#)

În viziunea materialistă se conturează ideea că bunurile colecționate îi pot aduce individului fericirea și respectul celorlalți în societate. Deși această atitudine presupune ilustrarea bunăstării unei persoane, nu oferă și „fericirea” mediului înconjurător, astfel aceste principii ale materialismului se află în antiteză cu ideologia ecologismului, ideologie ce pune accent pe protejarea mediului natural prin diferite mijloace posibile, de pildă: reducerea consumului. [\[11\]\[12\]](#)

Datorită evoluției societății și a tehnologiei, consumul de bunuri, atât pentru supraviețuire, cât și pentru satisfacerea unor dorințe sau a unor capricii, a devenit facil pentru majoritatea persoanelor, individul putând achiziționa orice, la orice oră din zi și din noapte, cu un minim de efort, singurul impediment fiind banii. Actualmente, unul dintre scopurile pe care dorește să le atingă evoluția umană este de a ajuta omul să supraviețuiască în confort, multe invenții de genul devenind indispensabile pentru trai, de pildă: supermarketurile, mașinile, electricitatea ș.a.m.d.

Pe de altă parte, supraviețuirea nu se mai regăsește în obiectivele unui individ, întrucât este considerat ca fiind un aspect subînțeles, așadar aceste obiective ajungând să se rezume la ultimele trei nivele ale Piramidei lui Maslow. Din păcate, nevoile de apartenență și de stimă sunt hrănite de cele mai multe ori prin materialism, această satisfacere aparentă fiind generată de modalitățile inventive de promovarea a bunurilor și serviciilor.

1.1.3. Influențarea în mediul online

Cum o persoană obișnuită petrece în medie 145 de minute pe zi pe rețele de socializare [\[13\]](#), acestea sunt terenul ideal de promovare al diferitelor campanii de marketing care au ca scop

principal creșterea vânzărilor, ceea ce presupune implicit și creșterea consumului. Cea mai mare influență asupra unei persoane o au în prezent creatorii de conținut pe care persoana respectivă îi urmărește.

Un creator de conținut, numit adesea „influencer”, este de fapt o persoană fizică, uneori chiar o celebritate, care și-a dezvoltat faima și notabilitatea prin intermediul rețelelor de socializare. Celebritățile din mediul online funcționează adesea ca guru al stilului de viață pe care îl promovează, astfel joacă un rol important în crearea și susținerea diferitelor tendințe de actualitate din diferite domenii, precum: sport, frumusețe, tehnologie, politică ș.a.m.d. [14] Prin urmare, influențatorii sunt cineva sau ceva cu puterea de a modifica obiceiurile de consum sau acțiunile cuantificabile ale indivizilor prin încărcarea unei forme de conținut, adesea sponsorizat, pe diferite rețele de socializare faimoase.

Denumirea de „influencer” provine de la modalitatea de marketing folosită și anume „marketingul prin influență”. Acest tip de marketing implică recomandări și plasare de produse de la diferite persoane sau organizații influente care pretind că au un nivel de cunoștințe în domeniul promovat. Caracteristica principală a formei de marketing prezentată se concentrează pe convingere și conformitate socială, astfel accentul nu este pus pe argumentarea unui punct de vedere în raport cu un produs sau un serviciu, ci pe interacțiunile dintre creator și comunitatea sa. [14][15][16]

Marketingul prin influență își are rădăcinile în *Teoria Comparăției Sociale*, propusă de Leon Festinger în anul 1954. Ipoteza psihologului pornește de la convingerea cum că ar exista un impuls care stârnește dorința individului de a obține autoevaluări concrete. Teoria explică modul în care o persoană își evaluează propriile abilități și opinii, comparându-se cu ceilalți pentru a reduce incertitudinea în anumite domenii și pentru a învăța cum să se definească pe sine. În urma cercetărilor amănunțite izvorâte din această teorie, comparația socială a primit o conotație pozitivă, fiind considerată o modalitate de auto-îmbunătățire. [18]

Creatorii de conținut renumiți servesc drept instrument de comparație, astfel consumatorul simplu compară direct imperfecțiunile vieții sale cu viața „perfectă” a celebrității urmărite, ridicând astfel statutul acestora deasupra lor. Ca atare, produsele promovate pot servi ca o scurtătură către un stil de viață complet. Prin promovarea unei produs prin intermediul unui *influencer*, compania respectivă se poate folosi de nesiguranțele consumatorilor în beneficiul propriu. Din păcate, această formă de marketing de influență este des întâlnită în zilele noastre și, pe lângă faptul că este nocivă din punct de vedere psihic pentru consumator, este nocivă în majoritatea cazurilor și pentru planetă prin promovarea materialismului și a conceptelor acestei ideologii. [17][19]

1.2 Prezentarea ideilor de reducere a amprente ecologice

Pe baza conceptelor explicate anterior, se poate contura un model izolat raportat la individ, un flux de acțiuni care se întâmplă în mod recurent și al cărui rezultat este agravarea stării mediului înconjurător pe termen lung, întrucât această înrăutățire a situației nu este vizibilă imediat.

Etapile principale ale modelului sunt următoarele:

- 1) Campaniile de marketing prin influență derulate în mediul digital de diferite companii pentru a-și spori vânzările motivează individul spre consum;
- 2) Modalitatea de promovare creează la nivelul persoanei dorința de apartenență, în special la nivelul unei comunități, și accentuează viziunea prin care se enunță faptul că un produs folosit de o persoană influentă sau o acțiune înfăptuită tot de aceasta te poate aduce mai aproape de traiul „ideal” al creatorului de conținut cu pricina;
- 3) Individul cumpără produsul respectiv sau realizează activitatea respectivă.

Ultima etapă are o serie de repercusiuni negative raportat la mediul natural, cel mai semnificativ fiind consumul în masă care este dăunător din mai mult privințe, pornind de la resursele naturale consumate, până la emisiile și reziduurile rezultate în urma prelucrării materialelor. Totodată, trebuie luat în calcul și ce se întâmplă cu bunurile consumate în urma folosirii, numărul în expansiune al gropilor de gunoi devenind o problemă din ce în ce mai des întâlnită la nivel global.

Modelul are la bază consumul, dar problematica în sine se rezumă la modalitatea de consum a unei persoane, criteriile principale de ghidare fiind calitatea și cantitatea. Din păcate, la nivelul rețelelor de socializare faimoase, persoanele care încearcă să promoveze un stil de viață sustenabil, cu un consum echilibrat, nu se bucură de același impact pe care îl au cei ce creează conținut de divertisment. Acest lucru se întâmplă din diferite motive, câteva exemple semnificative fiind: informațiile complexe despre sustenabilitate, dorința oamenilor de a evada din problemele realității și a se refugia în mediul online, promovarea agresivă a protecției mediului care pune accent pe dezastrele cauzate de om în natură; deși exprimă realitatea și este bine să fim conștienți de faptele noastre, această abordare are un impact de moment bazat pe sentimente negative, individul ajungând în timp să devină imun.

În același timp, un stil de viață durabil este greu de contorizat fără un efort substanțial depus. Totodată, fiind foarte multă informație despre durabilitatea în timp a unui trai sustenabil și o sumedenie de modalități de înrolare în această inițiativă, individul este copleșit, astfel devenind inapt să răspundă la următoarele întrebări: „Cum să încep?” , „De unde să încep?”, „De ce să încep?”.

Rezolvarea inconvenientelor menționate anterior se raportează la consumatorul de rând, țelul principal fiind crearea unui mediu protejat în care persoanele interesate să experimenteze o tranziție facilă către un stil de viață sustenabil alături de o comunitate sănătoasă. Un element cheie este posibilitatea individului de a-și observa impactul la nivelul comunității, rezolvând astfel problemele enunțate în raport cu Ierarhia lui Maslow și oferindu-i utilizatorului posibilitatea de a deveni „influencer” fără a depăși strictul necesar de implicare.

Prin urmare, planul de a fixa carențele ecologice se concretizează într-o platformă online dedicată integrării elementelor de sustenabilitate în viața de zi cu zi, prin intermediul unor provocări realizate în așa fel încât să-i creeze utilizatorului noi obiceiuri sănătoase care să fie recunoscute la nivel de colectiv.

Conceptul de bază folosit în crearea provocărilor este „obiceiul” (în engleză *habit*), scopul principal fiind de a ajuta individul să capete noi comportamente rutiniere, astfel încât acestea să fie efectuate în mod subconștient. [\[20\]](#)

Conform unor studii realizate în cadrul University College London (UCL) din Regatul Unit, obiceiurile sunt acțiuni declanșate automat ca răspuns în urma stimulării unor indici contextualii, indici care au fost asociați de către creier cu performanța lor. Pentru a căpăta un obicei nou, o persoană trebuie să repete procesul pe o perioadă cuprinsă între 18 și 254 de zile. În medie, sunt necesare 66 de zile, echivalentul a 9-10 săptămâni, pentru un obicei de a deveni un comportament recurent și automat. [\[21\]\[22\]](#)

Unele obiceiuri au nevoie de a o perioadă mai lungă pentru a fi întipărite în subconștient, altele mai puțin. Durata este afectată de mai multe variabile, de pildă: deschiderea individului la rutină, considerentul de eficiență al acțiunilor determinat de minte. Creierul este ghidat de aceste obiceiuri datorită performanței lor, acțiunile automate eliberând resurse mentale pentru alte sarcini. [\[21\]\[22\]](#)

Procesul de eliminare sau schimbare al unui obicei este complex, din cauza faptului că, în momentul în care efectuăm o acțiune automatizată, un comportament plăcut minții, creierul eliberează dopamină, această substanță devenind recompensa care întărește obiceiul și dorința de a-l repeta. [\[21\]\[22\]](#)

Pentru modificările majore în rutina individului, în cazul nostru integrarea elementelor de trai sustenabil, este necesar un plan bine structurat, care are la bază șablonul din [Anexa 5](#), și acceptat în mod conștient de utilizator; acest plan va fi integrat în structura fiecărei provocări și va ajuta individul să dobândească obiceiuri specifice unui stil de viață durabil.

Prin urmare, soluția propusă de mine este o aplicație web, mai exact, o rețea de socializare prin intermediul căreia să fie promovat un stil de viață sustenabil într-o atmosferă relaxată, dar conștientă de impactul omului în cadrul natural, unde fiecare individ are posibilitatea de a deveni un pilon de influență pe baza activității și rezultatelor sale în cadrul provocărilor de integrare a elementelor durabile în viața de zi cu zi.

1.3 Comparație cu alte soluții deja existente

La nivel global, problema punerii în pericol a mediului înconjurător și consecințele acestea devin din ce în ce mai alarmante și substanțiale, astfel o sumedenie de organizații din diferite țări ale lumii vin cu o serie de soluții inventive, inclusiv pe teritoriul României.

Odată cu interesul în ascensiune pentru protejarea cadrului natural, am încercat mai multe aplicații software dedicate acestei cauze, cele care mi-au captat atenția și au servit ca punct de inspirație fiind următoarele: [*Planet Patrol*](#), [*TrashOut*](#) și [*We don't have time*](#).

Planet Patrol

Planet Patrol, în traducere liberă „Patrula Planetei”, este o organizație non-profit înființată în 2016 de către Lizzie Carr în Regatul Unit, după ce aceasta a observat deșeurile din plastic care acaparează căile navigabile.

Proiectul a înaintat cu pași mari ajungând în peste 80 de țări, astfel obiectivele principale sunt protejarea planetei și a animalelor sălbatice. Îndeplinirii acestor misiuni se concentrează pe înștiințarea omenirii despre amenințările deșeurilor menajere aruncate iresponsabil, dar și despre cultura negativă a produselor de unică folosință. [\[23\]](#)

Aplicația software dezvoltată de echipa *Planet Patrol* oferă o hartă interactivă, folosind soluția celor de la *MapBox*, prin care utilizatorul prezintă deșeurile pe care le-a strâns dintr-o zonă anume sub formă de fotografii. Înainte de a trimite pozele spre verificare, se asociază fiecare element din poze cu tipul acestuia, de exemplu: pungă de plastic, doze de băutură ș.a.m.d., oferindu-se totodată posibilitatea de a specifica marca, alături de codul de bare al produsului aruncat. După aprobarea postării de către o echipă specializată, aprobare ce durează maximum 48 de ore, utilizatorul poate distribui pe rețeaua de socializare *Twitter* realizarea sa.

Din punctul meu de vedere, soluția este foarte bună și face apel la totalitatea persoanelor dintr-o comunitate socială. Pe de altă parte, dezavantajele sunt următoarele: lipsa posibilității de a lega noi conexiuni între utilizatori și a putea vedea influența reală pe care o ai asupra altora. De asemenea, se rezumă doar la colectat, punând accentul pe rezolvarea consecințelor acțiunilor dăunătoare ale omului, nu și pe schimbarea acțiunilor în sine.

TrashOut

TrashOut este un proiect de mediu care își propune să cartografieze toate depozitele ilegale de deșeuri din întreaga lume și să ajute cetățenii să recicleze mai mult, oferindu-i utilizatorului posibilitatea de a avea un impact real asupra mediului prin intermediul unui smartphone. [\[24\]](#)

Aplicația a fost introdusă și în România de către organizația civică „Let’s do it, Romania”, organizație al cărui scop este mobilizarea cetățenilor țării în participarea la evenimente dedicate curățării zonelor afectate de depozitarea iresponsabilă a deșeurilor.

Privind funcționalitățile, aplicația se aseamănă cu Planet Patrol, diferența majoră fiind faptul că problemele de nivel ecologic nu se mai adresează doar comunității doritoare, ci și cadrelor legale împuternicite de a rezolva aceste aspecte, cum ar fi: primăria răspunzătoare de zona afectată și Garda Națională de Mediu. În plus, oferă o secțiune dedicată de știri ecologice, informații despre reciclare și locațiile unde poți face acest lucru, precum și o multitudine de statistici interesante despre activitatea cetățenilor din cadrul unei țări sau activitatea tuturor utilizatorilor la nivel global.

Dezavantajele menționate la aplicația anterioară se mențin, fiind ușor îmbunătățită partea referitoare la activitatea de socializare, întrucât utilizatorul se poate alătura unei asociații verificate, poate participa la evenimente de colectare a deșeurilor sau poate organiza astfel de evenimente atât pe plan local, cât și pentru zone și spații verzi care au nevoie de curățare.

We don’t have time

We don’t have time (în traducere liberă „Nu avem timp”) este o mișcare și un startup care valorifică puterea rețelelor de socializare pentru a responsabiliza liderii și companiile în contextul schimbărilor climatice. De remarcat este faptul că fondatorii sunt oameni care activează în domenii diferite, niciunul expert în cauza pe care o promovează, uniți prin conștientizarea înrăutățirii sănătății Pământului, premisa de la care a pornit tot proiectul fiind: „Nu voi mai fi pasiv și doar voi privi. Dacă nu fac nimic, cine o va face pentru mine?”. [\[25\]](#)

Soluția oferită de echipa de dezvoltatori este în speță o rețea de socializare asemănătoare platformelor *Facebook* și *Twitter*, din punct de vedere al funcționalităților; concret, oferă următoarele specificații: creare, apreciere, distribuire de postări, comentare în cadrul unei postări, posibilitatea de a urmări alte persoane sau organizații și de a fi urmărit la rândul tău de alte entități.

Avantajul principal este mediul concentrat pe un singur subiect major, și anume, criza climatică, astfel toate discuțiile și postările vin cu probleme sau rezolvări, țelul principal fiind soluționare acestei crize pe cât de mult posibil.

Față de celelalte două aplicații, aceasta nu se raportează direct la individ, lipsind totodată și componentele interactive prin care orice persoană poate face o serie de activități, ne semnificative la prima impresie, dar care pe viitor vor avea un impact pozitiv asupra populației și în special asupra Terrei.

Capitolul 2. Analiza și proiectarea sistemului informatic propus

2.1 Prezentarea cerințelor sistemului ce soluționează problema sustenabilității

Sistemul ce va rezolva problemele prezentate anterior va fi o aplicație web, concret, o rețea de socializare care urmărește evoluția utilizatorului, ajutându-l să dezvolte un stil de viață sustenabil și totodată, prin puterea exemplului, să motiveze și să influențeze alți utilizatori să ia inițiativă, făcând astfel un bine planetei prin gesturi mici, dar semnificative.

Prin încercarea unor provocări specifice, bazate pe durata de execuție, utilizatorul va dezvolta obiceiuri și tendințe noi de a duce un trai durabil, realizările sale având capacitatea de a motiva la rândul lor alte persoane să se alăture mișcării.

Prima condiție de utilizare a aplicației este crearea unui cont. După autentificarea în aplicație, utilizatorul este întâmpinat de un flux de știri (*News Feed*), pe baza căruia va putea vedea cele mai recente activități ale persoanelor pe care le urmărește, precum și cele mai populare provocări.

Activitatea principală a utilizatorului este înrolarea într-o provocare, o provocare fiind împărțită în minim una sau mai multe etape. Din punct de vedere al cerințelor, etapele unei provocări devin din ce în ce mai complexe pe parcurs ce utilizatorul înaintează în cadrul acesteia. Construcția provocărilor și etapelor aferente urmărește modelul propus de cercetătorii Benjamin Gardner, Phillippa Lally și Jane Wardle [22], model ce poate fi regăsit și în [Anexa 5](#). Elementul cheie este dorința de a lărgi perspectiva persoanei prin expunerea concretă a scopului final și a beneficiilor acestuia, urmând ulterior prezentarea activităților sub o formă simplistă, dar care necesită consistență și pasiune din partea viitorului practicant.

Sistemul de verificare al activității unui utilizator în contextul îndeplinirii cerințelor unei provocări este bazat pe încredere, utilizatorul fiind nevoit să confirme continuarea provocării săptămânal. Raportat la o provocare în desfășurare, utilizatorul se poate afla într-una dintre următoarele perioade:

- **Perioada de încredere**, care începe de la momentul înrolării în provocare sau momentul ultimei confirmări și durează patru zile, inclusiv ziua de start; în această perioadă, utilizatorul nu este nevoit să facă nicio acțiune în privința provocării respective.
- **Perioada de verificare**, care are loc imediat după cea de încredere, aceasta întinzându-se pe o durată de trei zile; aici, utilizatorul trebuie să confirme că nu a renunțat la provocare, astfel va avea loc unul dintre următoarele scenarii:

- ❖ Scenariul 1: Dacă utilizatorul confirmă, procesul se va repeta, cu mențiunea că data confirmării va fi calculată ca fiind ultima dată a confirmării la care se adaugă o săptămână.
- ❖ Scenariul 2: Dacă utilizatorul nu confirmă în timp util, va fi descalificat din provocare, pierzând tot progresul până la momentul respectiv, singura modalitate de a-l recăpăta fiind reînceperea provocării.

La repornirea aplicației, serverul va verifica automat dacă utilizatorul a depășit „perioada de verificare” pentru fiecare provocare aflată în proces de execuție, iar – în caz afirmativ – îi va anula provocarea. Totodată, utilizatorul poate renunța la o provocare oricând dorește, independent de perioada în care se află, consecințele fiind aceleași.

Fiecare etapă dintr-o provocare are o durată de timp necesară realizării, dar și o modalitate de premiere în momentul completării. Atunci când utilizatorul își confirmă activitatea, serverul va verifica dacă este îndeplinită condiția necesară completării. Dacă este îndeplinită, utilizatorul este recompensat cu medalia aferentă etapei și este trecut automat în etapa următoare. În cazul în care a finalizat toate etapele unei provocări, poate continua, țelul în această situație fiind doborârea recordului personal. Recordurile personale, cât și medaliile câștigate, vor fi afișate pe pagina de profil a utilizatorului, cu specificația că un utilizator va avea o singură medalie asociată unei provocări și anume ultima deblocată.

Datorită faptului că aplicația este gândită ca fiind o rețea de socializare, va fi implementat un sistem de urmăritori și anume: un utilizator poate urmări mai mulți utilizatori, iar acesta poate fi urmărit la rândul său de mai mulți utilizatori. Totodată, utilizatorul își poate schimba datele afișate, cât și cele personale.

Soluția software urmărește dezvoltarea unei comunități conștiente de beneficiile unui stil de viață durabil, impactul fiecărui utilizator jucând un rol semnificativ în atingerea scopului principal. În momentul în care un utilizator se alătură unei provocări datorită altui utilizator, se creează o relație de influență. Prin intermediul acestor relații, utilizatorul va putea vedea concret impactul asupra comunității, fiind informat despre următoarele elemente: numărul de provocări începute datorită activității acestuia și numărul de persoane influențate. Procesul de influențare poate lua naștere fie la nivelul componentei de News Feed, fie pe pagina personală a altui utilizator.

În primă instanță, gestiunea și organizarea provocărilor și a etapelor corespunzătoare acestora va fi făcută de către administratorul aplicației, acesta fiind singurul care poate șterge, edita și adăuga provocări, respectiv etape. Pentru a efectua operațiile de tip C.R.U.D. (Create Read Update Delete) menționate anterior, administratorul realizează următoarele activități:

- Urmărirea rezultatelor utilizatorilor în cadrul provocărilor;
- Urmărirea popularității provocărilor și a etapelor;
- Documentarea despre noi tipuri de provocări și etape.

Pentru a face trecerea de la documentarea cerințelor realizată anterior la stabilirea detaliilor de proiectare, am folosit diagrama cazurilor de utilizare (Figura 1), specifică limbajului de modelare UML (Unified Modeling Language), identificând astfel trei actori și zece cazuri de utilizare, alături de relațiile regăsite între aceștia.

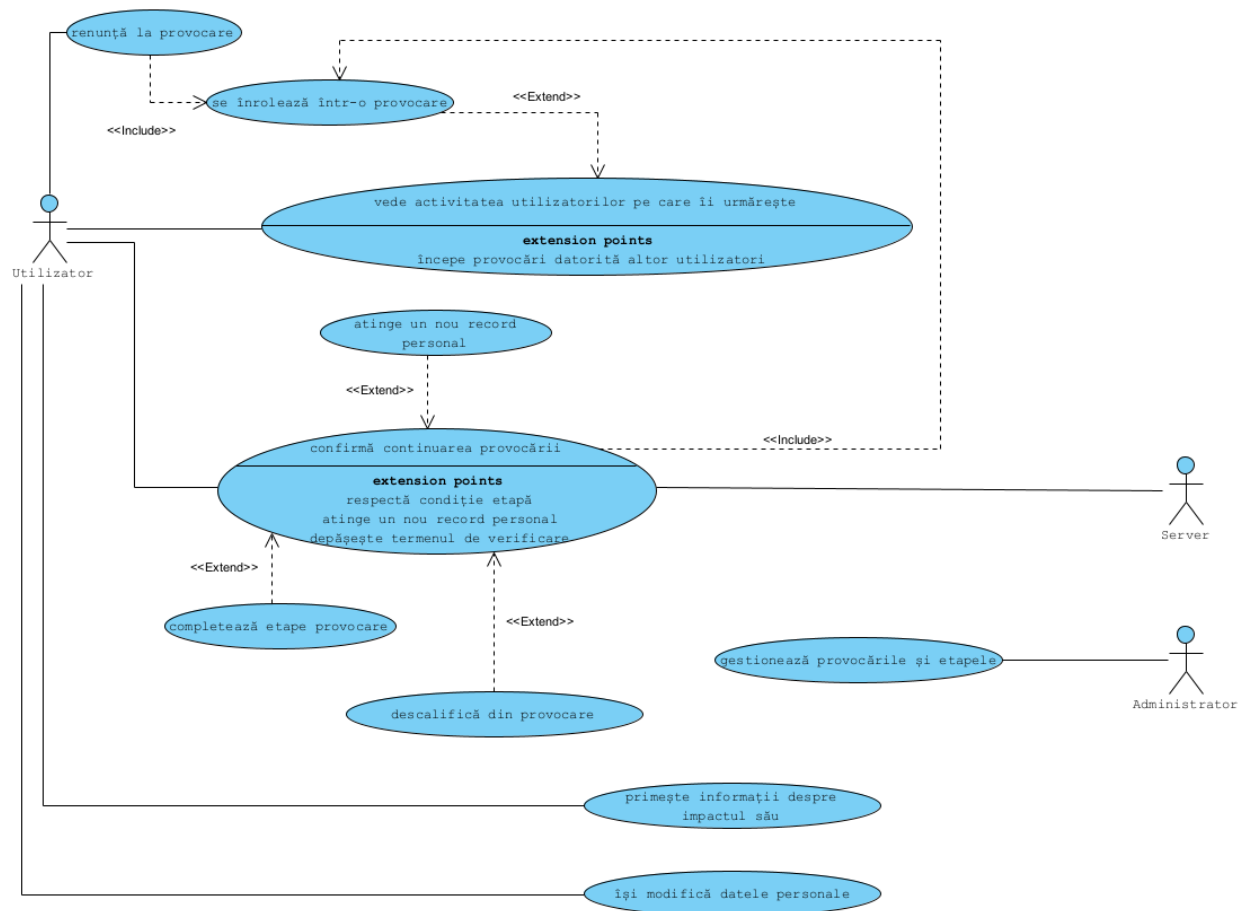


Figura 1. Diagrama cazurilor de utilizare

Actorii reprezintă entitățile care interacționează cu sistemul, astfel actorii identificați pentru sistemul prezentat fiind următorii: utilizatorul aplicației, serverul de back-end și administratorul soluției.

Cazurile de utilizare arată ce funcționalități are sistemul în raport cu interacțiunea actorilor, fiecare caz de utilizare în parte având asociat un actor principal. Deși în cadrul diagramei de la Figura 1 există mai multe cazuri de utilizare, în continuare, sunt prezentate descrierile contextuale

pentru cele mai complexe funcționalități, și anume: „se înrolează într-o provocare” (Figura 2), „confirmă continuarea provocării” (Figura 3). Totodată, prin prisma faptului că sunt acțiuni principale, acestea sunt conectate și cu celelalte cazuri de utilizare prin diferite relații.

Element al cazului de utilizare	Descriere
Cod	JOINED01
Stare	Schiță
Scop	Dezvoltarea unui stil de viață durabilă
Nume	Se înrolează într-o provocare
Actor principal	Utilizator
Descriere	Presupune înrolarea utilizatorului într-una sau mai multe provocări
Precondiții	Utilizatorul trebuie să fie autentificat pe platformă și să nu fie deja înrolat în provocare
Postcondiții	Înrolarea a fost făcută cu succes, utilizatorul poate îndeplini cerințele acesteia
Declanșator	Utilizatorul vede provocarea fie singur, fie datorită altui utilizator și se înrolează
Flux de bază	<ol style="list-style-type: none"> 1. Utilizatorul vede provocarea la alt utilizator prin intermediul fluxului de știri. [Flux alternativ A: Utilizatorul descoperă singur provocarea pe pagina de provocări] 2. Utilizatorul începe provocarea datorită altui utilizator. [Caz de utilizare extins de la INFLUENCED01 „vede activitatea utilizatorilor pe care îi urmărește ”] [Flux alternativ B: este deja înrolat în provocare] 3. Utilizatorul primește detaliile primei etape. 4. Utilizatorul începe să îndeplinească cerințele provocării.
Fluxuri alternative	<p>A:</p> <ol style="list-style-type: none"> 1. Se înrolează în provocare. 2. Se trece la punctul 3. <p>B:</p> <ol style="list-style-type: none"> 1. Se trece la punctul 4.
Relații	<p>Relație de extindere:</p> <ul style="list-style-type: none"> • Caz care extinde: INFLUENCED01 „vede activitatea utilizatorilor pe care îi urmărește”. <p>Relații de includere:</p> <ul style="list-style-type: none"> • Caz inclus: JOINED02 „confirmă continuarea provocării”, JOINED03 „renunță la provocare”
Frecvența utilizării	Frecvent
Reguli ale afacerii	Utilizatorul trebuie să confirme săptămânal, în „perioada de verificare”, că acesta îndeplinește cerințele etapei corespunzătoare din provocare

Figura 2 Descrierea textuală a cazului de utilizare „se înrolează într-o provocare”

Element al cazului de utilizare	Descriere
Cod	JOINED02
Stare	Schiță
Scop	Verificarea activității utilizatorului în cadrul unei provocări
Nume	Confirmă continuarea provocării
Actor principal	Utilizator
Descriere	Modalitatea prin care sistemul verifică dacă utilizatorul este eligibil pentru a trece la următoarea săptămână
Precondiții	Utilizatorul trebuie să fie înrolat într-o provocare
Postcondiții	Utilizatorul avansează în provocare cu o săptămână
Declanșator	Utilizatorul se află în „perioada de verificare”
Flux de bază	<ol style="list-style-type: none"> 1. Utilizatorul încearcă să confirme o provocare. 2. Serverul verifică dacă utilizatorul se află în „perioada de verificare”. 3. Serverul îl înștiințează pe utilizator că este în perioada corespunzătoare. [Flux alternativ A: Utilizatorul nu este încă în perioada corespunzătoare.] [Flux alternativ B: Utilizatorul a depășit termenul de verificare] 4. Serverul avansează utilizatorul în săptămână următoare. 5. Serverul verifică dacă utilizatorul și-a depășit recordul personal. 6. Utilizatorul își doboară recordul personal. [Punct extindere: JOINED05 „atinge un nou record personal”] [Flux alternativ C: Utilizatorul nu și-a depășit recordul personal] 7. Serverul verifică dacă utilizatorul respectă condiția de finalizare a etapei. 8. Utilizatorul finalizează etapa curentă și trece la următoarea. [Punct extindere: REACHED01 „completează etape provocare”] [Flux alternativ D: Utilizatorul nu respectă condiția de avansare]
Fluxuri alternative	<p>A:</p> <ol style="list-style-type: none"> 1. Serverul îl înștiințează pe utilizator că se află în „perioada de încredere”. 2. Se încheie scenariul. <p>B:</p> <ol style="list-style-type: none"> 1. Serverul îl descalifică din provocare. [Punct extindere: JOINED04 „descalifică din provocare”] 2. Serverul îl înștiințează că a depășit „perioada de verificare”. 3. Se încheie scenariul. <p>C:</p> <ol style="list-style-type: none"> 1. Se trece la punctul 7. <p>D:</p> <ol style="list-style-type: none"> 1. Se încheie scenariul.
Relații	<p>Relații de extindere:</p> <ul style="list-style-type: none"> • Caz de bază: JOINED04 „descalifică din provocare”, JOINED05 „atinge un nou record personal”, REACHED01 „completează etape provocare” <p>Relații de includere:</p> <ul style="list-style-type: none"> • Caz de bază: JOINED01 „Se înrolează într-o provocare”
Frecvența utilizării	Frecvent
Reguli ale afacerii	Verificarea fiecărei provocări se face săptămânal de la data înrolării.

Figura 3 Descrierea textuală a cazului de utilizare „confirmă continuarea provocării”

2.2 Analiza sistemului

Primul pas al analizei presupune crearea diagramei de clase, specifică limbajului UML, pentru a observa care sunt entitățile necesare soluției și pentru a înțelege relațiile dintre acestea.

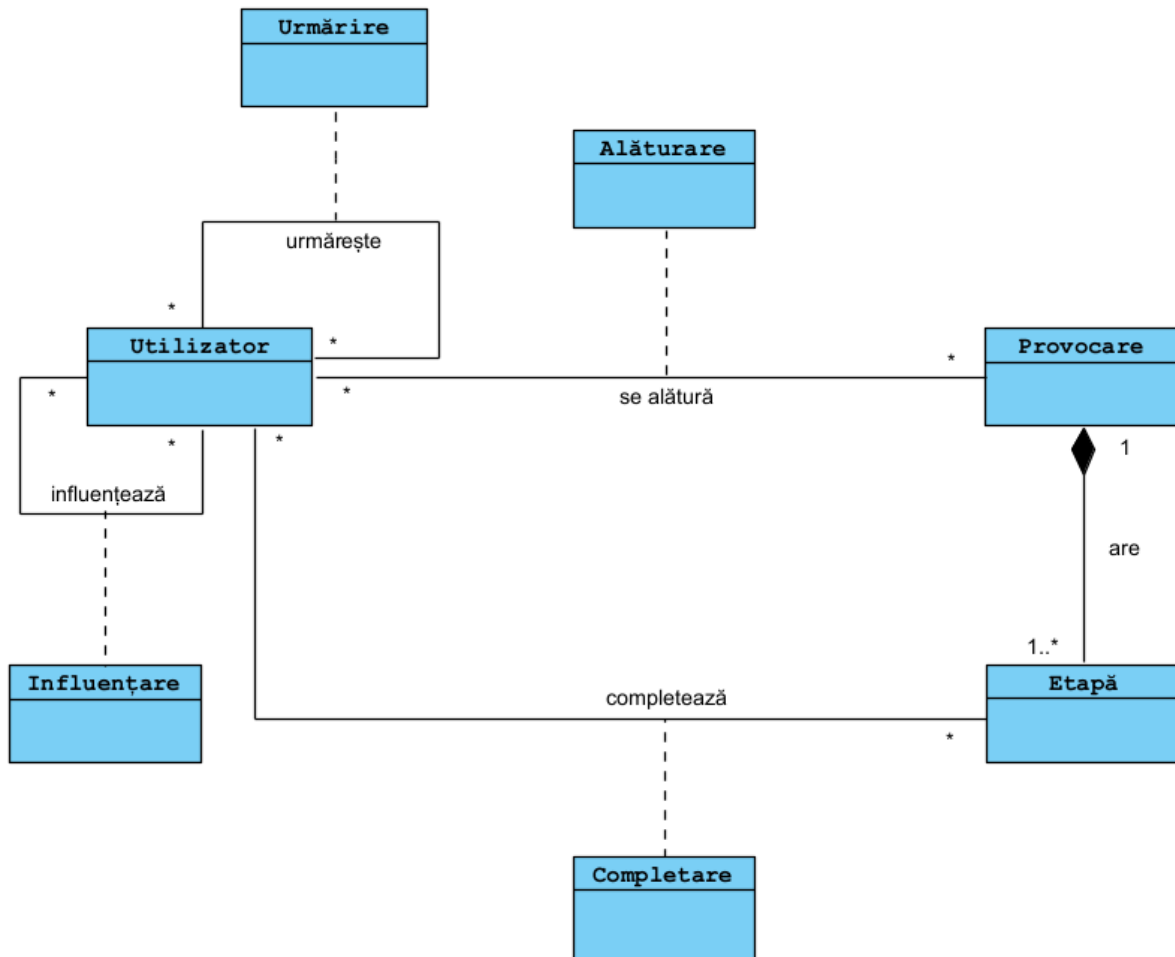


Figura 4 Diagrama de clase

Se poate observa faptul că entitățile principale sunt: *Utilizator*, *Provocare* și *Etapă*. Cu excepția relației de agregare compusă dintre *Provocare* și *Etapă* (unu-la-mulți), între celelalte clase principale există doar asocieri de tip mulți-la-mulți, astfel entitățile rămase, și anume: *Urmărire*, *Alăturare*, *Completare* și *Influențare* sunt clase ce conțin informații specifice asocierii pe care o modelează.

Pentru o mai bună reprezentare vizuală a fluxurilor de acțiuni prezentate în cele două descrieri textuale, vom înfăptui și diagramele de activitate specifice cazurilor de utilizare detaliate ale acestora. Prima acțiune pe care o vom analiza va fi cea de înrolare într-o provocare.

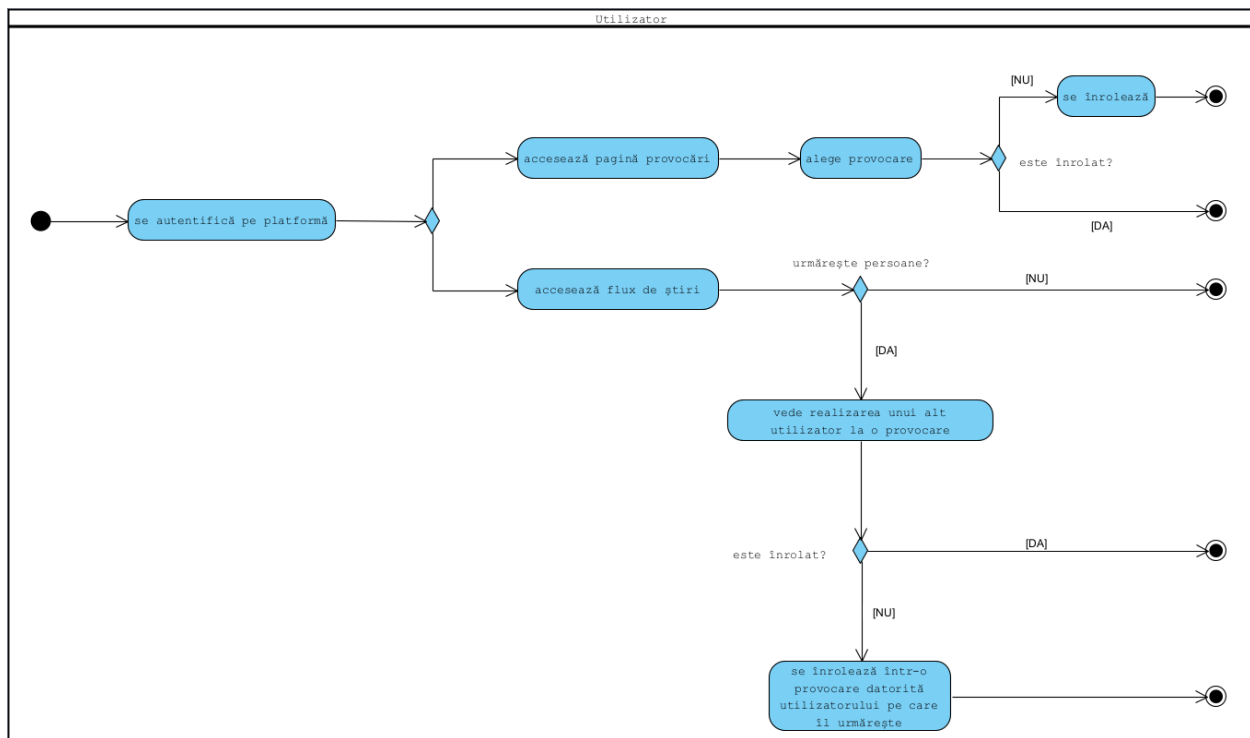


Figura 5 Diagrama de activitate pentru cazul de utilizare „se înrolează în provocare”

În scopul observării aspectelor dinamice ale acțiunii de înrolare în provocare, precum interacțiunea dintre obiecte, schimbul de mesaje și dimensiunea temporală a acestora, vom folosi diagrama de secvență. Un avantaj al acestei diagrame este că ne ajută să ne conturăm o viziune asupra viitorului cod, în special asupra metodelor pe care le vom crea.

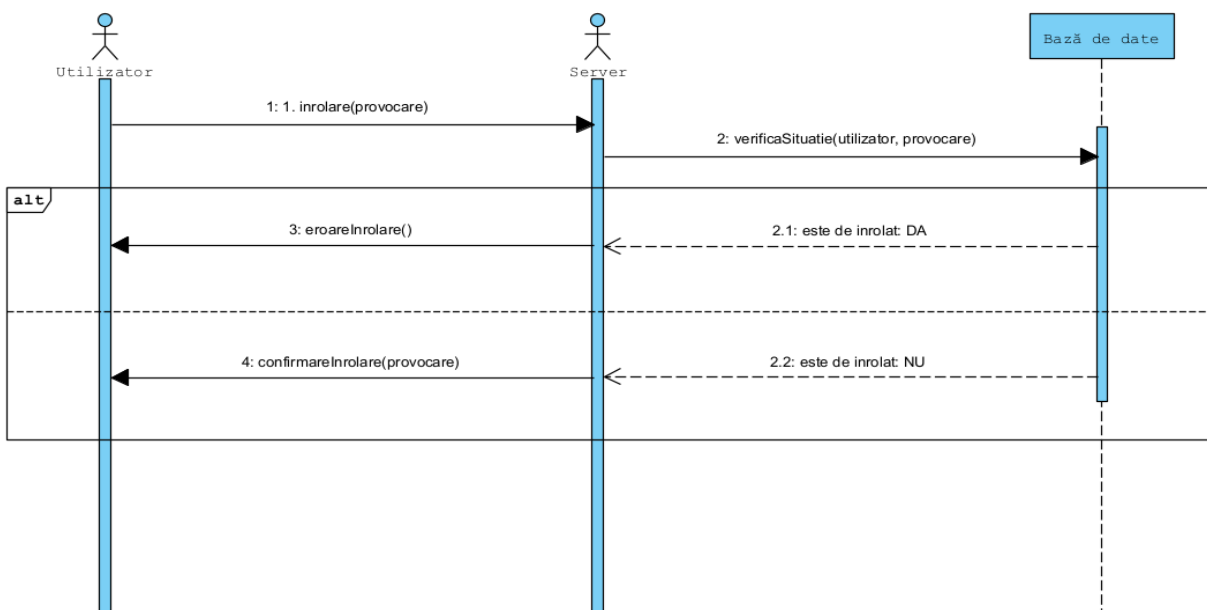
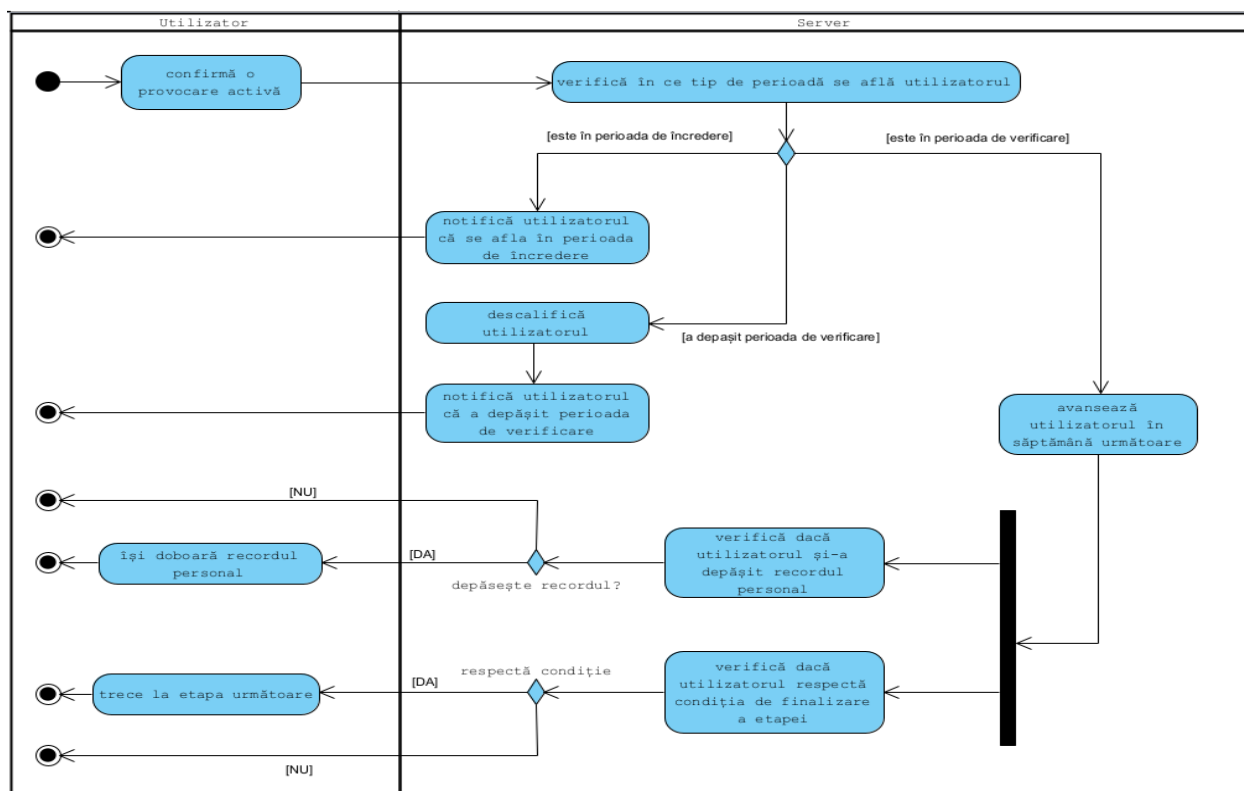


Figura 6 Diagrama de secvență pentru acțiunea de înrolare într-o provocare



Figură 7 Diagrama de activitate pentru cazul de utilizare „confirmă continuarea provocării”

După cum se poate observa în diagrama anterioară, sunt prezentate o serie de procese de afaceri ample, acțiunile serverului urmând a fi de fapt automatizate în produsul final, astfel am creat și diagrama de procese, specifică limbajului BPMN (Business Process Model and Notation).

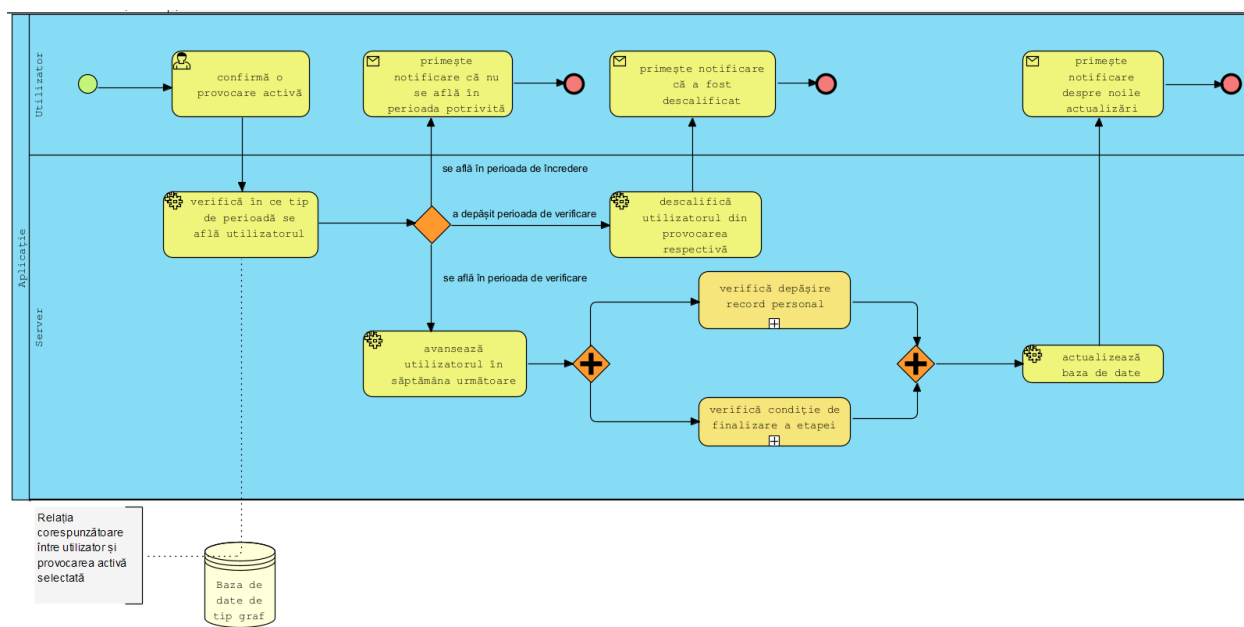


Figura 8 Diagrama de procese asociată procesului „confirmă continuare provocare”

Cu ajutorul diagramei de procese, sunt evidențiate elementele cheie pentru stadiul de proiectare și implementare, de pildă: intervențiile utilizatorului, acțiunile automatizate de tip serviciu, acțiunile de tipul primire sau trimitere mesaj. Totodată, această descriere deține un rol esențial în dezvoltarea și optimizarea algoritmilor creați pentru a îndeplini cu succes obiectivul procesului.

2.3 Proiectarea sistemului

Aplicația va fi creată în stadiul incipient în limba engleză pentru o deschidere la un public mult mai numeros, întrucât problema sustenabilității este o problemă de statut mondial. Elementele următoare, precum și codul sursă vor fi redactate tot în limba engleză pentru a avea ulterior o deschidere pentru dezvoltatori indiferent de naționalitate acestora.

Baza de date pe care urmează să o folosim va fi de o bază nerelațională, de tip graf, soluția fiind oferită de cei de la Neo4J. Decizia de a folosi baze de date de tip graf pornește de la vizualizarea efectivă a evoluției comunității prin prisma relațiilor ce se înfăptuiesc în cadrul acestora, la care se adaugă și performanța sporită față de o variantă relațională.

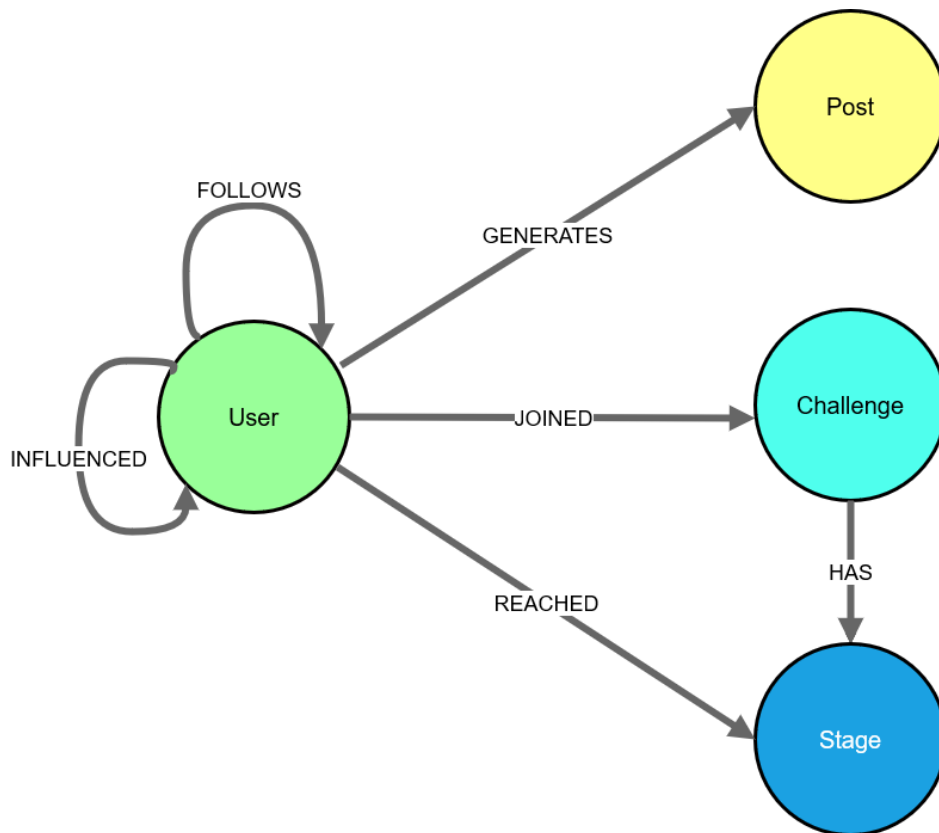


Figura 9 Schema bazei de date

După cum se poate observa în Figura 9, clasele principale identificate în etapa de analiză (Figura 4) s-au transformat în noduri, iar clasele de tip asociere în relații unidirecționale. Un element de noutate este secțiunea de postări, care funcționează astfel: o postare este un obiect care conține detalii despre activitatea unui utilizator la un moment anume și se generează automat într-unul dintre următoarele contexte:

- când utilizatorul se alătură unei provocări;
- când utilizatorul finalizează o etapă;
- când utilizatorul începe să urmărească alt utilizator.

Pentru o mai bună înțelegere a funcționalității bazei de date de tip graf, putem face o paralelă la bazele de date relaționale, considerând nodurile ca fiind tabelele entităților principale, iar relațiile – tabele de joncțiune, cu excepția relațiilor *HAS* și *GENERATES*.

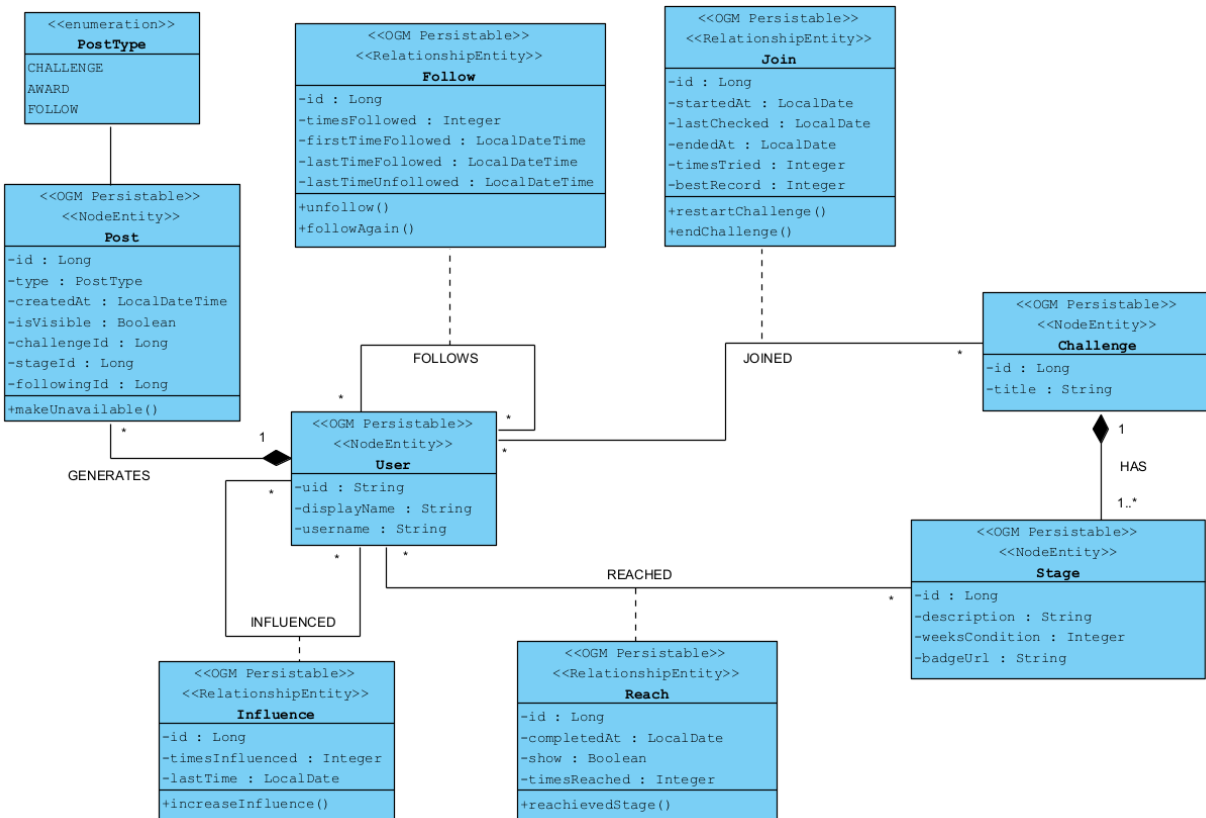


Figura 10 Diagrama de clase detaliată

Prin intermediul OGM (Object Graph Mapper) sunt mapate automat nodurile și relațiile din cadrul grafului într-un model de domeniu, abstractizând astfel baza de date și oferind o modalitate convenabilă de a păstra modelul de domeniu în graf fără a utiliza drivere de nivel scăzut.

Modelul arhitectural pe care îl vom folosi va fi **Model-view-controller** (în traducere liberă „model-vizualizare-controlor”, cunoscut adesea ca și **MVC** ([Anexa 6](#)). Motivul principal pentru folosirea acestei arhitecturi este puterea de modularizare a aplicației, astfel este izolată logica de business de interfața utilizatorului și fiecare nivel al aplicației este independent de celălalt ceea ce permite modificarea facilă, dar și scalabilitatea pe termen lung. [\[26\]](#)

Respectând modelul arhitectural stratificat ierarhic folosit de Spring Boot [\[27\]](#) ([Anexa 7](#)), în care fiecare strat comunică atât cu predecesorul, cât și cu succesorul său, există patru straturi în cadrul aplicației server:

- Stratul de prezentare (**Controller**) care se ocupă cu interacțiunea dintre client și server, gestionând diferite solicitări de tip HTTP.
- Stratul privind logica de business (**Service Layer**) care gestionează toată logica de business și prelucrează cererile primite folosindu-se de straturile de acces de date.
- Stratul de persistență (**Repository Layer**) care conține toată logica de stocare și traduce obiectele afacerii din și în baza de date, în cazul nostru prin OGM.
- Stratul bazei de date (**Database Layer**) la nivelul căruia au loc operațiunile de tip C.R.U.D.

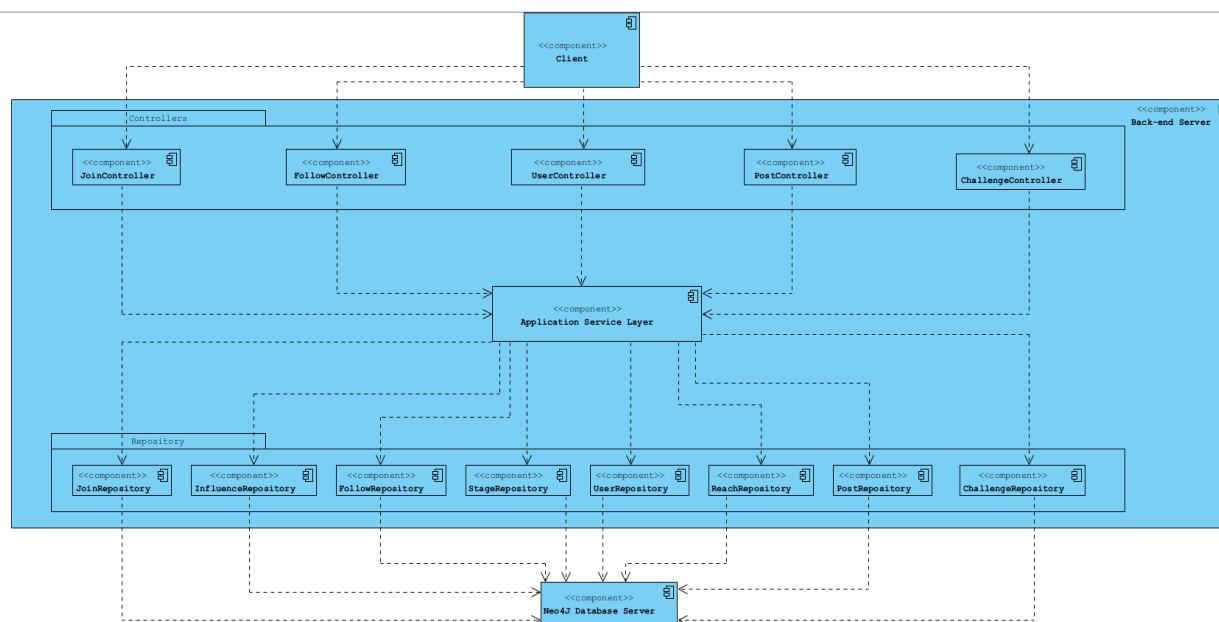


Figura 11 Diagrama de componente a sistemului

Fiecărui strat îi este asociat un pachet. După cum se poate observa, nu toate entitățile prezintă controllere, întrucât nu există interacțiune directă între client și acestea, prelucrările fiind realizate în stratul dedicat logicii de business. Diagrama de mai sus este varianta simplificată pentru a putea înțelege fluxul informațional, varianta completa putând fi regăsită în [Anexa 8](#).

Partea de autentificare va fi făcută prin intermediul unui back-end mobil ca serviciu (Backend-as-a-Service – BaaS), și anume Firebase Authentication, procesele fiind următoarele:

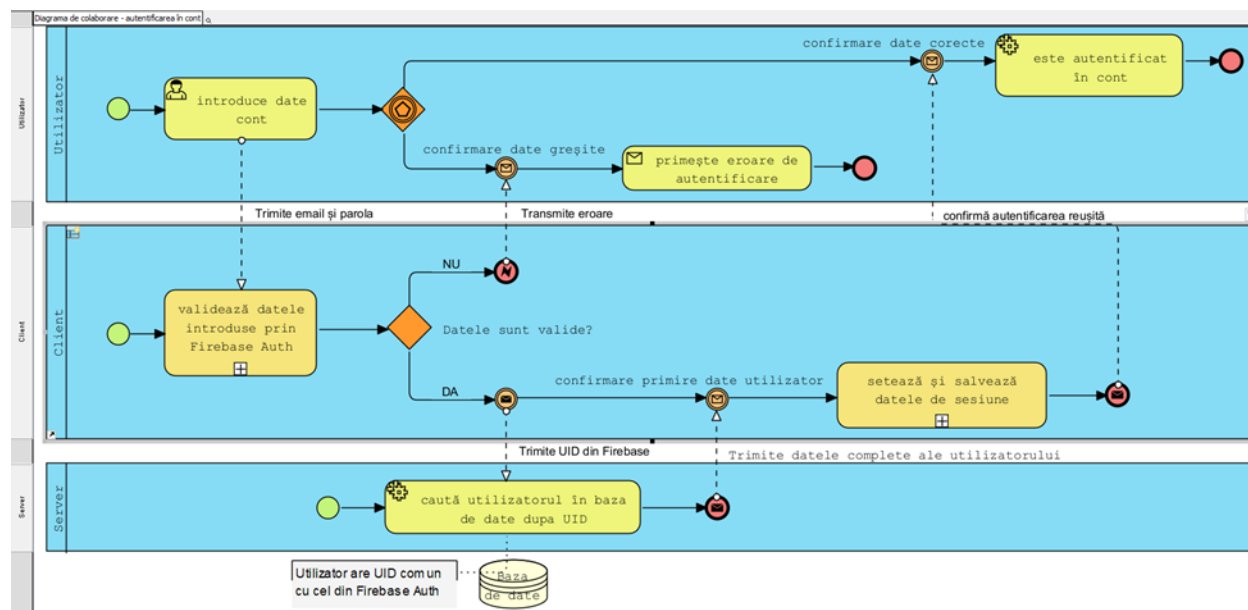


Figura 12 Diagrama de colaborare pentru procesul de autentificare în cont

Diagrama de colaborare este specifică limbajului BPMN și joacă în rol important în tranzațiile de mesaje și validări dintre diferite elemente ale sistemului complet, în cazul aplicației, între utilizatorul, serverul de client (*Client*), serverul de back-end (*Server*) și modulul de autentificare din Firebase.

Vom concluziona etapa de proiectare cu diagrama de desfășurare, specifică UML, prin care vom reprezenta mediul necesar pentru execuția fiecărei componentă software.

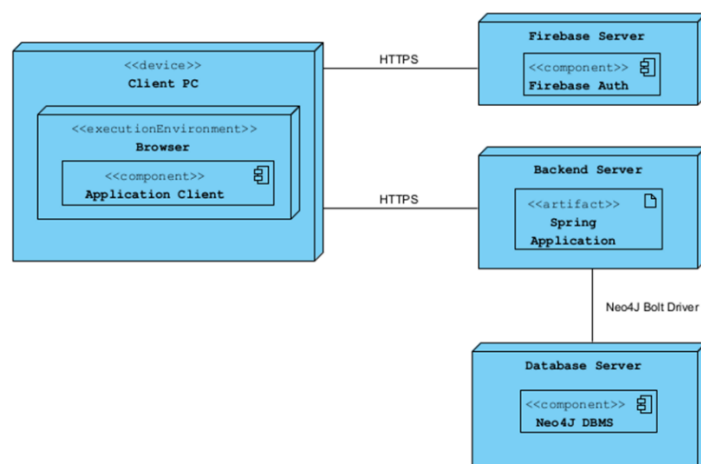


Figura 13 Diagrama de desfășurare a soluției

Capitolul 3. Implementarea sistemului informatic propus

3.1 Prezentarea tehnologiilor folosite în cadrul aplicației

Tehnologiile din cadrul suitei folosite pentru etapa de implementare a aplicației vor fi descrise în funcție de categoria din care fac parte.

Back-end (sau Backend) reprezintă dezvoltarea pe partea de server, concret porțiunea aplicației web pe care utilizatorului nu o vede în cadrul interacțiunii cu soluția, fiind responsabilă de managementul datelor. Pentru această secțiune am folosit versiunea 8 de Java în cadrul framework-ului Spring.

Spring Framework este un framework modular, *open-source* (cu sursă deschisă) ce folosește principiul de „inversare de control” (*Inversion of Control – IoC*) în cadrul aplicațiilor dezvoltate în limbajul Java, în special variantele EE (Enterprise Edition), mai nou existând și o versiune pentru platforma .NET. În cadrul aplicației am folosit următoarele module prin intermediul sistemului de administrare și versionare a dependențelor oferit de cei de la Apache, și anume **Maven**: [\[28\]](#)

- **Spring Boot**, care facilitează crearea aplicațiilor de producție independente printr-o simplă rulare. Web serverul folosit de acest modul este Apache Tomcat.
- **Spring Web**, care implementează modelul de arhitectură MVC pentru aplicațiile web.
- **Spring Data Neo4J**, oferă acces facil atât la configurarea și accesul la baza de date de tip graf Neo4J, cât și la driverul său de OGM (*Object Graph Mapping*).
- **Project Lombok**, o librărie dedicată simplificării sintaxei „prea verboasă” a limbajului Java. [\[29\]](#)

Baza de date folosită este una nerelațională și anume **Neo4J Graph Database**. Neo4J este un sistem de gestiune a bazelor de date de tip graf, fiind compatibilă cu proprietățile ACID (Atomicitate, Consistență, Izolare, Durabilitate) ale tranzacțiilor și oferind totodată un mediu de dezvoltare cu o interfață grafică pentru vizualizarea datelor și pentru scrierea diferitelor comenzi în limbajul mamă, Cypher Query Language. [\[30\]](#)

Front-end (sau Frontend) reprezintă dezvoltarea pe partea de client, fiind secțiunea cu care utilizatorul interacționează. Pentru această secțiune am folosit **ReactJS**, o bibliotecă JavaScript cu sursă deschisă creată de cei de la Facebook atât pentru aplicații mobile, cât și pentru aplicații web cu o singură pagină (*Single-Page Application*). În timp ce metoda implicită de funcționare a unui browser presupunea încărcarea integrală a diferitelor pagini web, acest tip de aplicație rescrie dinamic pagina web curentă cu date de la serverul web de back-end în momentul interacțiunii cu

utilizatorul. [31] În cadrul aplicației de client, am folosit următoarele dependențe gestionate printr-un manager de pachete JavaScript, numit **NPM** (*Node package manager*):

- **React Redux**, o bibliotecă JavaScript care gestionează starea aplicației astfel încât să afișeze interfețele utilizatorului corespunzător fluxului prezentat în [Anexa 9](#).
- **Node Sass**, o bibliotecă ce permite interpretarea sau compilarea în CSS (*Cascading Style Sheets*) a limbajului de script preprocesator Sass (*Syntactically Awesome Style Sheets*), limbaj ce oferă indexarea și simplificarea codului scris în CSS.
- **Axios**, o bibliotecă JavaScript folosită pentru gestiunea cererilor de tip HTTP.
- **Firebase**, o bibliotecă oferită de cei de la Google pentru introducerea serviciilor oferite de platforma Firebase.

Firebase este un „back-end ca serviciu” (*Backend-as-a-Service* sau *BaaS*), oferind o soluție completă în cloud de înlocuire a straturilor de logică de business, de persistență și al bazei de date. În cazul aplicației, folosim doar modulul de autentificare datorită securității bine pusă la punct de cei de la Google, dar și pentru posibilitatea autentificării în aplicație cu alte conturi deja existente pe diferite platforme consacrate, precum: Google, Facebook, Apple ș.a.m.d.

În final, pentru gestionarea versiunilor codului am folosit Git împreună cu Github. Git este o unealtă ce se instalează local pe dispozitivul pe care se face dezvoltarea și ajută la urmărirea evoluției codului, în timp ce Github este un serviciu de găzduire în cloud ce se ocupă de partea de stocare efectivă a versiunilor codului.

3.2 Implementarea soluției

Aplicația va fi împărțită în trei module: modulul de backend (serverul de backend), modulul de frontend (serverul de client) și modulul de baze de date. În cadrul produsului final, cele trei module vor funcționa ca aplicații separate și vor rula pe servere diferite, legătura dintre acestea făcându-se prin intermediul cererilor de tip HTTP.

Primul pas din cadrul implementării este descărcarea și configurarea tehnologiilor prezentate anterior pentru dezvoltarea soluției, astfel:

- Proiectul de Spring îl vom genera pe platforma [Spring Initializr](#) alături de toate dependențele necesare ([Anexa 10](#));
- Proiectul de ReactJS îl vom genera prin intermediul următoarei comenzi scrise în terminal la calea dorită și anume „*npm create-react-app collectio-web*”;
- Baza de date va fi creată și gestionată prin intermediul soluției Neo4J Desktop, fiind obligatorie configurarea unui utilizator responsabil de gestiunea acesteia.

Pe baza diagramei de clase detaliată din Figura 10, următorul pas se bazează pe crearea claselor corespunzătoare. Datorită dependenței Spring Data Neo4J, este nevoie doar de folosirea adnotărilor **@NodeEntity** și **@RelationshipEntity** pentru a face entitățile persistabile și gestionabile de OGM.

```

10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
@Data
@NoArgsConstructor
@AllArgsConstructor
@NodeEntity
public class User implements Serializable {
    @Id
    private String uid;

    private String displayName;

    private String username;

    private String email;

    @JsonIgnore
    @Relationship(type = "GENERATES", direction = Relationship.OUTGOING)
    private List<Post> posts = new ArrayList<>();
}

@Data
@AllArgsConstructor
@NoArgsConstructor
@RelationshipEntity(type = "JOINED")
public class Join {
    @Id
    @GeneratedValue
    private Long id;

    @StartNode
    private User user;

    @EndNode
    private Challenge challenge;

    private LocalDate startedAt;

    private LocalDate endedAt;
}

```

Figura 14 Exemple gestionare și persistare OGM pentru noduri de tip "User" și relații de tip "JOINED"

Toate entitățile au nevoie de un identificator unic, care va fi de tipul numeric și va fi autogenerat prin intermediul unei secvențe, cu excepția clasei destinate utilizatorului (*User*) care va avea de un identificator de tip șir de caractere care va corespunde cu cel generat de Firebase Authentication în momentul creării unui cont.

Totodată, se poate observa faptul că în cadrul unei entități de tip relație trebuie specificate nodul de start (**@StartNode**) și nodul de final (**@EndNode**); raportat la diagramă – acestea sunt entitățile între care se realizează asocierile.

Pentru a stabili conexiunea între baza de date și serverul de backend trebuie configurat fișierul de conexiune cu datele corespunzătoare ([Anexa 11](#)). În cadrul dezvoltării, Neo4J Desktop va crea un server local pentru accesul la baza de date, conexiunea dintre server și baza de date realizându-se prin intermediul unui driver numit Bolt, ce este integrat în Spring Data Neo4J.

Operațiile C.R.U.D. declanșate prin intermediul cererilor de tip HTTP asupra entităților reprezintă pasul următor în implementare, așadar putem verifica dacă maparea, persistarea și gestionarea obiectelor se realizează corect. Vizualizarea rezultatelor este vizibilă atât în cadrul secțiunii de răspuns din cererea emisă cât și în baza de date ([Anexa 12](#)). Un aspect important este identitatea clasei, concret – ce face o instanță să se deosebească de altă, lăsând la o parte identificatorul unic. Prin urmare, trebuie adăugate o serie de validări și de restricții pentru fiecare entitatea și algoritm, acolo unde este nevoie (de exemplu, provocările se deosebesc după titlu).

Cum am enunțat și în etapa de analiză, activitatea principală din cadrul aplicației este înrolarea în diferite provocări, la care se adaugă și procesul de confirmare al continuării unei provocări. Algoritmul de înrolare va fi o unul de tip UPSERT (update or insert), ce presupune faptul că va verifica condiția de existență a relației de tip JOINED, iar în caz afirmativ va actualiza corespunzător datele relației (va reîncepe provocarea doar dacă valoarea atributului *endedAt* este diferită de null); în cazul în care nu există relația dintre utilizator și provocare, o va crea.

```
public Join upsert(String userId, Long challengeId) {
    Join result = getByNodesIds(userId, challengeId);
    if (result == null) {
        Optional<User> persistedUser = userService.getById(userId);
        Optional<Challenge> persistedChallenge = challengeService.getById(challengeId);
        if (!persistedUser.isPresent() || !persistedChallenge.isPresent()) {
            LOGGER.warn("user(id=" + userId + ") or challenge(id=" + challengeId + ") does not exist");
            return null;
        }
        Join join = new Join(persistedUser.get(), persistedChallenge.get());
        try {
            createChallengePost(join.getUser(), join.getChallenge());
            return joinRepository.save(join);
        } catch (InvalidPostException e) {
            e.printStackTrace();
        }
    }
    Join joinUpdate = joinUpdate(result);
    if (joinUpdate != null) {
        return joinUpdate;
    }
    LOGGER.info(result + " is still active");
    return null;
}

private Join joinUpdate(Join result) {
    if (result.getEndedAt() != null) {
        try {
            result.restartChallenge();
            createChallengePost(result.getUser(), result.getChallenge());
            return joinRepository.save(result);
        } catch (InvalidPostException e) {
            e.printStackTrace();
        }
    }
    return null;
}
```

Figură 15 Algoritmul de înrolare într-o provocare în cadrul stratului de logică de business

Totodată, este posibilă înrolarea într-o provocare datorită altui utilizator, tot acest proces fiind de fapt identic cu cel anterior la care se adaugă apelarea metodei de tip UPSERT din cadrul serviciului de influențare (*InfluenceService.class*) în momentul în care înrolarea este făcută cu succes. Singura diferență dintre cele două este punctul final de comunicare (*Endpoint*).

```

@PutMapping("/{userId}->{challengeId}")
public ResponseEntity<Join> add(@PathVariable String userId, @PathVariable Long challengeId) {
    Join result = joinService.upsert(userId, challengeId);
    if (result == null || result.getEndedAt() != null) {
        LOGGER.error(" JOINED relationship requested with userId= " + userId +
            ", challengeId= " + challengeId +
            " does not exist or it's still active ");
        return ResponseEntity.badRequest().build();
    }
    return ResponseEntity.status(HttpStatus.CREATED).body(result);
}

@PutMapping("/{userId}->{challengeId}/influencedBy={influencerId}")
public ResponseEntity<Join> addByInfluenced(@PathVariable String userId,
    @PathVariable Long challengeId,
    @PathVariable String influencerId) {
    Join result = joinService.upsert(userId, challengeId);
    if (result == null || result.getEndedAt() != null) {
        LOGGER.error("JOINED relationship requested with userId= " + userId +
            ", challengeId= " + challengeId +
            ", influencerId= " + influencerId +
            " does not exist or it's still active. ");
        return ResponseEntity.badRequest().build();
    }
    Influence influence = influenceService.upsert(influencerId, userId);

    LOGGER.info("Influence relationship created" + influence);
    return ResponseEntity.status(HttpStatus.CREATED).body(result);
}

```

Figură 16 Endpoint-urile corespunzătoare înrolării (simplă și prin influență) într-o provocare în stratul de prezentare

Algoritmul corespunzător confirmării unei provocări declanșează o reacție în lanț la nivelul mai multor componente.

În primul rând, verifică starea relației dintre utilizator și provocare, parcurgând următoarele criterii: existența relației, validitatea relației (dacă este activă, adică valoarea atributului *endedAt* să fie nulă) și perioada în care se află utilizatorul în cadrul provocării respective. Ultimul criteriu este calculat în raport cu data curentă a serverului, prin urmare se va face diferența dintre data ultimei actualizări (atributul *lastChecked*) și data curentă, iar dacă valoarea rezultată este mai mare sau egală 4 și mai mică decât 7 (perioada de verificare) atunci se vor declanșa procesele:

1. Procesul de înaintare în cadrul provocării, ce presupune actualizarea recordului personal, dacă este cazul, și a atributului corespondent ultimei actualizări cu ultima sa valoare la care se adună 7 zile.

```

118 public Join checkChallengeActivity(String userId, Long challengeId) {
119     Join result = getByNodesIds(userId, challengeId);
120     if (result == null || result.getEndedAt() != null) {
121         return null;
122     }
123     int daysBetween = (int) ChronoUnit.DAYS.between(result.getLastChecked(), LocalDate.now());
124     if (daysBetween < 4) {
125         LOGGER.info("User (id = " + userId + ") is still in trust period for challenge (id= " + challengeId + ")");
126         return null; //trust days period -> you cant check now
127     }
128     if (daysBetween < 7) {
129         result.checkChallenge();
130         reachService.checkCompletedStage(result);
131         return joinRepository.save(result);
132     }
133     return endChallenge(userId, challengeId);
134 }
135 }

```

```

61
62 public void checkChallenge() {
63     this.lastChecked = this.lastChecked.plusDays(7);
64     int weeks = (int) ChronoUnit.WEEKS.between(this.startedAt, this.lastChecked);
65     if (weeks > this.bestRecord) {
66         this.bestRecord = weeks;
67     }
68 }

```

Figură 17 Algoritm de confirmare al activității în cadrul unei provocări

2. Procesul de verificare al situației în cazul completării unei etape. La nivelul acestuia sunt preluate etapele provocării ordonate descrescător după condiția privind numărul de săptămâni, aspect realizat pe baza unei interogări personalizate în limbajul Cypher (Figura 18). Ulterior, se calculează numărul de săptămâni prin diferența dintre momentul de început al provocării (atributul *startedAt*) și noua valoare a atributului *lastChecked*. Numărul de săptămâni calculat va fi criteriul de comparație cu condiția din cadrul etapelor. În momentul în care diferența este egală cu condiția de săptămâni, se va crea sau actualiza relația de tip REACH dintre utilizator și etapa respectivă, iar în cazul în care utilizatorul are deja o medalie afișată în raport cu provocarea în cauză, aceea se va ascunde și se va afișa cea din noua relația (sau relația nou actualizată). De menționat este faptul că un utilizator poate avea prezentă pe profil o singură medalie pentru fiecare provocare.

```

@Repository
public interface StageRepository extends Neo4jRepository<Stage, Long> {
    @Query("MATCH (challenge:Challenge)-[:HAS]->(stage:Stage) " +
        "WHERE id(challenge) = $challengeId " +
        "RETURN stage " +
        "ORDER BY stage.weeksCondition DESC")
    List<Stage> findAllByChallenge(Long challengeId);
}

```

Figura 18 Interogare pentru obținerea etapelor unei provocări ordonate descrescător în stratul de persistență a datelor


```

public Reach checkCompletedStage(Join join) {
    int currentWeeks = (int) ChronoUnit.WEEKS.between(join.getStartedAt(), join.getLastChecked());
    List<Stage> orderedStagesDesc = stageService.getAllByChallengeId(join.getChallenge().getId());
    if (orderedStagesDesc == null || orderedStagesDesc.isEmpty()) {
        LOGGER.error("Challenge does not have stages");
        return null;
    }
    for (Stage stage : orderedStagesDesc) {
        if (currentWeeks == stage.getWeeksCondition()) {
            hideActiveBadge(join);
            return upsert(join.getUser().getUid(), stage.getId());
        }
    }
    return null;
}

private void hideActiveBadge(Join join) {
    Reach updatedReach = reachRepository.hideActiveBadgeFromChallenge(join.getUser().getUid(),
        join.getChallenge().getId());
    if (updatedReach != null && updatedReach.getShow()) {
        LOGGER.error("Reach updated fail");
    }
}
}

```

Figura 19 Algoritm de verificare al completării unei etape

De asemenea, la nivelul procesului de anulare a unei provocări, trebuie verificată relația dintre utilizator și provocare, iar în cazul în care data încheierii (*endedAt*) este nulă, aceasta se completează cu data curentă.

```

JoinController.java
97 @PostMapping("/end/{userId}-{challengeId}")
98 public ResponseEntity<Join> endChallenge(@PathVariable String userId, @PathVariable Long challengeId) {
99     Join result = joinService.endChallenge(userId, challengeId);
100     if (result == null) {
101         LOGGER.error("JOINED relationship requested with userId= " + userId +
102             ", challengeId= " + challengeId +
103             " does not exist or it's already ended. ");
104         return ResponseEntity.badRequest().build();
105     }
106     return ResponseEntity.status(HttpStatus.OK).body(result);
}

JoinService.java
109 public Join endChallenge(String userId, Long challengeId) {
110     Join result = getByNodesIds(userId, challengeId);
111     if (result == null || result.getEndedAt() != null) {
112         return null;
113     }
114     result.endChallenge();
115     return joinRepository.save(result);
116 }

Join.java
54 public void endChallenge() {
55     this.endedAt = LocalDate.now();
56     int weeks = (int) ChronoUnit.WEEKS.between(this.startedAt, this.endedAt);
57     if (weeks > this.bestRecord) {
58         this.bestRecord = weeks;
59     }
}

```

Figură 20 Algoritm de anulare a unei provocări în cadrul straturilor din arhitectura Spring Boot

Prin prisma faptului că Neo4J suportă o cantitate foarte mare de date, peste 34 de miliarde de noduri, am venit cu un algoritm performant din punct de vedere al vitezei de răspuns pentru prelucrarea postărilor care vor afișate în cadrul atât la nivelul componentei de flux de știri, cât și în componenta ce reprezintă activitatea personală.

O postare este de fapt o clasă ce conține informații despre activitatea unui utilizator la un moment în timp și poate fi de 3 tipuri: de înrolare în provocare (*CHALLENGE*), de premiere (*AWARD*) și de urmărire a unei persoane (*FOLLOW*). Când un utilizator realizează cu succes una dintre acțiunile menționate, va genera automat o postare. Întrucât o postare are mai multe atribute care nu vor fi necesare pentru fiecare tip de postare, am folosit un Design Pattern creațional consacrat, și anume Builder. ([Anexa 13](#))

Pentru a transmite exact datele necesare afișării pe pagina web am creat un DTO (Data Transfer Object) pentru a evita posibilele cereri suplimentare. Pentru a spori iarăși performanța și a nu solicita memoria serverului de client, am folosit Pageable API din Spring care ne permite să aducem treptat postările și nu toate deodată. Principiul de funcționare este următorul: Pageable împarte numărul total de înregistrări la numărul specificat într-un set și astfel obține un număr de seturi, numit și pagini. Singurul lucru care trebuie specificat în cererea HTTP este pagina dorită, ceea ce în modulul de frontend vom automatiza: când utilizatorul dorește să vadă mai multe postări, vom aduce și pagina următoare prin intermediul apăsării unui buton.

```
@Query(value = "MATCH (loggedUser:User)-[:FOLLOWS]->(followings:User)-[rel:GENERATES]->(posts:Post) " +
    "WHERE loggedUser.vid = $loggedUserId AND posts.isVisible = true " +
    "RETURN posts, followings, rel",
    countQuery = "MATCH (loggedUser:User)-[:FOLLOWS]->(followings:User)-[:GENERATES]->(posts:Post) " +
    "WHERE loggedUser.vid = $loggedUserId AND posts.isVisible = true " +
    "RETURN count(posts)")
Page<Post> findNewsfeedPostsForLoggedUser(String loggedUserId, Pageable pageable);

@Query(value = "MATCH (loggedUser:User)-[rel:GENERATES]->(posts:Post) " +
    "WHERE loggedUser.vid = $loggedUserId AND posts.isVisible = true " +
    "RETURN loggedUser, posts, rel",
    countQuery = "MATCH (loggedUser:User)-[:GENERATES]->(posts:Post) " +
    "WHERE loggedUser.vid = $loggedUserId AND posts.isVisible = true " +
    "RETURN count(posts)")
Page<Post> findProfilePostForLoggedUser(String loggedUserId, PageRequest pageRequest);
```

Figura 21 Obținerea de postări pentru newsfeed și pentru activitatea personală folosind Pageable API în stratul de persistență

Modulul de autentificare oferit de cei de la Google prin soluția Firebase Authentication l-am folosit direct în cadrul modulului de frontend, oferind astfel în primă etapă posibilitatea de a te autentifica fie cu un email și o parolă, fie cu contul de Google. Pentru a stabili conexiunea a fost nevoie doar de prelucrarea unui fișier de configurare ce este furnizat de Firebase în momentul creării spațiului de dezvoltare.

```

1 import firebase from "firebase/app";
2 import 'firebase/auth';
3
4 const config = {
5   apiKey: "AIzaSyB4UovhcaVM6pLLKn0F9GwN9yeq0N0shJM",
6   authDomain: "collectio-auth.firebaseio.com",
7   projectId: "collectio-auth",
8   storageBucket: "collectio-auth.appspot.com",
9   messagingSenderId: "845062203591",
10  appId: "1:845062203591:web:bf923a37a083d00294b6e2",
11  measurementId: "G-DSZW7N2W9J"
12 };
13
14 firebase.initializeApp(config);
15
16 export const auth = firebase.auth();
17
18 const provider = new firebase.auth.GoogleAuthProvider();
19 provider.setCustomParameters({prompt: 'select_account'});
20 export const SignInWithGoogle = () => auth.signInWithPopup(provider);
21
22 export default firebase;

```

Figura 22 Configurare Firebase pentru autentificare prin email și parolă sau cont Google

Următorul pas face referință la diagrama de colaborare BPMN de la Figura 12, așadar în momentul în care utilizatorul își creează un cont nou sau se autentifică în cel deja existent cu succes, este trimisă o cerere HTTP de tip POST care va furniza identificatorul unic și emailul către serverul de backend, care va verifica la rândul lui dacă există utilizatorul în baza de date. Dacă există, îl va returna, dacă nu, îl va crea și va trimite utilizatorul nou creat.

Din cauza faptului că Firebase Auth stochează doar identificatorul unic și emailul ca elemente definitorii ale utilizatorului, utilizatorul este nevoit să-și seteze un nume de utilizator unic (atributul *username*) și un nume vizibil pentru ceilalți în momentul accesării aplicației pentru prima dată. Pentru a face aplicația mai interactivă, în momentul alegerii unui *username*, utilizatorului îi va fi asociat un avatar robot, unic, generat pe baza caracterelor din *username*, folosind un API oferit de cei de la [Robohash](https://robohash.org/).

```

<img src={`https://robohash.org/${username}?set=set1`} width='150' height='150'
alt='Profile avatar' />

```

Figura 23 Generarea avatarului unic în cadrul unui element imagine

Pentru a putea seta și accesa utilizatorul autentificat în toate componentele de ReactJS ale aplicației am folosit Redux. Pe lângă această caracteristică, Redux oferă și persistarea datelor utilizatorului autentificat la nivel de browser prin Redux Persist, componentă ce prelucrează datele sesiunii, astfel nu va fi nevoie de autentificare de fiecare dată când este accesată sau repornită

aplicația. Așadar, datele utilizatorului, la nivelul de stocare a stării din Redux, numit și *store*, mai sunt ținute și toate provocările din aceleași motive menționate anterior.

Toate acțiunile de atribuire se realizează în momentul în care modulul de frontend este accesat în componenta de bază (**App.js**), concret – la fiecare sesiune nouă de utilizare.

```
class App extends React.Component {
  constructor(props) {...}

  unsubscribeFromAuth = null;

  componentDidMount() {
    const {setCurrentUser, setAllChallenges} = this.props;
    this.unsubscribeFromAuth = auth.onAuthStateChanged( nextOrObserver: firebaseUser => {
      if (firebaseUser) {
        axios.post( url: 'http://localhost:8080/api/users',
          data: {
            uid: firebaseUser.uid,
            email: firebaseUser.email
          }
        )
        .then(response => {
          setCurrentUser(response.data);
        })
        .catch(reason => {
          console.log(reason)
          this.setState( state: {
            isServerAvailable: false
          })
        });
      } else {
        setCurrentUser(firebaseUser);
      }

      axios.get( url: 'http://localhost:8080/api/challenges/all' )
        .then(response => setAllChallenges(response.data))
        .catch(reason => console.log(reason))
    });
  }

  componentWillUnmount() {
    this.unsubscribeFromAuth();
  }
}
```

Figura 24 Setarea utilizatorului autentificat și a provocărilor prin intermediul Redux pe baza răspunsurilor cererilor HTTP

Componentele vizuale au fost create prin sintaxa JSX specifică ReactJS care prelucrează mai departe dinamic DOM-ul (Document Object Model) în funcție de acțiunile utilizatorului în cadrul aplicației web. Unul dintre avantajele principale, avantaj ce va fi regăsit și în cadrul aplicației, este posibilitatea de reutilizare a diferitelor componente, de pildă componenta asociată unei postări (**post.component.jsx**).

În final, baza de date de tip graf oferă un atuu raportat la analiza activității unui utilizator în cadrul aplicației, fiind vizibile direct relațiile ce se formează, evoluția utilizatorului, dar și impactul său la nivelul comunității.



Figura 25 Exemplu de scenariu simplu în cadrul aplicației

3.3 Prezentarea unei soluții pentru managementul stilului de viață - Collectio

Numele aplicației este dat de cuvântul latin „Collectio”. Cuvântul are în speță patru definiții, dar cea care caracterizează soluția software este „Actul de a colecta împreună”, care este de fapt și mottoul („The act of collecting. Together.”), la care se adaugă un mic artificiu, și anume acel punct pus înaintea lui „Together”, pentru a simboliza accentul pus pe comunitate.

Aplicația finală este creată în așa fel încât să dețină o interfață intuitivă, plăcută din punct de vedere al experienței de utilizator, astfel accentul să fie pus pe soluționarea problemei pentru care aceasta a fost dezvoltată. În continuare, vom urmări o serie de scenarii care vor descrie funcționalitățile de care se pot bucura utilizatorii în cadrul soluției create.

Înainte de toate, utilizatorul aplicației este nevoit să-și creeze un cont fie cu emailul și parola, fie cu contul de Google. Această obligație provine din dorința de a crea o comunitate, astfel fiecare individ să devină actorul principal al acestei schimbări de mentalitate și obiceiuri.

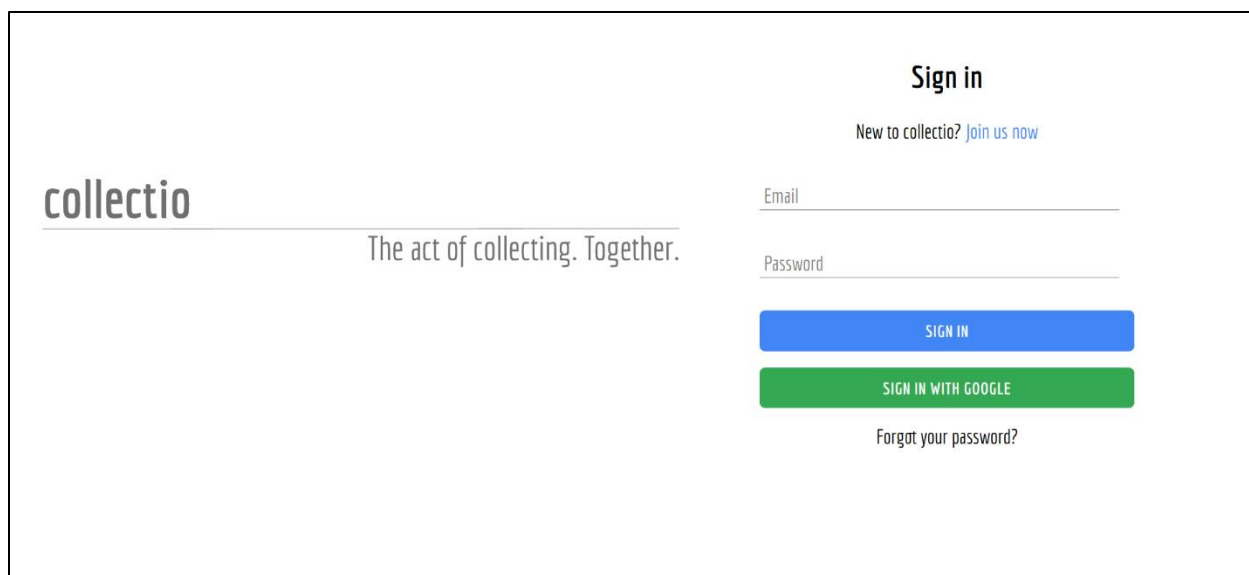


Figura 26 Pagina de autentificare în cont

După ce utilizatorul se autentifică în cont sau își creează unul nou, acesta este redirecționat către pagina principală. De menționat este faptul că la prima interacțiune cu aplicația, utilizatorul este întâmpinat de o fereastră modală în care este obligat să-și aleagă un nume de utilizator unic, care va descrie calea către profilul personal, și un nume care va fi vizibil pentru toți utilizatorii. La nivelul paginii principale, se regăsesc trei elemente: statisticile despre activitatea și impactul utilizatorului autentificat, fluxul de știri și cele mai populare provocări.

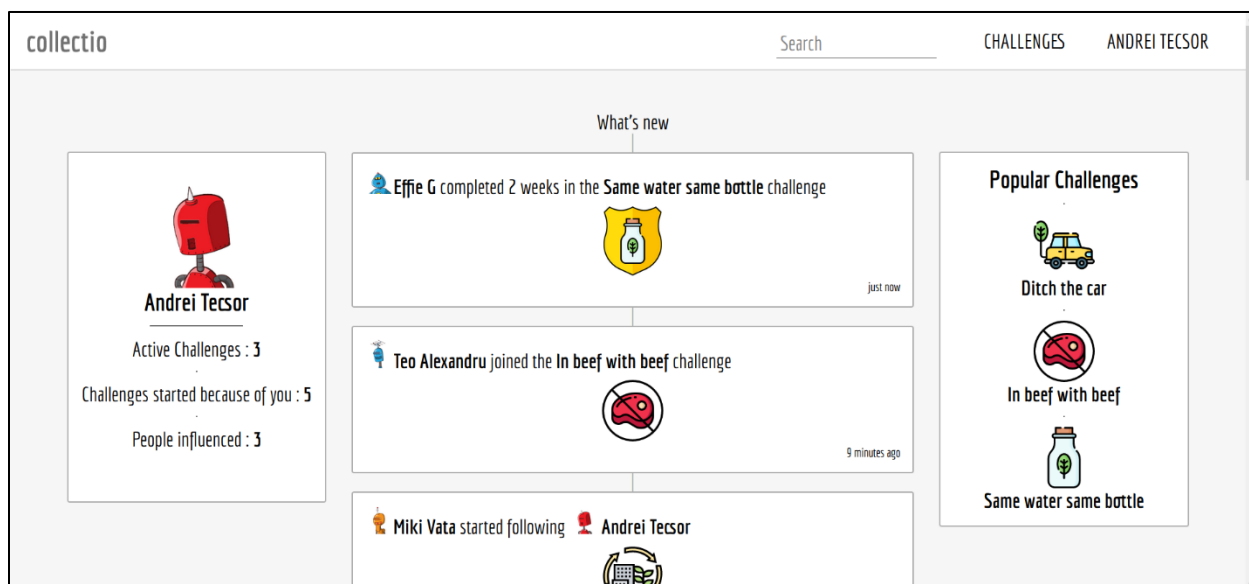


Figura 27 Pagina principală

În cadrul paginii principale, utilizatorul se poate înrola într-o provocare direct din postările pe care le vede sau din secțiunea ce prezintă cele mai populare provocări. Totodată, acesta se poate muta pe pagina de provocări cu ajutorul barei de navigare selectând opțiunea *CHALLENGES*.

Pagina de provocări este împărțită în două componente majore: componenta de gestiune a provocărilor active și componenta de vizualizare a tuturor provocărilor.

collectio

Search

CHALLENGES

ANDREI TECSOR

Active Challenges:




Challenge	Last Check	Best Record	Current Week	Check	End
 In beef with beef	2021-06-30	4	4	✓	✗
 Ditch the car	2021-06-30	14	14	✓	✗
 Same water same bottle	2021-06-30	2	2	✓	✗

Figura 28 Pagina de provocări – gestiunea provocărilor active

Utilizatorul poate vedea o serie de informații despre provocările sale active, precum: ultima validare, cel mai bun record în săptămâni până în prezent raportat la provocare și săptămâna curentă în care se află de când a început sau reînceput provocarea. Totodată, acesta are două opțiuni care îi definesc implicarea și anume: opțiunea de a-și valida activitatea în cadrul provocării, opțiune validă doar în „perioada de verificare” (după 4 zile de la ultima verificare) și opțiunea de a renunța la provocare oricând dorește.

În partea de jos a paginii se regăsește cea de-a doua componentă și anume cea care conține toate provocările vizibile în care utilizatorul se poate înrola la nivelul aplicației. Fiecare provocare are un logo specific, un nume sugestiv și o descriere care expune problema pe care aceasta încearcă să o soluționeze alături de acțiunea principală ce trebuie îndeplinită pentru acest scop. În momentul selectării unei provocări din listă, utilizatorul este întâmpinat de o fereastră modală care îi expune structura etapelor din provocarea respectivă, activitatea pe care trebuie să o facă în cadrul fiecărei etape, durata de timp și medaliile pe care le poate debloca. De asemenea, îi oferă posibilitatea de a se înrola în aceasta prin intermediul unui buton cu eticheta *JOIN NOW*.

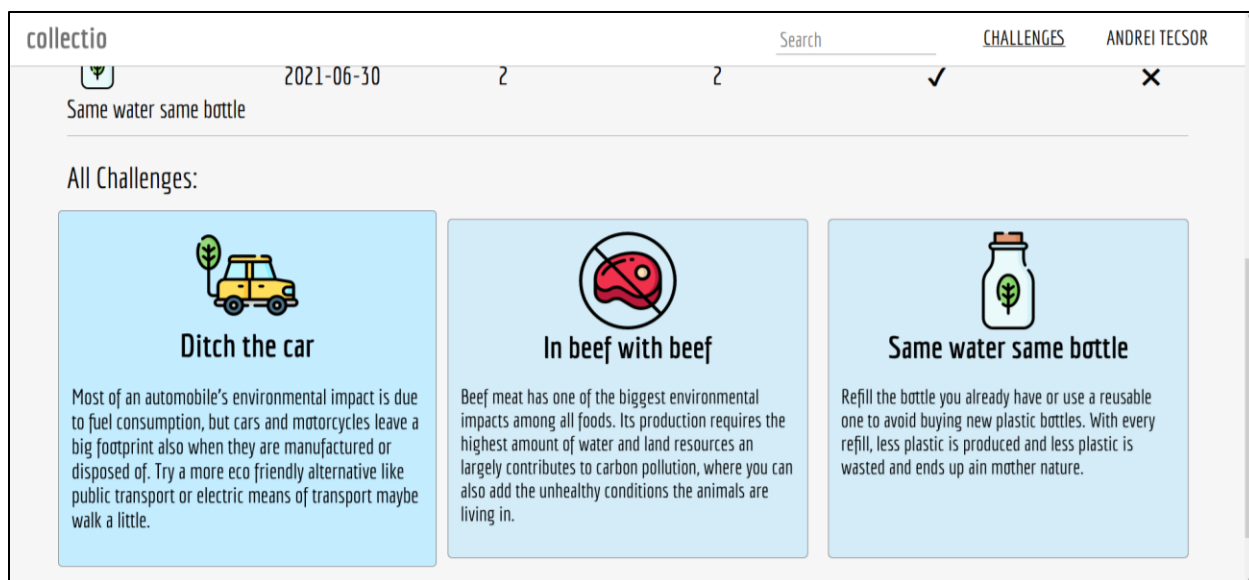


Figura 29 Pagina de provocări – toate provocările

Pe lângă cele menționate anterior, opțiunea de înrolare este disponibilă și la nivelul paginii de profil atât a utilizatorului autentificat, cât și a altui utilizator, fiind vorba de aceeași fereastră modală. Utilizatorul curent poate căuta alți utilizatori pe baza numelui prin intermediul elementului de tip Input Text din bara de navigare a cărei etichetă este *Search*.

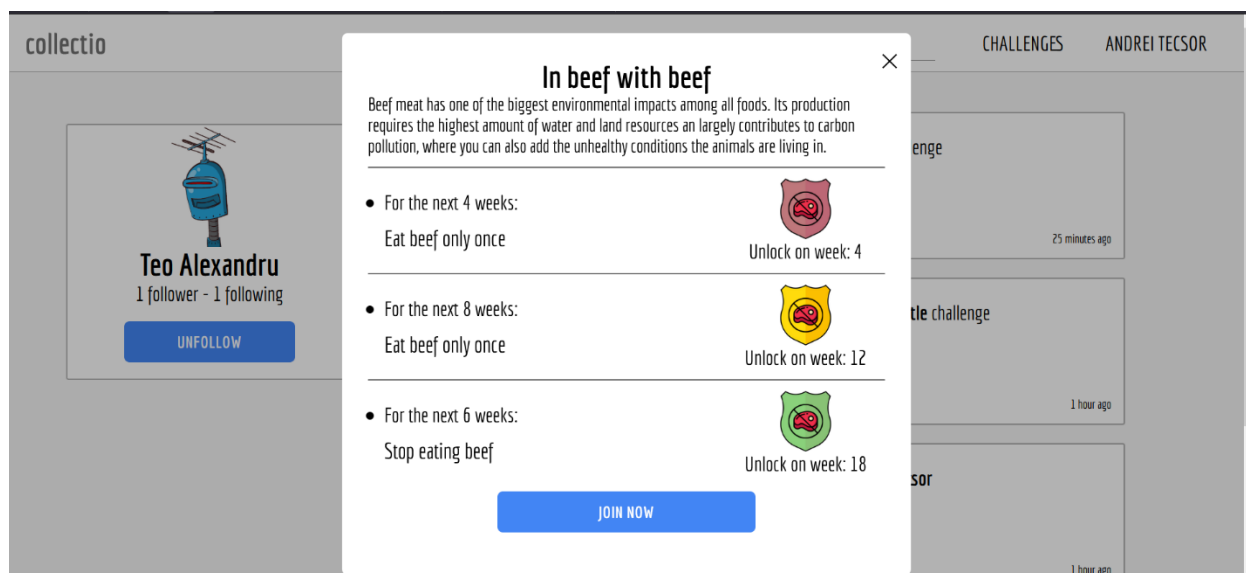


Figura 30 Fereastră modală de înrolare într-o provocare în cadrul paginii de profil a unui utilizator

În cadrul paginii de profil a oricărui utilizator diferit de cel autentificat sunt prezente trei componente majore:

- componenta de prezentare - care oferă detalii despre numărul de relații de interacțiune ale persoanei căutate alături de opțiunea utilizatorului autentificat de a se abona sau dezabona de la activitățile acesteia;
- medaliile obținute în urmă activității sale alături de cel mai bun record, numărul de încercări din cadrul provocării și ultima dată a deblocării medaliei respective (o medalie poate fi deblocată de mai mult ori);
- activitatea cronologică a utilizatorului.

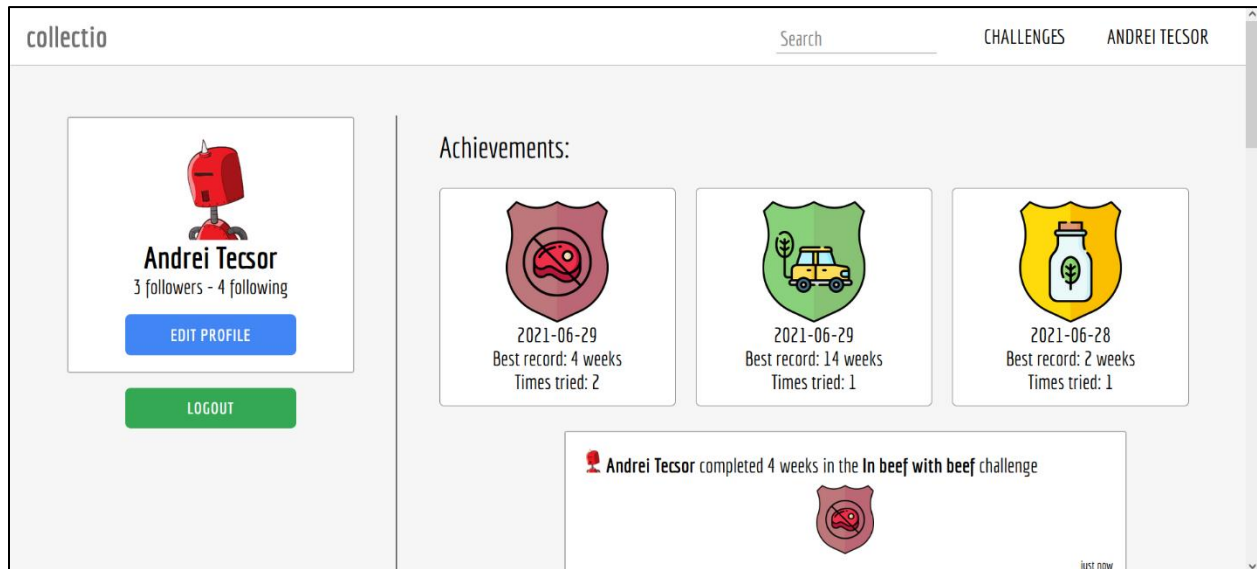


Figura 31 Pagina de profil a utilizatorului autentificat

Pagina personală de profil se poate accesa din bara de navigare, selectând numele utilizatorului conectat. Singura diferență se regăsește în cadrul componentei de prezentare, unde utilizatorul are opțiunea de a-și modifica detaliile contului și de a se deconecta de la aplicație.

Concluzii

În concluzie, soluția software dezvoltată are un real potențial în a juca un rol important în educația ecologică a individului, tratând activitățile de tip sustenabil în provocări ce au ca scop schimbarea traiului de viață într-unul durabil prin dobândirea de noi obiceiuri sănătoase pentru mediul înconjurător, reducând astfel amprenta ecologică de consum a persoanei. Totodată, posibilitatea de a-și putea observa impactul în cadrul comunității completează cu succes ultimele trepte ale piramidei lui Maslow datorită sentimentelor de recunoaștere, realizare și autoapreciere acumulate pe parcurs.

O realizare semnificativă pe termen lung a aplicației face referință la Efectul de expunere simplă (în engleză *Mere-exposure effect*) care enunță faptul că oamenii tind să dezvolte o preferință pentru diferite aspecte doar pentru că le sunt familiare [32]. Așadar, repetând acțiunile sustenabile, acestea devin din ce în ce mai familiare, oferindu-i o bunăstare interioară individului, dar și o serie nouă de valori, pe care, în timp, le va expune în cadrul societății.

Aplicația web prezentată este la prima versiune funcțională, versiune ce poate fi prezentată unui public larg. Totodată, aceasta a fost proiectată și implementată în așa fel încât să fie deschisă pentru noi funcționalități care i-ar putea spori atât rezultatele în raport cu soluționarea problemei principale, cât și performanțele din punct de vedere tehnic, concret - vizuale și algoritmice.

O funcționalitate ce ar mări tendința utilizării și dezvoltarea comunității ar fi cea de grupuri și anume: un utilizator își poate forma un grup cu minim doi prieteni și se pot înrola simultan în provocări. În momentul verificării, fiecare membru al grupului trebuie să valideze dacă activitatea celorlalți este în conformitate cu cerințele, asemenea unui sistem de tip blockchain. Deopotrivă, un avantaj major pentru individ ar fi posibilitatea contorizării zilnice a activităților ecologice.

Soluția software a fost creată integral de mine, atât modulul de backend, cât și modulul de frontend, pornind de la partea de analiză și până la implementare, având la bază cunoștințele acumulate de-a lungul anilor de facultate, alături de o multitudine de resurse din mediul online, precum cursuri, site-uri de specialitate, proiecte ș.a.m.d. Provocările sunt inspirate din diferite studii realizate de cercetători certificați și sunt gândite în așa fel încât să fie accesibile pentru toată lumea, indiferent de diversele contexte sociale.

Inspirația dezvoltării aplicației pornește de la observarea ultimelor tendințe în contextul rețelelor de socializare și lipsa de interes pe tema protejării mediului din cadrul acestora. Lucrarea își propune să lărgescă perspectiva cititorului, ajutându-l să descopere faptul că prin gesturi mici poate avea un impact substanțial atât la nivelul societății, cât și la nivelul mediului natural.

Bibliografie

1. Tien Ming Lee, Ezra M. Markowitz, Peter D. Howe, Chia-Ying Ko, Anthony A. Leiserowitz. (2015). *Predictors of public climate change awareness and risk perception around the world* https://www.nature.com/articles/nclimate2728.epdf?sharing_token=GAKFUXqYLTi5s0_ceg_ivzNRgN0jAjWel9jnR3ZoTv0PJgEpWYMgtckBjIRQs3UsNbKirRo2rHVOya8xVP2gF3pjXX5nFfyiTZqweyMePLQIqWN63QZ0ShhgZmej4T_NyUVgn4STsJ_K5Dqn2lDja989Npsirh3D-10pEzWXPkAIHLqcdlArLO4Y8CX0ulwNaJGZC207Ofv3TreTiBINuvc7HwpswLEb69BfCXH3egE%3D&tracking_referrer=www.carbonbrief.org - accesat la 17.03.2021
2. Wikipedia. (2019). *Abraham Maslow*. https://ro.wikipedia.org/wiki/Abraham_Maslow - accesat la 01.04.2021
3. Wikipedia. (2021). *Mathis Wackernagel*. https://en.wikipedia.org/wiki/Mathis_Wackernagel - accesat la 25.04.2021
4. Mathis Wackernagel, Williams E. Rees, Phil Testemale. (1995) *Our Ecological Footprint: Reducing Human Impact on the Earth*. New Society Publishers
5. Wikipedia. (2021). *Ecological footprint*. https://en.wikipedia.org/wiki/Ecological_footprint - accesat la 25.04.2021
6. Global Footprint Network. (2021). *Ecological Footprint*. <https://www.footprintnetwork.org/our-work/ecological-footprint/> - accesat la 25.04.2021
7. Global Footprint Network. (2021). *National Footprint and Biocapacity Accounts*. <https://www.footprintnetwork.org/resources/data/> - accesat la 25.04.2021
8. Wikipedia. (2021). *Biocapacity*. <https://en.wikipedia.org/wiki/Biocapacity> - accesat la 26.04.2021
9. David Lin, Laurel Hanscom, Adeline Murphy, Alessandro Galii, Mikel Evans, Evan Neill, Maria Serena Macini, Jon Martindill, Fatime-Zahra Medouar, Shiyu Huang, Mathis Wackernagel. (2018). *Ecological Footprint Accounting for Countries: Updates and Results of the National Footprint Accounts, 2012–2018*. <https://www.mdpi.com/2079-9276/7/3/58> - accesat la 01.05.2021
10. Global Footprint Network. (2021). *Earth Overshoot Day*. https://www.footprintnetwork.org/our-work/earth-overshoot-day/?_ga=2.247985799.1610304913.1621376394-1667256855.1620331094 - accesat la 05.05.2021
11. Wikipedia. (2019). *Consumerism* https://ro.wikipedia.org/wiki/Consumerism#cite_note-Modern_Consumerism-6 – accesat la 10.05.2021
12. Wikipedia. (2021). *Materialism economice* https://ro.wikipedia.org/wiki/Materialism_economic - accesat la 10.05.2021

13. Milos Djordjevic. (2021). *Global Social Media Usage Statistics: How Much Time Do People Spend on Social Media?*. <https://letter.ly/how-much-time-do-people-spend-on-social-media/> - accesat la 12.05.2021
14. Wikipedia. (2021). *Internet Celebrity*. https://en.wikipedia.org/wiki/Internet_celebrity - accesat la 20.05.2021
15. Bryan Lipiner. (2020). *What Is Influencer Marketing? An Industry on the Rise*. <https://entrepreneurship.babson.edu/what-is-influencer-marketing/> -accesat la 20.05.2021
16. Paris Martineau. (2019). *The WIRED Guide to Influencers*. <https://www.wired.com/story/what-is-an-influencer/> - accesat la 21.05.2021
17. Francisco J. Martinez-Lopez, Rafael Anaya-Sanchez, Marisel Fernandez Giordano, David Lopez-Lopez. (2020). *Behind influencer marketing: key marketing decisions and their effects on followers' responses*. Journal of Marketing Management - volumul 36
18. Leon Festinger. (1954). *A theory of social comparison processes*. Human Relations
19. Jiyoung Chae. (2018). *Explaining Females' Envy Toward Social Media Influencers*. Media Psychology
20. Wikipedia. (2018). *Obicei (psihologie)*. [https://ro.wikipedia.org/wiki/Obicei_\(psihologiei\)](https://ro.wikipedia.org/wiki/Obicei_(psihologiei)) – accesat la 03.06.2021
21. Phillippa Lally, Cornelia Van Jaarsveld, Henry Potts, Jane Wardle. (2009). *How are habits formed: Modelling habit formation in the real world*. European Journal of Social Psychology
22. Benjamin Gardner, Phillippa Lally, Jane Wardle. (2012). *Making health habitual: the psychology of 'habit-formation' and general practice*. British Journal of General Practice – volumul 62
23. Planet Patrol. (2021). *About Planet Patrol*. <https://planetpatrol.co/about/> - accesat la 15.06.2021
24. TrashOut. (2021). *What is TrashOut*. <https://www.trashout.ngo/about> - accesat la 15.06.2021
25. We don't have time. (2021). *About us*. <https://www.wedonthavetime.org/about-us> - accesat la 15.06.2021
26. Wikipedia. (2021). *Model–view–controller*. <https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller> – accesat la 18.06.2021
27. JavaTPoint. (2021). *Spring Boot Architecture*. <https://www.javatpoint.com/spring-boot-architecture> - accesat la 18.06.2021
28. Spring. (2021). *Spring Framework Documentation*. <https://docs.spring.io/spring-framework/docs/current/reference/html/> - accesat la 21.06.2021
29. Baeldung. (2021). *Introduction to Project Lombok*. <https://www.baeldung.com/intro-to-project-lombok> - accesat la 21.06.2021

30. Neo4J, Inc. (2021). *Neo4j Documentation*. <https://neo4j.com/docs/> - accesat la 21.06.2021
31. Facebook Inc. (2021). *ReactJS Getting Started*. <https://reactjs.org/docs/getting-started.html> - accesat la 21.06.2021
32. Wikipedia. (2021). *Mere exposure effect*. https://en.wikipedia.org/wiki/Mere-exposure_effect - accesat la 29.06.2021

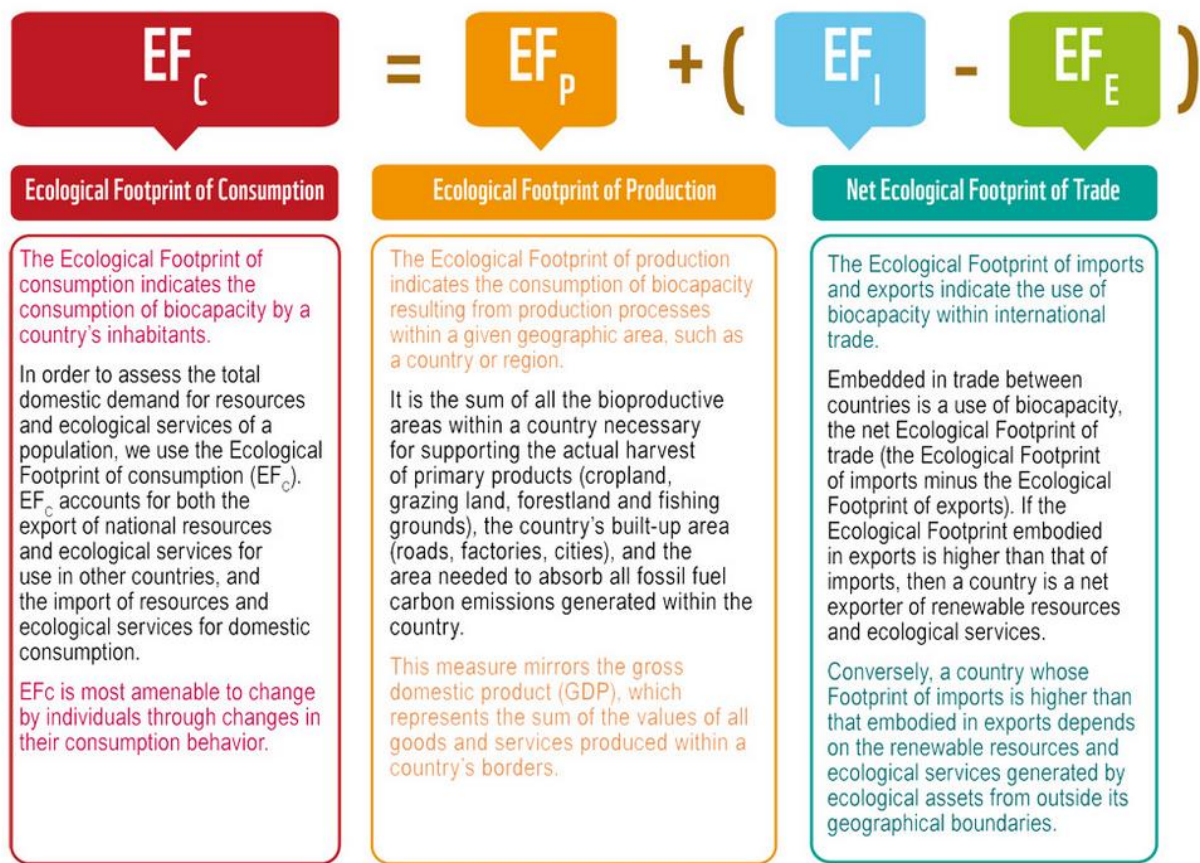
Anexe

Anexa 1. Piramida lui Maslow



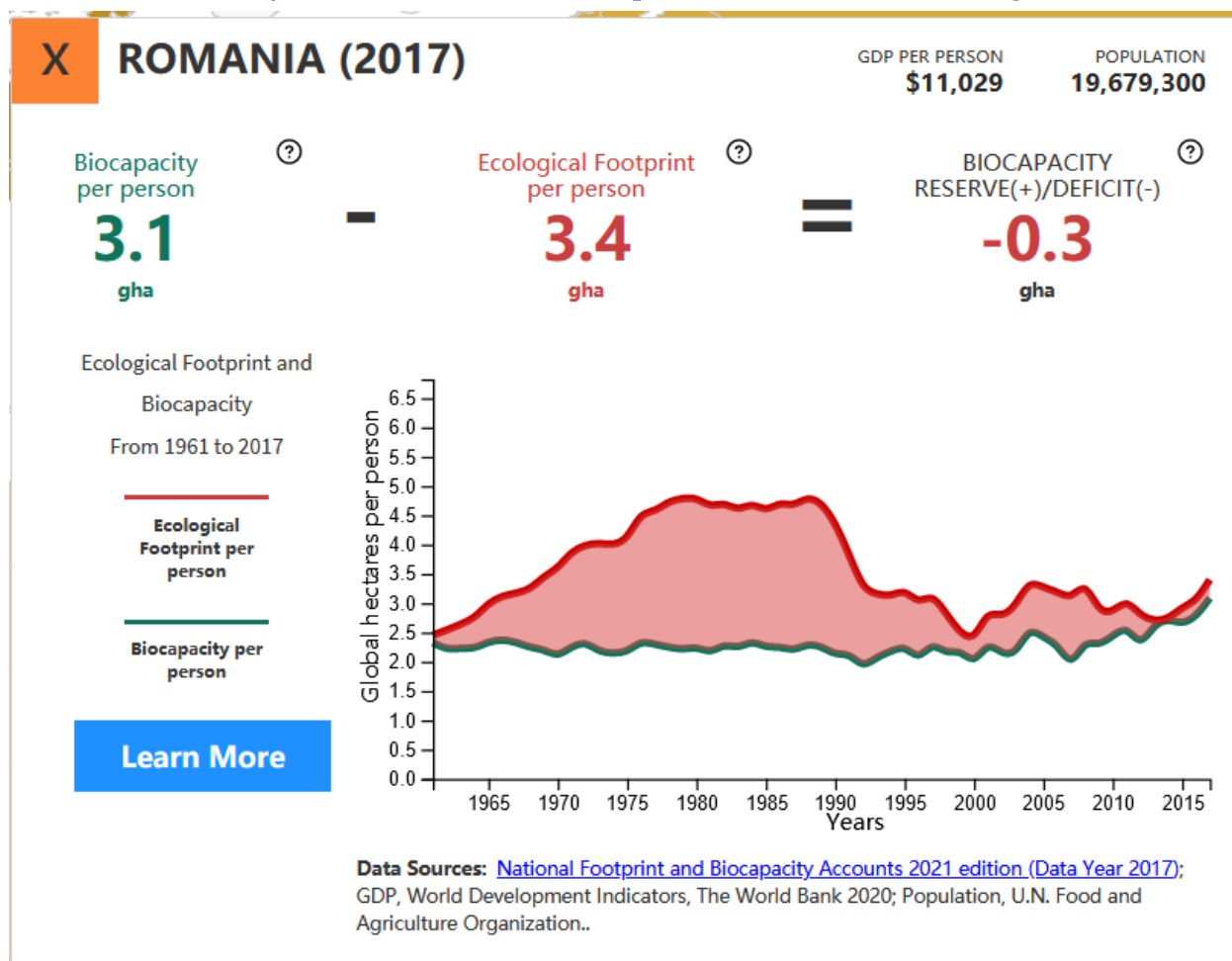
Sursă: <https://lemonice.ro/cum-sa-vinzi-cu-ajutorul-piramidei-lui-maslow/>

Anexa 2. Formula specifică analizei impactului ecologic al unei țări conform Global Footprint Network



Sursă: <https://www.footprintnetwork.org/resources/data/>

Anexa 3. Situația României în anul 2017 privind contabilitatea ecologică



Sursă: https://data.footprintnetwork.org/?_ga=2.98208278.886506416.1622578011-1667256855.1620331094#/

Anexa 4. Situația României în anul 2014 privind contabilitatea ecologică

X

ROMANIA (2014)

GDP PER PERSON

\$9306

POPULATION

19,972,700

Biocapacity
per person

2.7

gha

-

Ecological Footprint
per person

2.7

gha

=

BIOCAPACITY
RESERVE(+)/DEFICIT(-)

0.0

gha

Ecological Footprint and

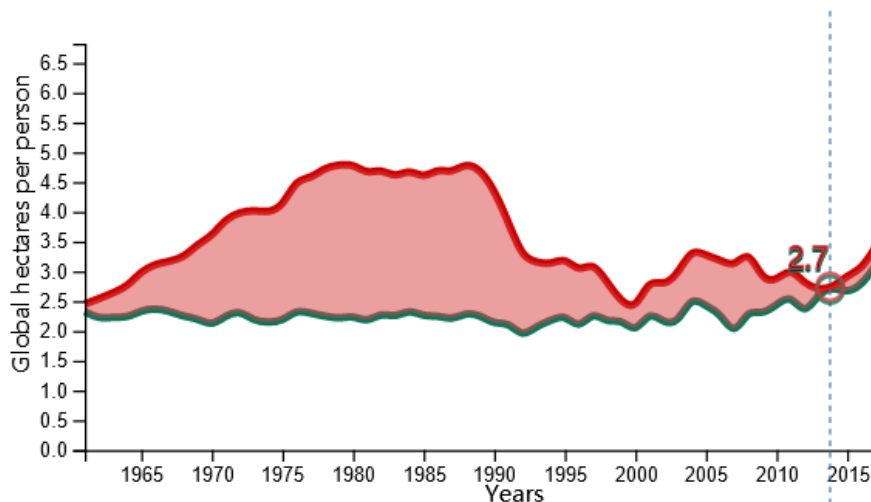
Biocapacity

From 1961 to 2017

Ecological
Footprint per
person

Biocapacity per
person

Learn More



Data Sources: [National Footprint and Biocapacity Accounts 2021 edition \(Data Year 2017\)](#); GDP, World Development Indicators, The World Bank 2020; Population, U.N. Food and Agriculture Organization..

Sursă: https://data.footprintnetwork.org/?_ga=2.98208278.886506416.1622578011-1667256855.1620331094#/

Anexa 5. Plan specific creării unui nou obicei

Box 1. A tool for patients

Make a new healthy habit

1. Decide on a goal that you would like to achieve for your health.
2. Choose a simple action that will get you towards your goal which you can do on a daily basis.
3. Plan when and where you will do your chosen action. Be consistent: choose a time and place that you encounter every day of the week.
4. Every time you encounter that time and place, do the action.
5. It will get easier with time, and within 10 weeks you should find you are doing it automatically without even having to think about it.
6. Congratulations, you've made a healthy habit!

My goal (e.g. 'to eat more fruit and vegetables') _____

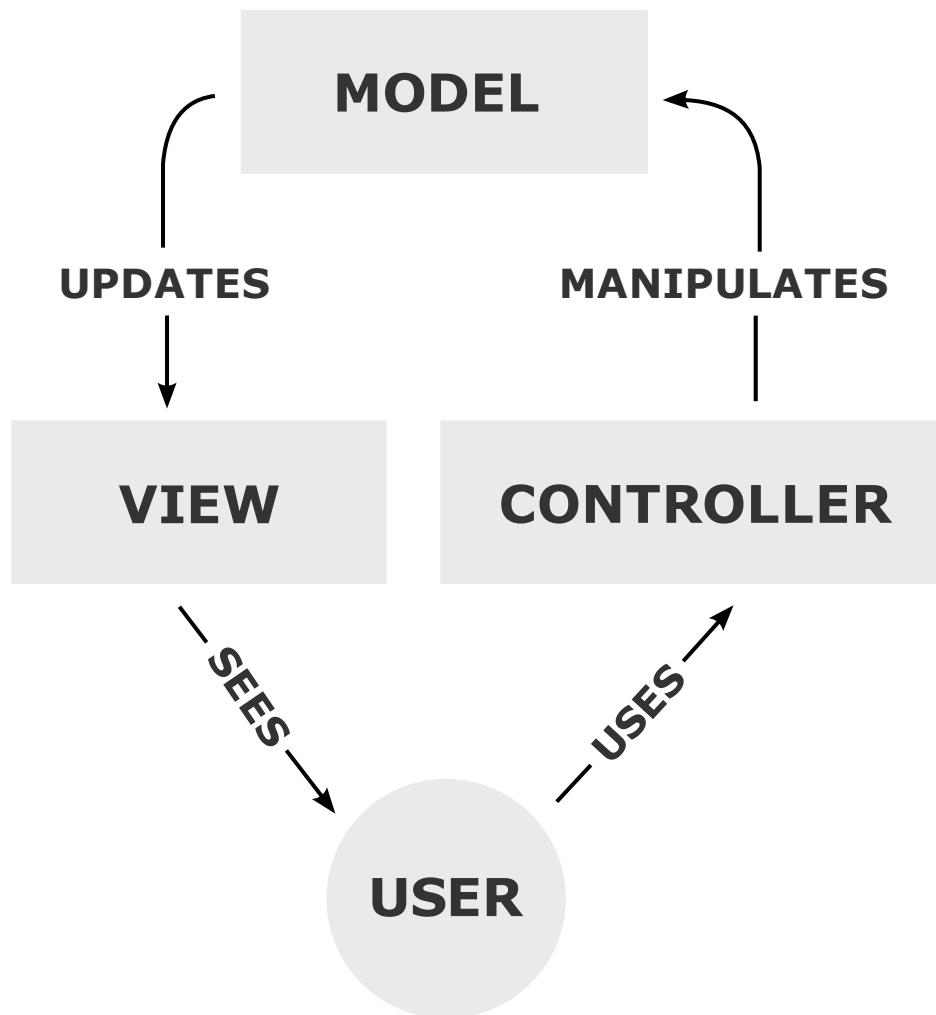
My plan (e.g. 'after I have lunch at home I will have a piece of fruit')
(When and where) _____ I will _____

Some people find it helpful to keep a record while they are forming a new habit. This daily tick-sheet can be used until your new habit becomes automatic. You can rate how automatic it feels at the end of each week, to watch it getting easier.

	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10
Monday										
Tuesday										
Wednesday										
Thursday										
Friday										
Saturday										
Sunday										
Done on >5 days, yes or no										
How automatic does it feel? Rate from 1 (not at all) to 10 (completely)										

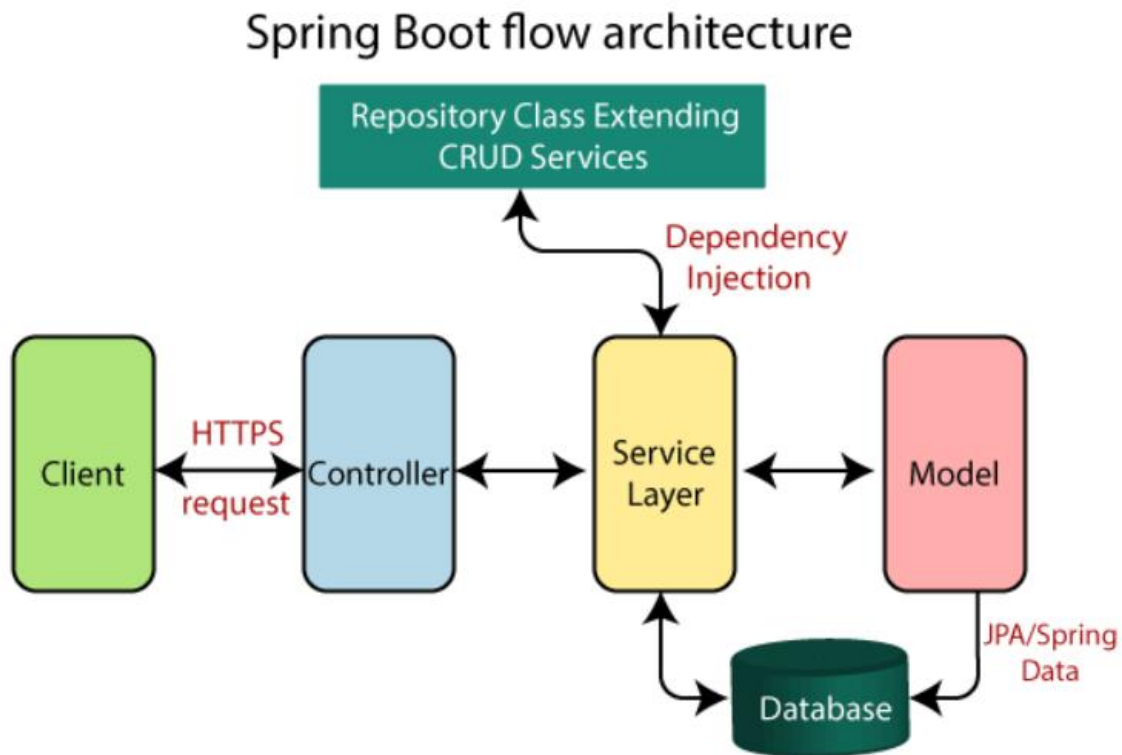
Sursă: <https://bjgp.org/content/bjgp/62/605/664.full.pdf>

Anexa 6. Diagrama de interacțiune privind modelul MVC



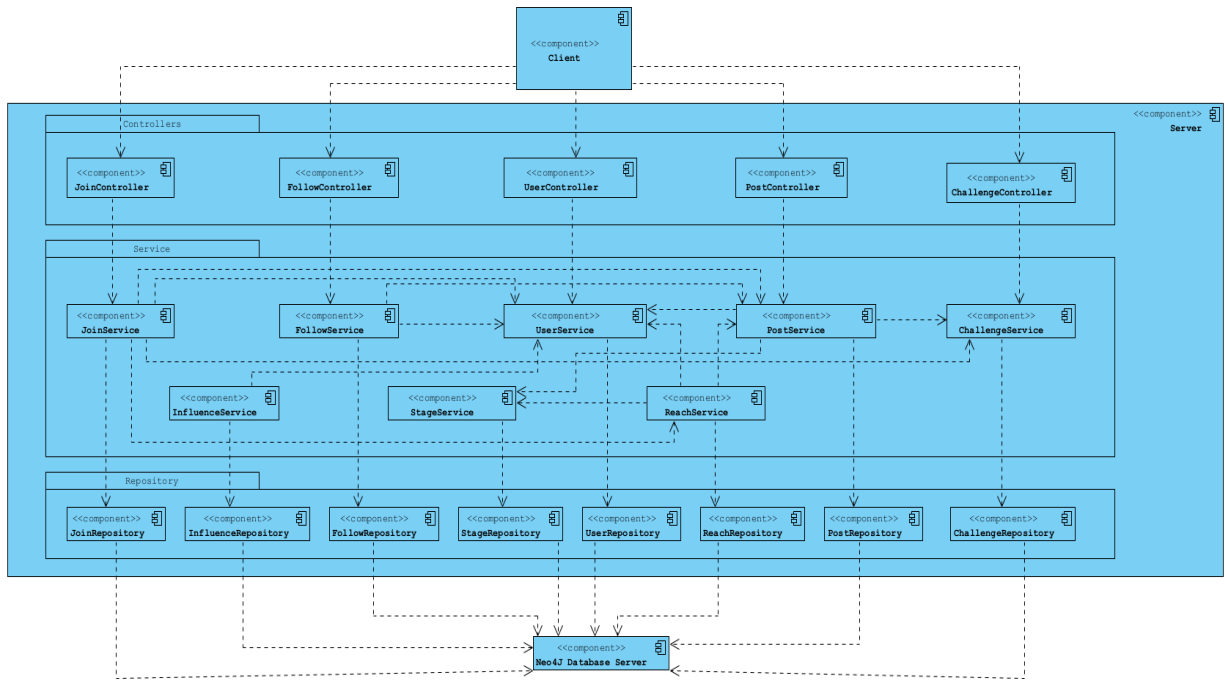
Sursă: <https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller#/media/File:MVC-Process.svg>

Anexa 7. Fluxul informațional din cadrul arhitecturii Spring Boot bazată pe cele patru straturi

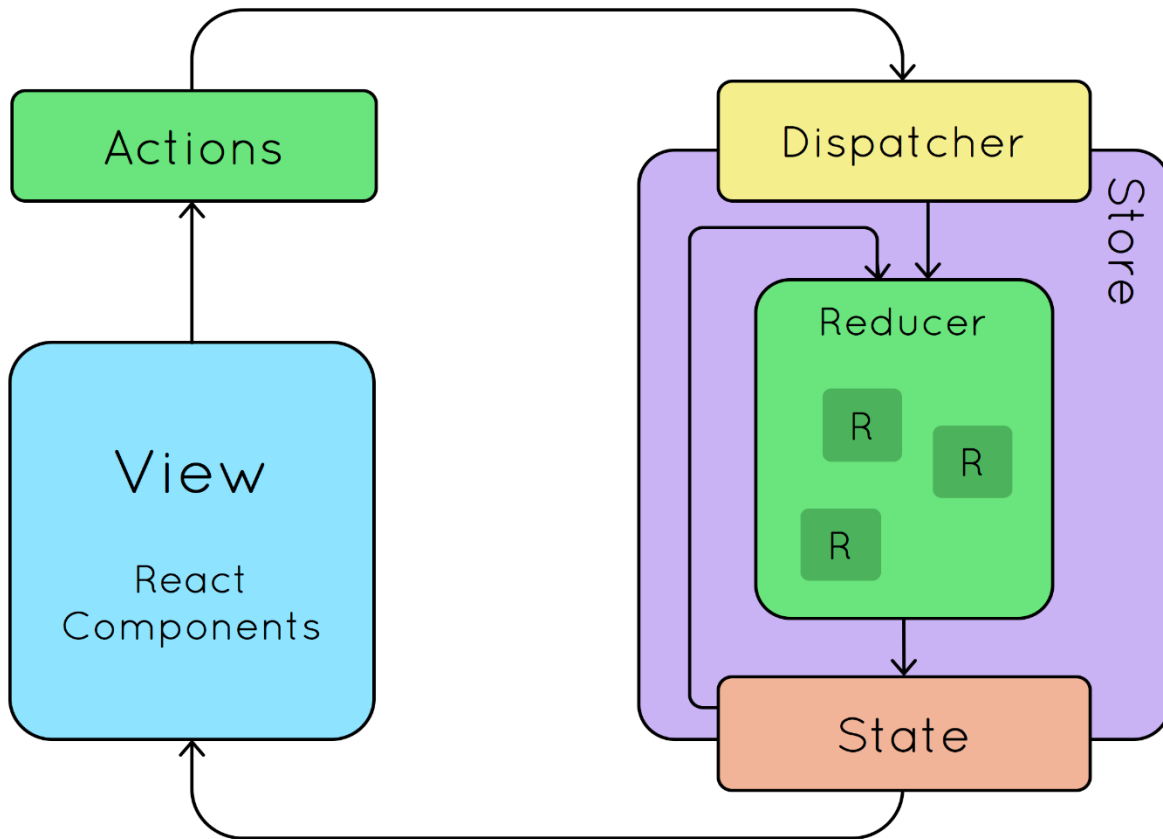


Sursă: <https://www.javatpoint.com/spring-boot-architecture>

Anexa 8. Diagrama de componente completă

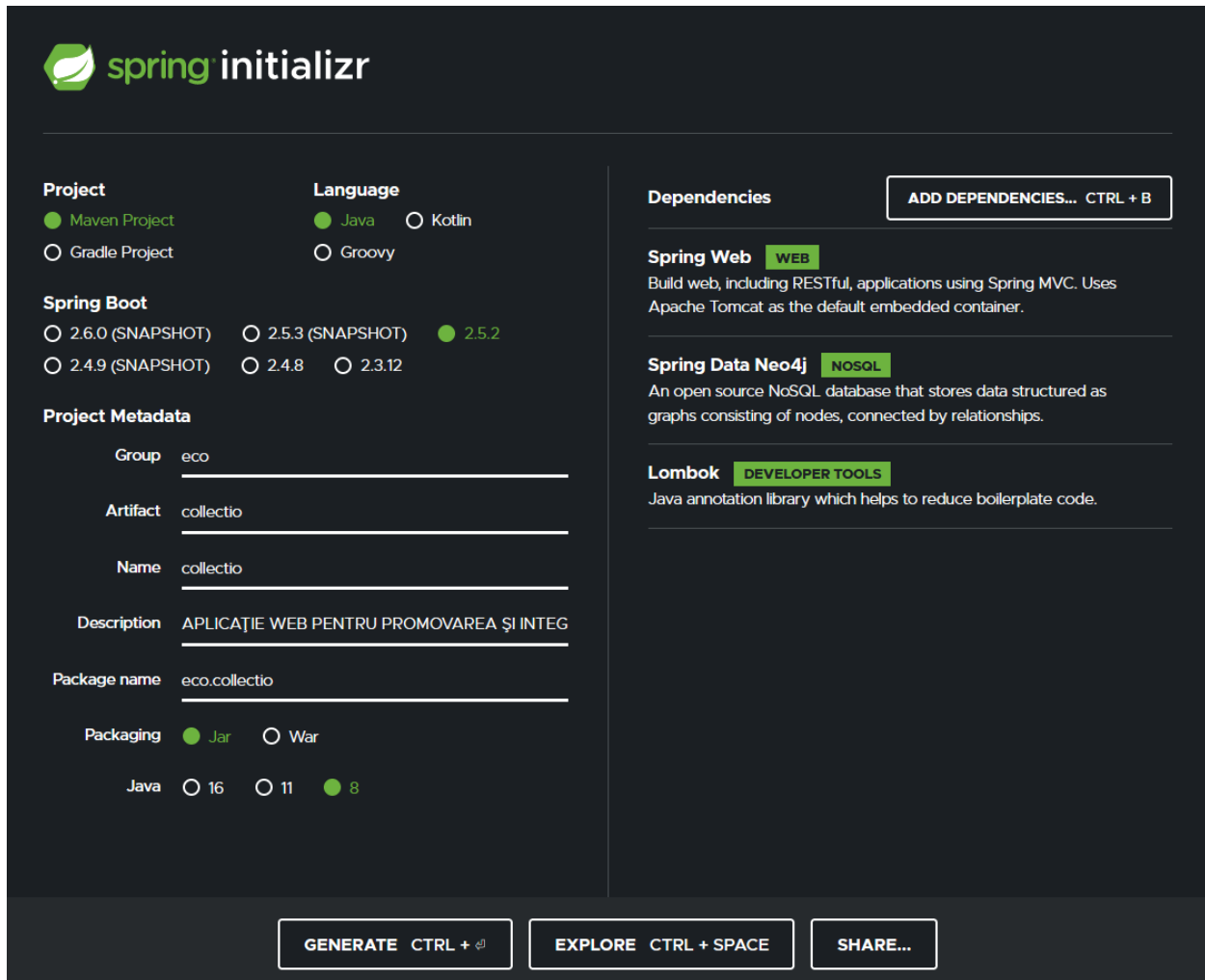


Anexa 9. Fluxul de funcționare React Redux



Sursă: <https://www.esri.com/arcgis-blog/products/3d-gis/3d-gis/react-redux-building-modern-web-apps-with-the-arcgis-js-api/>

Anexa 10. Generare proiect Spring prin Spring Initializr



The image shows the Spring Initializr web interface. It has a dark theme with green accents. The interface is divided into several sections: Project, Language, Spring Boot, Project Metadata, Dependencies, and a bottom bar with action buttons.

Project

- ☒ Maven Project
- ☐ Gradle Project

Language

- ☒ Java
- ☐ Kotlin
- ☐ Groovy

Spring Boot

- ☐ 2.6.0 (SNAPSHOT)
- ☐ 2.5.3 (SNAPSHOT)
- ☒ 2.5.2
- ☐ 2.4.9 (SNAPSHOT)
- ☐ 2.4.8
- ☐ 2.3.12

Project Metadata

Group:

Artifact:

Name:

Description:

Package name:

Packaging: ☒ Jar ☐ War

Java: ☐ 16 ☐ 11 ☒ 8

Dependencies

ADD DEPENDENCIES... CTRL + B

Spring Web **WEB**

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Spring Data Neo4j **NOSQL**

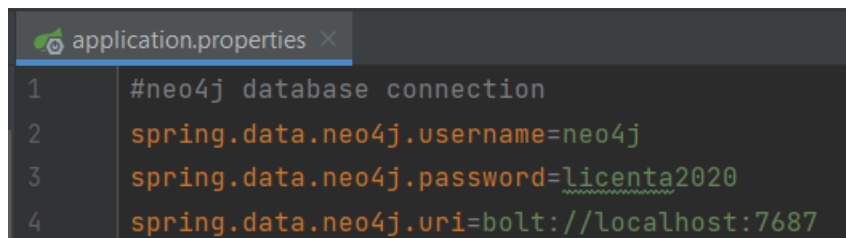
An open source NoSQL database that stores data structured as graphs consisting of nodes, connected by relationships.

Lombok **DEVELOPER TOOLS**

Java annotation library which helps to reduce boilerplate code.

GENERATE CTRL + G **EXPLORE CTRL + SPACE** **SHARE...**

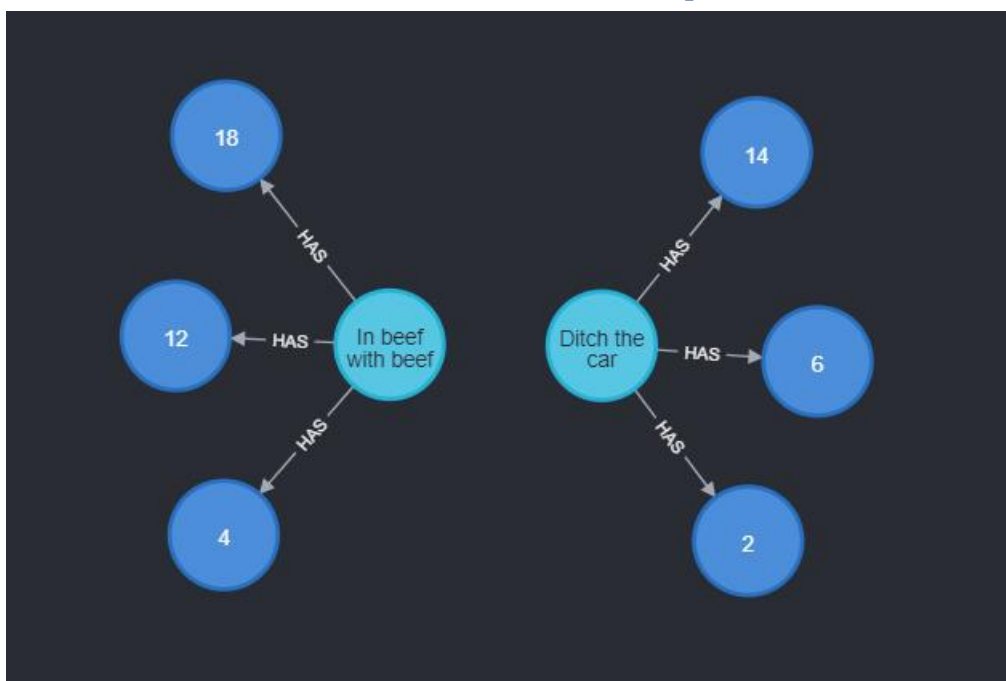
Anexa 11. Fișier configurare bază de date Neo4J



The image shows a code editor window titled "application.properties" with a close button. It contains four lines of configuration for Neo4J database connection.

```
1 #neo4j database connection
2 spring.data.neo4j.username=neo4j
3 spring.data.neo4j.password=licenta2020
4 spring.data.neo4j.uri=bolt://localhost:7687
```

Anexa 12. Rezultat vizual în Neo4J în urmă creării de provocări



Anexa 13.Design Pattern – Builder pentru postare

```
public static class PostBuilder {
    private Post post;

    public PostBuilder(PostType type, User user) {
        this.post = new Post(user);
        this.post.type = type;
    }

    public PostBuilder setChallenge(Challenge challenge) {...}

    public PostBuilder setStage(Stage stage) {...}

    public PostBuilder setFollowing(User following) {...}

    public Post build() throws InvalidPostException {
        switch (post.type) {
            case AWARD: {
                if (this.post.challengeId == null || this.post.stageId == null) {
                    throw new InvalidPostException("An AWARD type post must have a stage that was " +
                        "completed and a challenge");
                }
                return this.post;
            }
            case CHALLENGE: {
                if (this.post.challengeId == null) {
                    throw new InvalidPostException("A CHALLENGE type post must have a started challenge");
                }
                return this.post;
            }
            case FOLLOW: {
                if (this.post.followingId == null) {
                    throw new InvalidPostException("A FOLLOW type post must have a person who started following");
                }
                return this.post;
            }
            default:
                throw new InvalidPostException("There is a problem with the PostType");
        }
    }
}
```