

UNIVERSITATEA "ALEXANDRU IOAN CUZA" DIN IAȘI

FACULTATEA DE INFORMATICĂ



LUCRARE DE LICENȚĂ

Aplicație web *serverless* pentru partajarea informațiilor media

propusă de

Chiriac Dorin-Virgiliu

Sesiunea: iulie, 2019

Coordonator științific

Lect. dr. Cosmin Vârlan

UNIVERSITATEA "ALEXANDRU IOAN CUZA" DIN IAȘI

FACULTATEA DE INFORMATICĂ

Aplicație web *serverless* pentru partajarea informațiilor media

Chiriac Dorin-Virgiliu

Sesiunea: iulie, 2019

Coordonator științific

Lect. dr. Cosmin Vârlan

Avizat,

Îndrumător Lucrare de Licență

Titlu, Nume și prenume **Lect. dr. Vârlan Cosmin**

Data _____ Semnătura _____

DECLARAȚIE privind originalitatea conținutului lucrării de licență

Subsemnatul **Chiriac Dorin-Virgiliu**, domiciliat în **Jud.IS Mun.Iași Str.Costache Negruzzi nr.1 bl.1 sc.A et.1 ap.4**, născut la data de **11.07.1996**, identificat prin CNP **1960711226875**, absolvent al Universității „Alexandru Ioan Cuza” din Iași, Facultatea de **Informatică** specializarea -, promoția **2019**, declar pe propria răspundere, cunoscând consecințele falsului în declarații în sensul art. 326 din Noul Cod Penal și dispozițiile Legii Educației Naționale nr. 1/2011 art.143 al. 4 și 5 referitoare la plagiat, că lucrarea de licență cu titlul: **Aplicație web serverless pentru partajarea informațiilor media** elaborată sub îndrumarea dl. **Cosmin Vârlan**, pe care urmează să o susțină în fața comisiei este originală, îmi aparține și îmi asum conținutul său în întregime.

De asemenea, declar că sunt de acord ca lucrarea mea de licență să fie verificată prin orice modalitate legală pentru confirmarea originalității, consimțind inclusiv la introducerea conținutului său într-o bază de date în acest scop.

Am luat la cunoștință despre faptul că este interzisă comercializarea de lucrări științifice în vederea facilitării falsificării de către cumpărător a calității de autor al unei lucrări de licență, de diploma sau de disertație și în acest sens, declar pe proprie răspundere că lucrarea de față nu a fost copiată ci reprezintă rodul cercetării pe care am întreprins-o.

Data azi, _____ Semnătură student, _____

DECLARAȚIE DE CONSIMȚĂMÂNT

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

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

Iași, data _____

Absolvent **Chiriac Dorin-Virgiliu**

(semnătura în original)

CUPRINS

INTRODUCERE	1
CONTRIBUȚII	5
CAPITOLUL 1 – INTEGRAREA CU <i>FIREBASE</i>	7
CAPITOLUL 2 – PAGINA DE EXPLORARE A CONȚINUTULUI	29
CAPITOLUL 3 – PAGINA DE CANAL	38
CAPITOLUL 4 – PAGINA PANOULUI DE COMANDĂ	44
CAPITOLUL 5 – PAGINI DE CREARE DE CONȚINUT	49
CAPITOLUL 6 – ALTE PAGINI	53
CONCLUZII	57
BIBLIOGRAFIE	59

INTRODUCERE

Statistica serviciilor *cloud* a arătat că există perioade lungi de timp în care multe aplicații nu sunt utilizate, acestea blocând resurse fără să aibe nevoie de ele. Astfel, inginerii *cloud* au dezvoltat și pus la dispoziția creatorilor un nou sistem ce oferă resurse doar atunci când aplicația lor este folosită. Acest lucru permite reducerea costurilor semnificativ, fapt demonstrat de următorul exemplu: să presupunem că un *server* pune la dispoziție o aplicație ce va fi interogată o singură dată pe durata unei luni. În cazul sistemelor clasice, *serverul* trebuie să țină aplicația disponibilă pe întreaga lună pentru a servi acea unică interogare, consumând resurse nejustificat, pe când în cazul sistemelor *serverless*, *serverul* va aloca resurse pentru respectiva aplicație doar atunci când va fi interogată. Costul unei astfel de interogări devine neglijabil, iar creatorul nu mai trebuie să plătească pentru o lună întreagă de menținere a aplicației active ci doar pentru cât timp a fost folosită. Este bine totuși de menționat că în cazul traficului foarte intens, de ordinul miilor de interogări simultane constante, este mai rentabil să se folosească un sistem clasic *cloud* decât o structură *serverless*.

Problema cu care ne confruntăm în prezent este zgomotul din cadrul conținutului media cu care interacționăm zilnic. A devenit o rutină verificarea platformelor sociale în diverse perioade ale zilei, însă de cele mai multe ori, informația ce ne este oferită nu ne ajută la

dezvoltarea abilităților. Spun de cele mai multe ori, deoarece se întâmplă ca unele elemente de conținut să fie calitative și de foarte mare interes, dar sunt mult mai rare comparativ cu volumul de informație contra productivă dezvoltării noastre ce ne este oferit, iar acest lucru duce la nevoia noastră de a găsi și interacționa cu o platformă ce găzduiește conținut de calitate. Următoarele capitole vor prezenta cum a fost dezvoltată o aplicație ce încearcă să rezolve această problemă prin oferirea utilizatorului abilitatea de a posta conținut ce i se pare util și de a găsi informație ce li s-a părut altor utilizatori utilă.

Această lucrare de licență este o aplicație web de partajare a conținutului media, dezvoltată pe baza unei arhitecturi *serverless*, ce pune la dispoziția utilizatorului abilitatea de a posta dar și de a găsi conținut ce îi poate fi util în diverse contexte. Din punct de vedere al asemănării cu alte aplicații, această lucrare folosește ideea de canal de pe platforma *youtube.com*, cât și modul de afișare a noului conținut în stilul platformei *9gag.com* ce se bazează pe o ierarhie de tip post nou, în vogă și foarte în vogă, cât și a unei categorii. Originalitatea sa este dată de tematică, și anume cerința ca informația fiecărui post să poată fi utilizată într-un anumit context. Obținerea utilității conținutului este principalul obiectiv al acestei platforme.

Pentru aceasta, aplicația pune la dispoziția utilizatorului următoarele pagini:

- Pagina de explorare a conținutului

Pagina oferă posibilitatea de a vedea postări ale diferiților utilizatori ordonate pe baza a mai multor principii, principii detaliate în CAPITOLUL 2 al acestei documentații, iar fiecare postare oferă informații relevante despre conținutul făcut disponibil, cât și un set de acțiuni ce le implică, precum aprecierea, adăugarea acesteia în lista preferențială a utilizatorului autentificat, partajarea ei pe diferite platforme sociale, abilitatea de a fi raportată dacă un utilizator consideră că aceasta încalcă regulile platformei.

- Pagina de canal

Aici este prezentată identitatea unui mediu ce adaugă conținut aplicației, cât și întreg conținutul postat de către acesta. Această pagină are 5 componente, componenta de postări, componenta de colecții, componenta de recomandări, componenta de postare individuală, cât și componenta de mai multe informații despre canal. Acestea vor fi prezentate mai în detaliu în capitolele ce urmează. Utilizatorii au abilitatea de a se abona acestui mediu și vor putea urmări activitatea lui din componenta noutăți a paginii panoului de comandă personal.

- Pagina panoului de comandă personal

Această pagină pune la dispoziția utilizatorului abilitatea de a urmări conținutul canalelor urmărite cu ajutorul componentei de noutăți, să vizualizeze postările pe care le consideră importante în cadrul componentei listei utile, să fie înștiințat

când cineva interacționează cu resursele puse la dispoziție de către el, fie o postare, un comentariu sau un canal, în cadrul componentei de notificări, abilitatea de a crea noi canale sau de a le gestiona pe cele existente din componenta canalele mele, cât și modificarea informațiilor contului din componenta setări.

- Pagina de creare de conținut

Această pagină servește la crearea noului conținut. Conținutul poate fi fie o pagină de canal, fie o nouă colecție din componenta acelui canal, fie o nouă postare.

- Pagina de căutare de conținut

De aici utilizatorul are abilitatea de a găsi fie un canal, o colecție sau o postare pe baza unui sir de caractere ce trebuie să se afle la începutul secvenței de căutare specific fiecăreia dintre cele 3 menționate.

- Pagina de informații legale

Aceasta pune la dispoziția utilizatorului 3 componente. O componentă ce face referire la ce fel de conținut nu este permis să fie postat pe această platformă, o a doua menționează termenii la care utilizatorii se supun prin folosirea platformei, iar o a treia specifică modul de procesare al datelor personale.

CONTRIBUȚII

Când vine vorba de dezvoltarea unei aplicații web pe baza unei arhitecturi *serverless*, trei mari nume se aud cel mai des în rândul programatorilor și anume *Amazon AWS*, *Google Cloud Platform* și *Microsoft Azure*. Din punct de vedere al prețurilor, cele trei platforme sunt asemănătoare, însă *Amazon AWS* este mai ieftin cu aproximativ 0.003\$ la fiecare serviciu utilizat. Cu toate acestea, am folosit pentru dezvoltarea lucrării de licență produsul *Firebase*, dezvoltat de către *Google*, deoarece a fost conceput specific pentru a veni în ajutorul aplicațiilor *serverless*, iar *Amazon AWS* are nevoie de servicii suplimentare pentru a obține funcționalitățile necesare creării arhitecturii, și anume *API Gateway* și *Identity and Access Management (IAM)*, ce aduc costuri suplimentare semnificative.

În cazul sistemelor *cloud* clasice, pentru a-și face disponibile aplicațiile web, creatorii trebuie să folosească un program *server* ce să accepte interogări la portul 80 pe care mai apoi să le transmită platformei dezvoltate, platformă ce de cele mai multe ori respectă modelul arhitectural *Model-view-controller (MVC)*. Comparativ cu această structură în care toate interogările sunt transmise în același loc pentru a fi procesate, structurile *serverless* folosesc servicii predefinite pentru fiecare operațiune necesară. În cazul platformei *Firebase*, pentru a obține fișierele statice cu extensii precum *.html*, *.css*, *.js*, este recomandat să se folosească *Firebase*

Hosting, pentru serviciul de autentificare, *Firebase Authentication*. În cazul în care este nevoie de o bază de date, *Firebase* pune la dispoziție una de tip nerelațională (NoSQL), aflată sub serviciul *Firebase Database*. Pentru stocare de fișiere de tip imagine, video, pdf, etc., se poate utiliza serviciul *Firebase Storage*. În cazul sistemelor *serverless*, logica de procesare a datelor se află la nivelul clientului de cele mai multe ori, dar sunt anumite scenarii precum adăugare și editare de conținut aflat în baza de date, ce nu ar trebui să aibă loc la nivelul clientului, ci la nivelul *serverului*, pentru o mai mare integritate a operațiunii. În acest scop, *Firebase* introduce conceptul de funcție aflat sub serviciul *Firebase Functions*. O aplicație poate avea mai multe funcții, fiecare rulând atunci când adresa la care au fost definite este apelată.

Această lucrare de licență folosește serviciile menționate anterior, *Firebase Hosting*, pentru livrarea fișierelor statice, *Firebase Authentication*, pentru autentificarea utilizatorilor pe baza unei adrese de email și a unei parole, sau pe baza unui cont deja existent la o parte terță precum *Facebook*, *Twitter* sau *Google+*, *Firebase Database*, pentru salvarea postărilor, canalelor, notificărilor, comentariilor, *Firebase Storage*, pentru stocarea imaginilor, *Firebase Functions*, pentru operațiuni de creare, editare, ștergere a unui post, adăugare a unui comentariu, apreciere a unui post sau comentariu. Capitolele următoare descriu cum a fost folosit fiecare serviciu pentru a face disponibile diverse operațiuni către utilizator.

CAPITOLUL 1 – Integrarea cu *Firebase*

Pentru a realiza integrarea serviciului *Firebase* cu aplicația web, dezvoltatorul trebuie să își creeze un cont *Google* prin intermediul căruia să se autentifice pe platforma web a serviciului, iar odată ce este autentificat pe aceasta, trebuie să creeze un nou proiect ce va găzdui datele aplicației dezvoltate corespunzătoare fiecărei componente folosite. În cazul în care creatorul nu are un domeniu ce să directeze către *serverul* unde sunt găzduite fișierele statice ale proiectului său, *Firebase* oferă fiecărui proiect o adresă de referință sub forma: `<id_proiect>.firebaseapp.com`. Pentru a face aplicația disponibilă la adresa menționată, trebuie să se instaleze pe mașina de dezvoltare programul *Firebase CLI* (*Firebase Command Line Interfac*) cu ajutorul căruia să se încarce fișierele proiectului în serviciul *Firebase Hosting*.

Odată ce *Firebase CLI* a fost instalat, se crează pe mașina de dezvoltare un director nou cu denumirea noului proiect, din acesta se deschide un terminal și se rulează comanda `firebase init`. Această comandă va porni o secvență de interogări pentru setarea mediului de lucru, printre care se vor cere datele de autentificare ale contului *Google*, menționarea proiectului *Firebase* (cel creat anterior), cât și specificarea serviciilor *Firebase* utilizate în cadrul acestui proiect. Pentru rularea locală a proiectului se va executa comanda `firebase serve` într-un terminal deschis din directorul sursă al





acestui, ce va face aplicația disponibilă la adresa *localhost:5000*, iar pentru a face platforma disponibilă la adresa menționată anterior, se va rula comanda `firebase deploy`, instrucțiune ce va urca fișierele locale pe *serverele Google*. În cazul în care se dorește menținerea fișierelor pe un *server* personal, este posibilitatea de a face serviciile *Firebase* disponibile cu ajutorul unei variabile javascript de configurare și prin adăugarea unor adrese în antetul fișierului `index.html` pentru fiecare serviciu *Firebase* utilizat. Variabila de configurare conține următoarele câmpuri:

```
var firebaseConfig = {  
  apiKey: "api-key",  
  authDomain: "project-id.firebaseio.com",  
  databaseURL: "https://project-id.firebaseio.com",  
  projectId: "project-id",  
  storageBucket: "project-id.appspot.com",  
  messagingSenderId: "sender-id",  
  appId: "app-id",  
};
```

Această lucrare de licență folosește serviciul *Firebase Hosting* pentru a face aplicația disponibilă la următoarea adresă: *usefulindeed-c3831.firebaseio.com*. De asemenea, aplicația folosește la nivelul clientului o arhitectură de tip *Single Page Web Application* ce permite o navigare mai rapidă între pagini decât varianta clasică în care se trimite o interogare la *server* pentru fiecare pagină accesată. Un alt avantaj al acestei abordări este limitarea codului duplicat, având posibilitatea refolosirii anumitor secvențe de cod în pagini diferite, nefiind nevoie să fie rescrise, iar în cazul unei modificări, aceasta se realizează într-un singur loc. Un exemplu în acest sens este bara de meniu a aplicație.

În cazul aplicațiilor web complexe, de cele mai multe ori este nevoie de autentificarea utilizatorului pentru a-i restricționa accesul la anumite date, pentru a-i oferi dreptul de a face noi acțiuni sau de a ține evidența activității acestuia de pe *site*. În acest sens, *Firebase* pune la dispoziția programatorului serviciul *Firebase Authentication*, serviciu ce este folosit mai departe de celelalte componente, *Firebase Database*, *Firebase Storage*, *Firebase Functions*, etc, pentru a stabili dacă instrucțiunile primite sunt în concordanță cu drepturile utilizatorului ce le-a trimis.

Această lucrare de licență folosește serviciul *Firebase Authentication* și permite autentificarea utilizatorului în două forme, cu ajutorul unui cont creat pe baza unei adrese de email cât și a unei parole, sau folosind serviciul de autentificare a unei terțe părți. Aplicația folosește serviciile de autentificare a platformelor *Facebook*, *Twitter* și *Google+*. Pentru a face disponibile cele 2 moduri de autentificare, dezvoltatorul trebuie să le activeze individual, iar în cazul autentificării pe baza unui cont aflat la o altă platformă, trebuie să se urmeze un șir de configurări specific fiecăreia dintre ele.

Provider	Status
 Email/Password	Enabled
 Phone	Disabled
 Google	Enabled
 Play Games	Disabled

În cazul platformelor *Facebook* și *Twitter*, este necesar să se creeze un cont, să se acceseze consola de dezvoltator pusă la dispoziție

de fiecare din cele două, să se configureze o nouă aplicație din fiecare, să se acceseze pagina de setări a fiecărei aplicații nou create, iar datele afișate în respectivele date trebuie copiate în formularul pus la dispoziție de către *Firebase Authentication*.

Serviciul *Firebase Authentication* restricționează abilitatea de autentificare a utilizatorilor din aplicații ce se află la alte adrese decât cele menționate în câmpul *Authorized domains*.

Authorized domain	Type
localhost	Default
usefulindeed-c3831.firebaseio.com	Default
www.useofthis.com	Custom

Acest lucru ajută la securizarea datelor, eliminând abilitatea altor platforme de a folosi resursele proiectului fără acceptul dezvoltatorului.

Firebase Database este un serviciu ce pune la dispoziția dezvoltatorului o bază de date nerelațională (NoSQL) ce salvează datele într-o structură *JSON* [1] în care fiecare nod ajută la ierarhizarea valorilor salvate în frunze. Astfel, pentru a obține valorile salvate, trebuie specificată secvența de noduri până la nodul ce se dorește a fi obținut.

Pentru securizarea datelor, *Firebase* introduce noțiunea de reguli la nivel de nod ce permit restricționarea accesului pentru utilizatorii ce nu le respectă, acestea propagându-se și la nodurile copil. Ca și exemplu de reguli, acestea pot verifica dacă utilizatorul curent este autentificat cu ajutorul serviciului *Firebase Authentication*, dacă nodul

pe care dorește să-l acceseze a fost creat de el sau nu, dacă dimensiunea datelor introduse respectă o lungime maximă, și multe altele.

Această lucrare de licență are următoarea structură a bazei de date:

```
usefulindeed-c3831
  categories
    [element_id] : [value]
  ...
  feedback
    [element_id]
    text: [value]
    uid: [value]
  ...
  private
    [uid]
    likes
      [post_id]
      state: [value]
      type: [value]
    ...
    subscribed
      [channel_id]
      state: [value]
    ...
    usefullist
      [element_id]
      id: [value]
    ...
    notifications
      [element_id]
      for: [value]
      time: [value]
      type: [value]
    ...
    user
      channels
        [channel_id]
        coverPic: [value]
        id: [value]
        name: [value]
        profilePic: [value]
      ...
      settings
        email: [value]
        nsfw: [value]
    ...
```

```

public
  channels
    [channel_id]
      identity
        coverPic: [value]
        id: [value]
        name: [value]
        profilePic: [value]
      uid: [value]
      values
        likes: [value]
        subscribers: [value]

    ...

  collections
    [element_id]
      channel: [value]
      content
        description: [value]
        name: [value]
      uid: [value]
      values:
        nrPosts: [value]
        thumbnail: [value]

    ...

  comments
    [comment_id]
      followers:
        uids: [value...]
      for: [value]
      from: [value]
      text: [value]
      type: [value]
      uid: [values]
      values
        comment_hot: [value]
        comment_new: [value]
        likes: [value]
        replays: [value]

    ...

  posts
    [post_id]
      channel: [value]
      content
        category: [value]
        collection: [value]
        nsfw: [value]
        source: [value]
        title: [value]
        titleSearch: [value]
        type: [value]
        usability: [value]

```

```

        ref
            articleTitle: [value]
            articleURL: [value]
            imageURL: [value]
            type: [value]
        uid: [value]
        values
            category_hot: [value]
            category_new: [value]
            channel_hot: [value]
            channel_new: [value]
            collection: [value]
            comments: [value]
            likes: [value]
            rating: [value]
            type_hot: [value]
            type_new: [value]
        ...
    report
        [element_id]
        email: [value]
        id: [value]
        reasons: [value]
        type: [value]
        uid: [value]
    ...

```

Din punct de vedere a regulilor de accesare și adăugare de conținut, acestea sunt un obiect *JSON* în care se specifică o instrucțiune de validare pentru operațiunea de citire, ștergere, cât și posibilitatea de a valida datele introduse, în scopul de a nu fi adăugate câmpuri nedorite, sau alte valori decât cele așteptate. Acest arbore respectă structura definită în serviciul *Firebase Database*. Ele sunt reprezentate astfel:

```

1  {
2    "rules": {
3      "report": {
4        "$id": {
5          ".read": false,
6          ".write": "auth != null && newData.child('uid').val() == auth.uid && !data.exists()",
7          ".validate": "newData.hasChildren(['id', 'reason', 'type']) && $id.length <= 100",
8        }
9        "$child": {
10         ".validate": "($child == 'id' || $child == 'reason' || $child == 'type' || $child == 'uid' ||
11         $child == 'email') && newData.isString() && newData.val().length <= 100"
12       }
13     },
14     "feedback": {
15       "$id": {
16         ".read": false,
17         ".write": "!data.exists() && newData.exists()",
18         ".validate": "$id.length <= 100",
19       }
20     }
21   }

```

Regulile serviciului *Firebase Database* au mai multe primitive ce reprezintă referința către ce tip de acțiune trebuie să se aplice respectivele reguli. Printre ele se numără:

- primitiva *.read* - reguli de citire a nodului cât și a copiilor săi
- primitiva *.write* - reguli de scriere la nivelul nodului
- primitiva *.validate* - pentru validarea datelor introduse

Baza de date poate fi interogată în două moduri, fie prin specificarea unei căi exacte, fie prin filtrarea nodurilor rezultate pe baza numelui lor, fie pe baza valorii unei frunze din componența sa. Pentru a putea face o filtrare a nodurilor copil de la nivelul unei căi din baza de date, trebuie să specificăm în primitiva *.indexOn* toate căile care servesc la filtrarea datelor cerute, acțiune realizată la nivelul clientului. În cazul filtrării la nivelul nodului de postări, avem următorii indecși:

```

"posts": {
  ".indexOn": ["channel", "content/collection", "values/collection",
    "content/titleSearch", "values/category_new",
    "values/category_hot", "values/rating", "values/type_new",
    "values/type_hot", "values/channel_new", "values/channel_hot"],
  [post_id]
  ...
}

```


Această lucrare de licență are definite reguli pentru următoarele căi din baza de date:

- `report/[id]`

Acest nod servește la salvarea cerințelor utilizatorilor de a șterge un conținut deoarece consideră că încalcă regulile platformei. Acest nod nu poate fi citit, dar pentru a se scrie la nivelul său, utilizatorul trebuie să fie autentificat iar data introdusă trebuie să conțină un câmp `uid` la care să se afle valoare `id`-ului de utilizator personal. Pentru ca data introdusă să fie validă, aceasta trebuie să conțină nodurile copil `id`, `reason` și `type`.

- `report/[id]/[child]`

Acest nod permite ca și noduri frunză doar pe cele cu denumirea `id`, `reason`, `type`, `uid` și `email`.

- `feedback/[id]`

Acest nod servește la salvarea recomandărilor utilizatorilor pentru îmbunătățirea platformei. Acest nod nu permite citirea, însă permite ca oricine să poată adăuga conținut, indiferent dacă este autentificat sau nu. Câmpul `text`, care conține mesajul utilizatorului, trebuie să fie de tip șir de caractere și să aibă o lungime maximă de 4000 de caractere.

- `categories`

Acest câmp conține toate categoriile ce pot fi atribuite unui post. El permite citirea de către orice utilizator, dar nu permite scrierea.

- `public`

Aici sunt salvate toate datele care pot fi citite de către orice utilizator, indiferent dacă este autentificat sau nu, însă regulile de scriere sunt definite pentru fiecare nod copil în parte.

- `public/channels/[id]`

Acest nod conține datele tuturor canalelor create în cadrul platformei. Fiecare nod are în componența sa alte noduri ce fac referire la identitatea canalului, câmpuri ce presupun numele, referința către imaginea de profil, referința către imaginea de copertă, cât și informațiile despre acesta, și anume descrierea sa, datele de contact, adrese făcute disponibile, cât și nodul de valori ce conține numărul de aprecieri, cât și numărul de abonați.

În acest nod poate scrie doar utilizatorul care a creat canalul, numele canalului trebuie să fie unic în întreaga aplicație și pot fi introduse valori în câmpurile specificate anterior doar dacă se respectă formatul datelor. Indiferent dacă utilizatorul este autentificat sau canalul îi aparține, nimeni nu poate modifica valorile din nodul `values`, numărul de aprecieri cât și numărul de abonați. Ele sunt modificate de o funcție din cadrul serviciului *Firebase Functions*, serviciu descris în continuarea acestui capitol.

- `public/collections/[id]`

Acest nod cuprinde referințe către postări ale aceluiași utilizator ce l-a creat cu scopul de a le grupa sub o singură tematică. Un utilizator poate scrie în acest nod doar dacă este autentificat iar câmpul obligatoriu ce specifică canalul ce deține

această colecție aparține aceluiași utilizator. De asemenea, utilizatorul nu poate modifica valoarea numărului de postări referențiate în cadrul colecției, aceasta este disponibilă doar funcțiilor din cadrul serviciului *Firebase Functions*.

- `public/posts/[id]`

Acest nod cuprinde datele tuturor postărilor platformei. Un nod de tip post poate fi adăugat doar de către un utilizator autentificat, în cadrul unui canal ce aparține respectivului utilizator, într-o colecție care aparține aceluși canal. Câmpurile ce indică numărul de aprecieri, numărul de comentarii, cât și canalul din care face parte nu pot fi modificate ulterior decât de o funcție din cadrul serviciului *Firebase Functions*.

- `public/comments/[id]`

Aici sunt salvate toate comentariile postărilor, inclusiv comentariile de tip răspuns. Acestea pot fi adăugate de orice utilizator autentificat, de asemenea pot fi modificate, dar doar de către utilizatorul ce le-a creat.

- `private/[uid]`

Acest nod și respectiv toți copiii acestuia sunt accesibili doar utilizatorului autentificat al cărui id de utilizator corespunde numelui nodului. Prin urmare, operațiile de citire și scriere sunt permise doar utilizatorului ce deține nodul, existând totuși reguli de validare la nivelul nodurilor copil.

- `private/[uid]/user/channels`

Acest nod conține referințe către canalele deținute de către utilizator. Este permisă adăugarea de noi canale doar dacă numele acestora este unic în cadrul platformei.

- `private/[uid]/notifications/[id]`

Aici sunt salvate toate notificările utilizatorului, fie ele despre o nouă apreciere la o postare sau comentariu, o nouă abonare, fie despre un răspuns la un comentariu. Utilizatorul are dreptul doar să modifice valoarea câmpului văzut, pentru a putea fi diferențiate notificările noi de cele deja analizate. Notificările sunt introduse în funcție de operație de către funcțiile din serviciul *Firebase Functions*.

- `private/[uid]/stopNotifications/[id]`

Utilizatorul are abilitatea de a opri notificările de la toate resursele ce îl înștiințează. Înainte de a adăuga o notificare în câmpul descris anterior, funcțiile din serviciul *Firebase Functions* verifică acest câmp pentru a vedea dacă resursa modificată poate sau nu să ofere o notificare utilizatorului abonat la ea.

- `private/[uid]/likes/[id]`

În cazul în care utilizatorul apreciază un conținut, fie el o postare sau un comentariu, este necesar să i se afișeze acest lucru în cazul în care revine la acea resursă. Pentru a verifica dacă utilizatorul a apreciat o resursă, se verifică la nivelul clientului acest câmp. Câmpul poate fi modificat doar de către utilizatorul ce deține acest nod.

- `private/[uid]/subscribed/[id]`

La fel ca și în cazul unei aprecieri, utilizatorul este înștiințat în aceeași manieră dacă este sau nu abonat la un anumit canal. Deasemena, câmpurile acestui nod pot fi modificate doar de către utilizatorul ce le deține.

Aplicația permite unui utilizator să creeze canale în stilul celor de pe platforma *youtube.com*, ce oferă abilitatea de a posta conținut. Un canal poate fi personalizat prin setarea unei imagini de copertă cât și a unei imagini de profil, iar o postare creată poate fi setată să prezinte o imagine reprezentativă informației oferite. Toate fișierele de tip imagine folosite în cadrul platformei pentru personalizarea canalelor și postărilor sunt stocate în cadrul serviciului *Firebase Storage* în directoarele corespunzătoare fiecărui utilizator.

Din punct de vedere al securității, se urmează același principiu ca și la serviciul *Firebase Database*, securitatea fiind asigurată la nivel de reguli de acces pe directoarele utilizatorilor. Din punct de vedere al regulilor pentru stocarea de fișiere, complexitatea pentru această aplicație este mult mai mică comparativ cu cea a bazei de date. Regulile platformei pentru baza de date au 215 linii, dar regulile corespunzătoare fișierelor stocate au doar 14 linii. Aceste reguli oferă oricărui utilizator autentificat sau neautentificat dreptul de a citi orice fișier din directorul sursă a proiectului, însă nu permite niciunui utilizator să scrie informație în directorul `/defaults`, dar le este permis să introducă fișiere de tip imagine, spre exemplu cele cu

extensii precum .jpg, .png, .gif, etc., cu o dimensiune mai mică de 1MB, dacă sunt autentificați și doresc să introducă respectiva imagine în directorul cu denumirea identificatorului contului de utilizator, aflat la calea /userPictures. Regulile sunt prezentate în următoarea imagine:







```
1  service firebase.storage {
2    match /b/{bucket}/o {
3      match /defaults/{id} {
4        allow read: if true
5        allow write: if false
6      }
7      match /usersPictures/{$uid}/{id} {
8        allow read: if true
9        allow write: if request.auth != null && request.auth.uid == $uid &&
10           request.resource.contentType[0:6] == 'image/' &&
11           request.resource.size < 1024 * 1024
12      }
13    }
14  }
```

Din punct de vedere al integrității manipulării datelor, instrucțiunile de adăugare de conținut, editare de conținut propriu, vizualizare a conținutului postat pe platformă, pot fi realizate la nivelul clientului, deoarece regulile serviciilor *Firebase Database* și *Firebase Storage* sunt deajuns pentru a o asigura. În cazul în care un utilizator adaugă un comentariu la postarea unui alt utilizator, apasă pe butonul de apreciere a unui post sau se abonează la un canal, acțiuni ce necesită modificarea datelor altor utilizatori, lucru ce din motive de securitate și menținere a integrității datelor nu ar trebui să se întâmple, se va folosi serviciul *Firebase Functions*.

Serviciul *Firebase Functions* pune la dispoziția dezvoltatorului abilitatea de a rula un set de instrucțiuni la nivelul *serverului* fără să

fie nevoie să se țină active resurse dedicate mereu, acestea alocându-se fiecărei funcții doar atunci când un declanșator le cere să ruleze. Declanșatorul poate fi ori un eveniment dintr-un anumit serviciu *Firebase*, spre exemplu o adăugare, modificare, ștergere de date din *Firebase Database* cât și a unui fișier din *Firebase Storage*, dar și o apelare de tip REST a unei adrese predefinite a aplicației.

Pentru a crea o funcție este nevoie să fie dezvoltată pe mașina locală și apoi să fie urcată pe *serverele Google* prin intermediul programului *Firebase CLI*.

deletePost	 ref.delete public/posts/{id}	us-central1	Node.js 8	256 MB	60s
like	 ref.create private/{uid}/likes/{id}	us-central1	Node.js 8	256 MB	60s
likeUnlike	 ref.update private/{uid}/likes/{id}	us-central1	Node.js 8	256 MB	60s
share	 HTTP Request https://us-central1-usefulindeed-c3831.cloudfunctions.net/share	us-central1	Node.js 8	256 MB	60s
subscribe	 ref.create private/{uid}/subscribed/{child}	us-central1	Node.js 8	256 MB	60s
subscribeUnsubscr...	 ref.update	us-central1	Node.js 8	256 MB	60s

În definirea unei funcții, trebuie să specificăm anumiți parametri de configurare. În primul rând, trebuie să oferim un nume acelei funcții, unic la nivelul aplicației. De asemenea, trebuie să specificăm un declanșator al ei. Un declanșator poate fi o modificare dintr-un anumit serviciu, spre exemplu o adăugare de fișier în serviciul *Firebase Storage*, operații de modificare a unui anumit nod din serviciul *Firebase Database*, dar și o apelare de tip REST la o anumită adresă a aplicației poate porni o funcție ce se va comporta ca o interogare clasică a unui *server cloud*, acceptând toate verbele serviciului REST, oferind un răspuns prin același protocol HTTPS. De

asemenea, se specifică regiunea serverului de pe care să ruleze funcția, mediul de rulare, în aceste cazuri, Node.js 8, cât și memoria alocată acesteia. Se pot alocă pentru rularea unei funcții maxim 2048MB. Toate funcțiile au și o durată maximă de execuție, limita standard fiind 60 de secunde, dar poate fi modificată ulterior, aceste limitări fiind o măsură de siguranță împotriva erorilor netratate sau buclelor infinite accidentale.

Această lucrare de licență folosește serviciul *Firebase Functions* și are definite următoarele funcții:

- funcția de partajare de conținut

Această funcție se execută prin apelarea adresei `https://us-central1-usefulindeed-c3831.cloudfunctions.net/share/[post_id]` și are scopul de returna o pagină html cu componente de *metadata* precompletate specifice resursei accesate, capabilă să fie procesată de rețelele de socializare *Facebook*, *Twitter*, sau orice mediile ce folosesc *Open Graph Protocol* pentru afișarea conținutului în formă mimificata. Pentru asta, se interoghează baza de date pentru postul cerut, după care se completează corespunzător câmpurile *metadatelor* cu numele `twitter:card`, `twitter:title`, `twitter:description`, `twitter:image`, `og:type`, `og:url`, `og:title`, `og:description`, `og:image`.

- funcția de creare a unui post

Această funcție se execută atunci când se crează conținut nou la calea `/public/posts/{postId}/content/collection` în baza de date a serviciului *Firebase Database* și va adăuga

imaginea postării ca și imagine de prezentare colecției din care face parte, cât și să incrementeze cu 1 numărul de postări din cadrul acesteia.

- funcția de modificare a filtrelor unui post

Această funcție se execută atunci când se modifică valoarea de la calea `/public/posts/{postId}/content` în baza de date a serviciului *Firestore Database* și va modifica valorile câmpurilor ce ajută la filtrarea datelor în funcție de numărul de aprecieri ale postării.

- funcția de modificare a unei colecții

Această funcție se execută atunci când se modifică valoarea de la calea `/public/posts/{postId}/content/collection` în baza de date a serviciului *Firestore Database*, ce presupune o modificare a colecției din care face parte un post și va modifica valorile colecției din care făcea parte prin scăderea numărului de postări din interiorul ei, cât și modificarea imaginii de prezentare cu imaginea corespunzătoare ultimului post din ea, iar în legătură cu colecția în care postarea a fost mutată, numărul de postări din interiorul ei va fi incrementat cu 1, iar imaginea de prezentare va fi modificat cu cea a noului post.

- funcția de abonare la un canal

Această funcție se execută atunci când se adaugă o valoare la calea `/private/{uid}/subscribed/{chId}` în baza de date a serviciului *Firestore Database* și va incrementa numărul de

abonați a canalului la care s-a abonat utilizatorul și va crea o nouă notificare la nivelul utilizatorului care deține respectivul canal, în cazul în care acesta nu a optat să nu mai primească notificări în legătură cu acesta.

- funcție de modificare a abonărilor unui canal

Această funcție se execută atunci când se modifică valoarea de la calea `/private/{uid}/subscribed/{chId}/state` în baza de date a serviciului *Firestore Database*, lucru ce presupune o abonare sau dezabonare de la un canal, dar utilizatorul a fost deja abonat la acesta. Comparativ cu funcția precedentă, utilizatorul ce deține acel canal nu este înștiințat de fiecare dată când aceeași abonare se întâmplă iar și iar, evitând notificările de tip spam.

- funcția de apreciere a unui conținut

Această funcție se execută atunci când se adaugă o valoare la calea `/private/{uid}/likes/{id}` în baza de date a serviciului *Firestore Database* și va incrementa numărul de aprecieri a unei postări sau comentariu, va modifica corespunzător filtrele pe baza noului număr de aprecieri și va crea o nouă notificare la nivelul utilizatorului care deține respectiva postare, în cazul în care acesta nu a optat să nu mai primească notificări în legătură cu aceasta.

- funcția de modificare a unei aprecieri de conținut

Această funcție se execută atunci când se modifică valoarea de la calea `/private/{uid}/likes/{id}` în baza de date a serviciului *Firestore Database*, lucru ce presupune o apreciere sau dezapreciere a unui post sau comentariu, dar utilizatorul a mai apreciat deja resursa. Comparativ cu funcția precedentă, utilizatorul ce deține acel post sau comentariu nu este înștiințat de fiecare dată când aceeași are loc iar și iar, evitând notificările de tip spam.

- funcția de creare de comentariu

Această funcție se execută atunci când se adaugă o valoare la calea `/public/comments/{id}` în baza de date a serviciului *Firestore Database*, având două tipuri de comentarii, fiecare tratate individual. În cazul în care se adaugă un comentariu la o postare, ce incrementează numărul de comentarii ale acelei postări și se înștiințează printr-o notificare utilizatorul ce a creat postarea, în cazul în care nu a optat pentru opțiunea de a nu primi notificări de la acea resursă. În cazul în care comentariul adăugat este de tip răspuns la un comentariu deja existent, se va incrementa numărul de răspunsuri al comentariului și va fi înștiințată toată lista de utilizatori ce au adăugat răspunsuri la acel comentariu, în cazul în care nu au optat să fie șterși din aceasta, lista la care se poate adăuga orice utilizator autentificat ce dorește să urmărească firul de răspunsuri la un comentariu,

cât și cel ce a postat acel comentariu, în cazul în care nu a optat să nu fie notificat în legătură cu acesta.

- funcția de ștergere a unui comentariu

Această funcție se execută atunci când se șterge o valoare de la calea `/public/comments/{id}` în baza de date a serviciului *Firestore Database* iar în funcție de tipul comentariului șters, va executa un șir de instrucțiuni. În cazul în care este șters un comentariu al unei postări, se va scădea cu 1 valoare numărului de comentarii a postării, și se vor șterge secvențial toate comentariile de tip răspuns ce aparțin de respectivul comentariu. În cazul ștergerii unui comentariu de tip răspuns, se va scădea cu 1 valoarea numărului de răspunsuri a comentariului din care fac parte.

- funcția de ștergere a unui post

Această funcție se execută atunci când se șterge o valoare de la calea `/public/posts/{id}` în baza de date a serviciului *Firestore Database*, fapt ce va modifica valorile colecției din care face parte, scăzând cu 1 valoare numărului de postări din cadrul ei, cât și să schimbe imaginea de prezentare cu imaginea reprezentativă a ultimei postări din cadrul ei, cât și să șteargă secvențial fiecare comentariu ce aparține ei, fapt ce va declanșa și ștergerea comentariilor de tip răspuns a acestora. De asemenea, numărul de aprecieri totale a canalului din care făcea parte va fi decrementat cu numărul de aprecieri ale respectivei postări.

- funcția de ștergere a unei colecții

Această funcție se execută atunci când se șterge o valoare de la calea `/public/collections/{id}` în baza de date a serviciului *Fireabase Database*, acțiune ce va șterge toate postările canalului din componența ei, iar implicit se va executa funcția menționată anterior pentru fiecare post șters.

- funcția de ștergere a unui canal

Această funcție se execută atunci când se șterge o valoare de la calea `/public/channels/{id}` în baza de date a serviciului *Fireabase Database*, acțiune ce va șterge toate colecțiile canalului din componența sa, iar implicit se va executa funcția menționată anterior pentru fiecare colecție ștersă.

- funcția de ștergere a unui cont de utilizator

Această funcție se execută atunci când se șterge o valoare de la calea `/private/{uid}/user` în baza de date a serviciului *Fireabase Database*, acțiune ce va șterge toate canalele deținute de către utilizator, iar implicit se va executa funcția menționată anterior pentru fiecare canal șters.

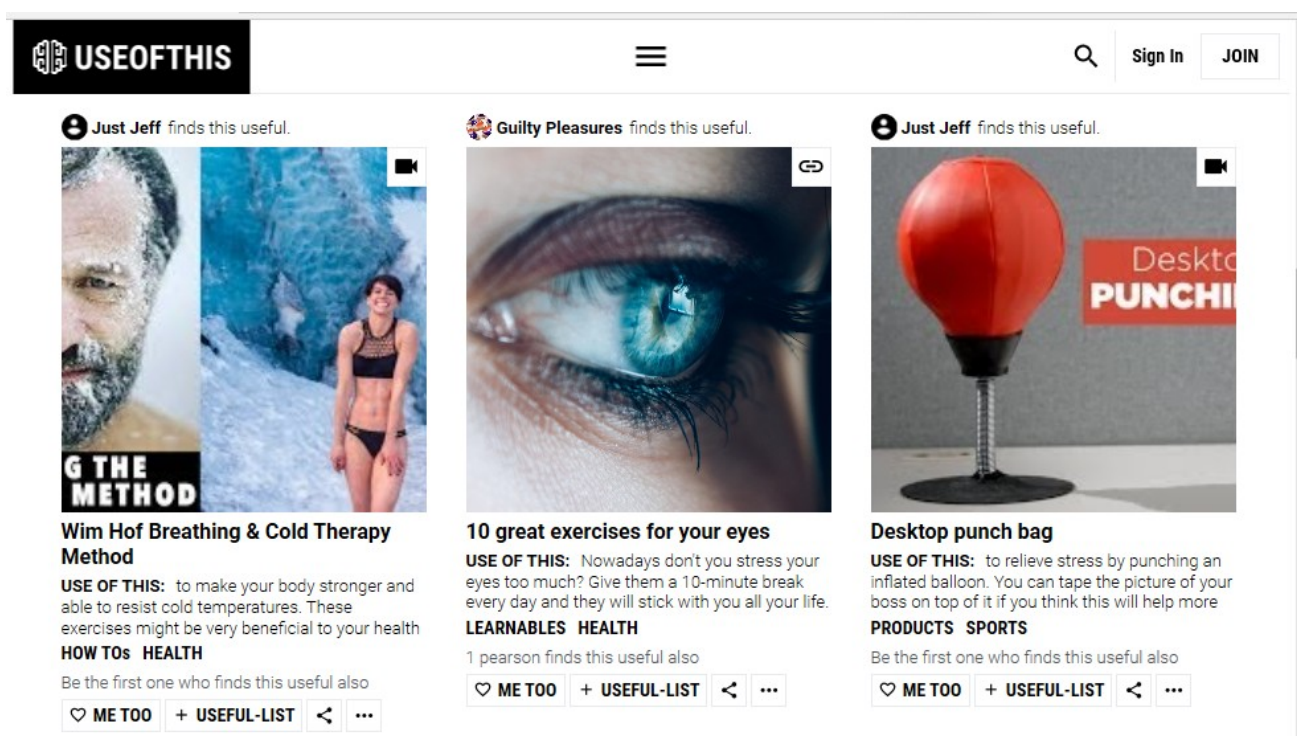
- funcția de tratare a dezabonării utilizatorilor ce și-au șters contul

Această funcție se execută atunci când se șterge o valoare de la calea `/private/{uid}/subscribed/{chId}` în baza de date a serviciului *Fireabase Database*, acțiune declanșată implicit prin ștergerea contului unui utilizator, care va scădea cu 1

valoarea numărului de abonați a tuturor canalelor la care utilizatorul ce și-a șters contul era abonat.

CAPITOLUL 2 – Pagina de explorare a conținutului

Atunci când se accesează adresa lucrării de licență, utilizatorul este direcționat către pagina de explorare a conținutului, unde filtrele sunt inițializate pentru a arăta ultimele 30 cele mai noi postari ale platformei:



Comparativ cu o bază de date relațională, baza de date a serviciului *Firebas Database*, care folosește un obiect *JSON* pentru stocarea datelor, nu permite operații de filtrare și ordonare pe mai multe câmpuri, spre exemplu, în cazul filtrării postărilor, pe baza numărului de voturi și a categoriei simulat. Pentru filtrarea postărilor este nevoie ca la nivelul datei dorite, în cazul acesta, a postării, să existe un câmp al cărui valoare să combine cele doua câmpuri astfel

încât filtrarea dorită să fie obținută. În cazul postărilor, pentru filtrare, exista următoarele câmpuri la calea `public/posts/[post_id]/values`:

- `category_hot`

Valoarea acestui câmp ajută la filtrarea postărilor pe baza câmpurilor `category`, `rating` și `timestamp`, valoare obținută prin concatenarea acestor valori, astfel încât interogarea specifică să primească în ordine descrescătoare timpului postării, acele postări care aparțin categoriei cerute, în funcție de *rating*-ul obținut pe baza numărului de aprecieri.

- `category_new`

Valoarea acestui câmp este obținută prin concatenarea valorilor câmpurilor `category` și `timestamp` a postării, astfel încât interogarea să obțină în ordine descrescătoare timpului postării, acele postări care aparțin categoriei cerute.

- `channel_hot`

Valoarea acestui câmp este obținută prin concatenarea valorilor câmpurilor `channel`, `rating` și `timestamp` a postării, astfel încât interogarea să obțină în ordine descrescătoare timpului postării, acele postări care aparțin unui anumit canal, în funcție de *rating*-ul obținut pe baza numărului de aprecieri.

- `channel_new`

Valoarea acestui câmp este obținută prin concatenarea valorilor câmpurilor `channel` și `timestamp` a postării, astfel încât interogarea să obțină în ordine descrescătoare timpului postării, acele postări care aparțin unui anumit canal.

- `collection`

Valoarea acestui câmp este obținută prin concatenarea valorilor câmpurilor `collection` și `timestamp` a postării, astfel încât interogarea să obțină în ordine descrescătoare timpului postării, acele postări care aparțin unei anumite colecții.

- `rating`

Valoarea acestui câmp este obținută prin concatenarea valorilor câmpurilor `rating` și `timestamp` a postării, astfel încât interogarea să obțină în ordine descrescătoare timpului postării, acele postări care aparțin unei anumite categorii de *rating*, spre exemplu nou, în vogă sau foarte în vogă.

- `type_hot`

Valoarea acestui câmp este obținută prin concatenarea valorilor câmpurilor `type`, `rating` și `timestamp` a postării, astfel încât interogarea să obțină în ordine descrescătoare timpului postării, acele postări care aparțin unui anumit tip, în funcție de *rating*-ul obținut pe baza numărului de aprecieri.

- `type_new`

Valoarea acestui câmp este obținută prin concatenarea valorilor câmpurilor `type` și `timestamp` a postării, astfel încât interogarea să obțină în ordine descrescătoare timpului postării, acele postări care aparțin unui anumit tip.

În cazul filtrării inițiale, atunci când se accesează pagina pentru prima dată, se vor afișa ultimele 30 de postări ale platformei, iar

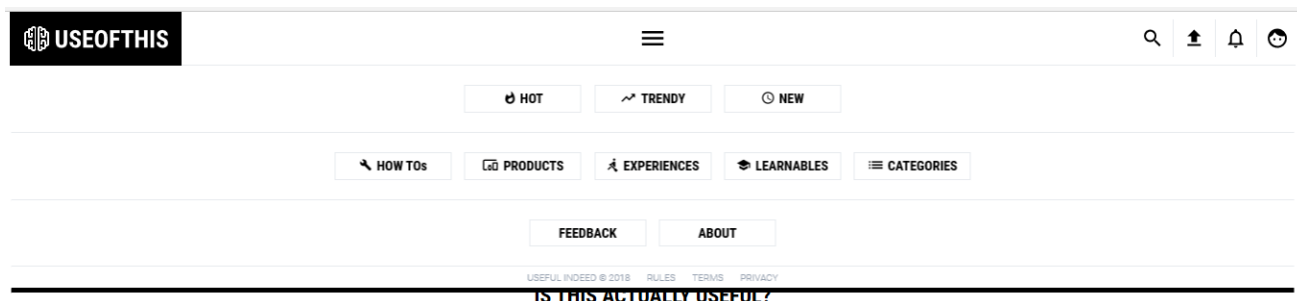
pentru aceasta, se interoghează baza de date cu următoarea secvență de cod:

```
let ref = firebase.database().ref('/public/posts').
  orderByChild('values/rating').
  startAt(['NEW','TRENDY','HOT'].indexOf(urlVars.title) + 1) + '-').
  endAt(['NEW','TRENDY','HOT'].indexOf(urlVars.title) + 1) + "-\uf8ff").
  limitToLast(local.postsRetrieved);
ref.once("value").then(function(snapshot){ displayPosts(snapshot) }
```

Aceasta presupune ca prin intermediul obiectului *javascript* să se folosească serviciul *Firebase Database* pentru această referință `public/posts`, astfel încât să se ordoneze nodurile copil ale acestuia în funcție de valoarea câmpului *rating* de la calea `values/rating` din interiorul nodului ce trebuie sortat, câmp ce trebuie să înceapă cu un număr reprezentativ categoriei căutate, 1 pentru categoria *NEW*, 2 pentru categoria *TRENDY* și 3 pentru categoria *HOT*, limitându-ne la ultimele 30 de postări, valoare salvată în variabila `local.postsRetrieved`. Motivul pentru care la filtrul `endAt` folosim ca și valoare de sfârșit `\uf8ff` este că ea reprezintă un caracter foarte înalt ce îi permite serviciului *Firebase Database*, în procesul de ordonare al câmpurilor, să ia în considerare câmpurile care încep cu o anumită secvență de caractere, în cazul nostru, 1-, 2- sau 3-. După declararea referinței, se accesează baza de date și se procesează răspunsul.

Pentru alte tipuri de filtre, platforma pune la dispoziția utilizatorului un meniu din interiorul căruia se pot selecta cele 3 categorii de filtrare, *rating*, tip și categorie, iar din pagina fiecărui filtru, se pot obține postările în funcție de noutate sau numărul de

aprecieri. În toate cazurile de filtrare, se va urmări un proces similar celui menționat anterior. Meniul arată astfel:



Bara de meniu are în partea stângă un *logo* ce direcțează către pagină de start, butonul din centru ce deschide panoul de meniu, butonul de căutare ce direcțează către o pagină creată specific acestui lucru, butonul de creare a unui post, butonul de notificări cât și butonul de utilizator.

Panoul de meniu afișat prin apăsarea butonului specific din centrul barei de meniu, este împărțit în patru părți. Prima parte conține butoanele de filtrare a postărilor pe baza popularității lor. Butoanele *Hot* și *Trendy* vor afișa doar postările care au depășit un anumit număr de aprecieri, dar butonul *New* va afișa toate postările în ordine descrescătoare creării lor. Partea a două are butoane pentru categoriile postărilor, aceste categorii fiind de două tipuri, categorie de aplicabilitate și categorie de domeniu. Categoriile de aplicabilitate sunt *How Tos*, unde este postat conținut ce arată cum un utilizator poate crea un anumit lucru, *Products*, unde este postat conținut ce prezintă produse, *Experiences*, categorie unde se află postările despre evenimente, sau produse ce oferă experiențe, cât și categoria *Learnables*, unde se află conținut ce vine în ajutorul utilizatorului de a

învață un anumit lucru. Categoriile de domeniu înglobează în cadrul termenilor o sferă din care face parte informația, spre exemplu artă, design, programare, sănătate, divertisment, etc.

Scopul acestei platforme este de a pune în evidență utilitatea informației, iar pentru asta, fiecare post pus la dispoziția utilizatorului, fie în pagina de explorare fie pe pagina dedicată lui din cadrul canalului din care face parte, are următoarele componente: componenta de canal, ce reprezintă ce canal a găsit respectiva postare utilizabilă, componenta de imagine reprezentativă, pentru analiza eficientă a conținutului oferit, componenta de titlu, componenta de utilitate, în care utilizatorul specifică de ce crede că ce a postat este util, componenta de categorie, și componenta de acțiuni. Pentru a realiza acțiuni de tipul apreciere, adăugare în lista proprie de conținut util, adăugare de comentarii, utilizatorul trebuie să fie autentificat.

Atunci când un utilizator autentificat apreciază o postare prin apăsarea butonului *me too*, se apelează o funcție care verifică dacă este vorba de o apreciere sau o dezapreciere, după care tratează instrucțiunea ca atare. În cazul unei aprecieri, se va modifica baza de date cu instrucțiunea:

```
firebase.database().ref('private/' + uid + '/likes/' + scope.index.posts[index].id).set({
  state: 'on',
  type: 'post'
}).then(function(){ ... });
```

În cazul unei dezaprecieri, modificarea se va realiza cu instrucțiunea:

```
firebase.database().ref('private/' + uid + '/likes/' + scope.index.posts[index].id).update({
  state: 'off',
  type: 'post'
}).then(function(){ ... });
```

În ambele cazuri se modifică valorile din baza de date aflate la referința `private/[uid]/likes/[id-ul postului apreciat]` cu un obiect ce specifică tipul aprecierii, dacă e pentru un post sau pentru un comentariu, și dacă aprecierea este activă sau nu. În cazul dezaprecierii, starea aprecierii devine dezactivată.

Prin modificarea câmpului de apreciere a unui post găsit la calea menționată anterior, se va declanșa o funcție din serviciul *Firebase Functions* ce va incrementa numărul aprecierilor de la nivelul postării cu 1, iar în funcție de numărul de voturi, va modifica valorile de filtrare ce fac referire la categoria de *rating* la care se încadrează postul, cu noul număr de aprecieri. Astfel, un post cu 20 de aprecieri va deveni în vogă, iar odată ce depășește 50 de aprecieri, va deveni foarte în vogă.

Pentru autentificare, utilizatorul trebuie să apese unul din cele două butoane puse la dispoziție în partea dreaptă sus, în cazul vizualizării de tip *desktop*, sau stânga jos, în cazul vizualizării de pe dispozitive mobile, și anume *Sign In*, dacă dorește să folosească un cont deja existent, sau *Join*, dacă este un utilizator nou și dorește să-și creeze un cont nou.

Utilizatorul are posibilitatea de a se autentifica cu ajutorul unui cont existent ce aparține unei terțe părți, precum *Facebook*, *Twitter* sau *Google+*, dar și prin intermediul unui cont creat pe platforma în cauză cu ajutorul unui email și o parolă. În cazul în care utilizatorul a uitat parola, acesta poate apăsa pe butonul *I forgot my password*, fapt ce îl va direcționa printr-o secvență de panouri ce îi vor cere adresa de email a contului, iar dacă există un cont atribuit ei, el va primi pe ea o adresă ce directează către o pagină de resetare a parolei.

Evidența statusului autentificării a utilizatorului curent este menținută la nivelul clientului prin intermediul serviciului *Firebase Authentication*. La început, când se inițializează aplicația web, se apelează următoarea funcție:

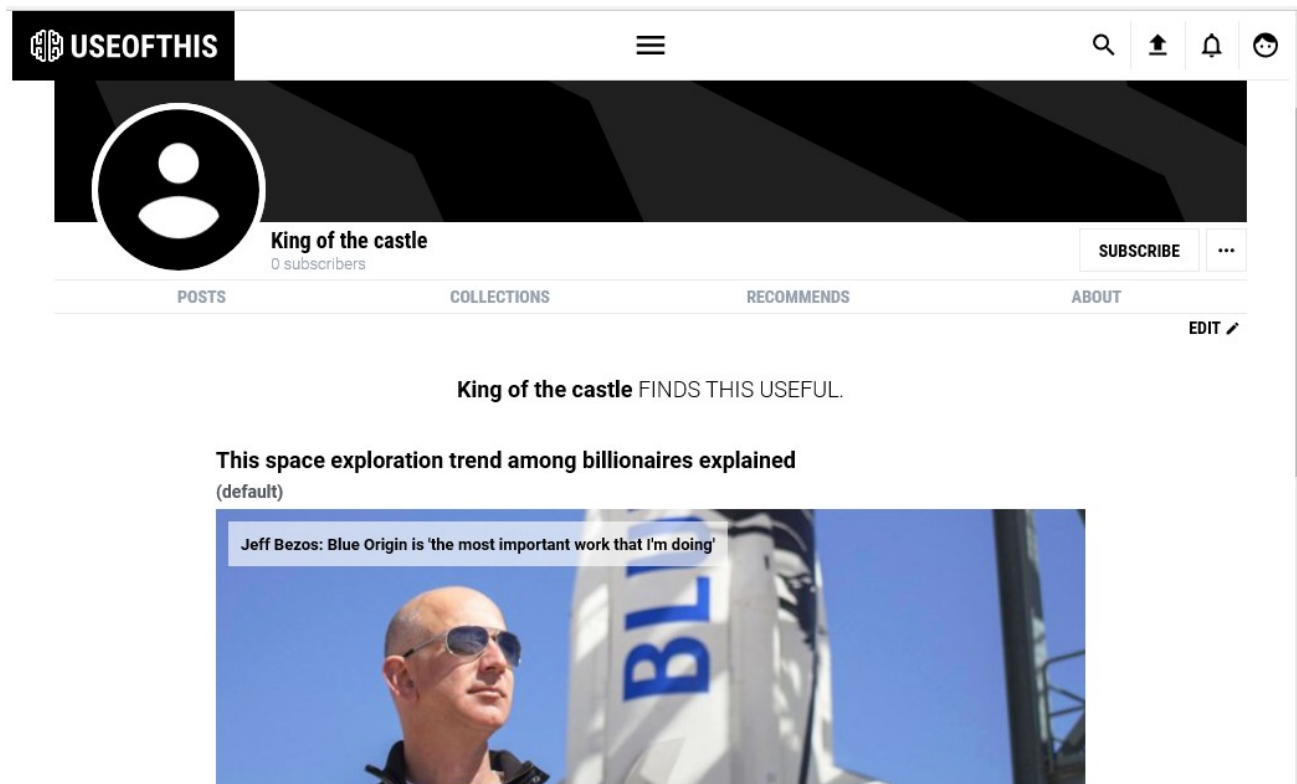
```
firebase.auth().onAuthStateChanged(function(user) {  
  if (user) {  
    checkNotifications();  
    local.notificationCheck = setInterval(checkNotifications, 30000);  
    scope.index.authState = true;  
    getUserData(user);  
  } else {  
    scope.index.authState = false;  
    scope.index.user = {};  
    clearInterval(local.notificationCheck);  
  }  
});
```

Această secvență de cod urmărește statusul autentificării utilizatorului curent, status ce se schimbă la comenzi de autentificare sau de părăsire al acestui status. Dacă utilizatorul se autentifică, se va

verifica dacă există noi notificări pentru respectivul utilizator, se va seta ca această verificare să aibă loc la fiecare 30 de secunde și se modifică variabile care înștiințează întreaga platformă de noua modificare. În cazul invers, când utilizatorul se deloghează, se întrerupe verificarea notificărilor și se schimbă variabilele pentru propagarea schimbărilor în întreaga aplicație.


CAPITOLUL 3 – Pagina de canal

Din pagina de explorare a postărilor se poate accesa o singură postare ce va directa utilizatorul către componenta respectivului post din interiorul paginii canalului ce la făcut disponibil:



Fiecare pagină de post începe prin a prezenta premisa platformei, și anume că acest canal găsește postarea făcută disponibilă că fiind folositoare, urmat de titlu, numele colecției din care face parte, imaginea reprezentativă, rubricat utilizabilității, acțiunile de apreciere, adăugare în lista proprie de postări găsite ca fiind utile, butonul de partajare a postării pe alte rețele sociale, rubrica de sursă, unde apare adresa către care face postarea referință, și rubrica de

comentarii. Utilizatorul are nevoie să dețină un canal pentru a posta un comentariu.

 Just Jeff ▾


MESSAGE

COMMENT

NEW


HOT

1 comment

 King of the castle


Awesome! I will try to be consistent about it for a week or so, and i'll tell you my results :D

LIKE (1), REPLAY (1)

 Just Jeff

Thank you!

LIKE (0)

 Just Jeff ▾

MESSAGE

REPLAY

Pentru a posta un comentariu, trebuie să se selecteze canalul căruia îi va fi atribuit acesta, se scrie mesajul, după care se apasă butonul de *comment*. Comentariile unui post pot fi sortate în funcție de vechime, sau în funcție de numărul de aprecieri. În acest sens, fiecare comentariu postat poate fi apreciat. Pentru ordonarea comentariilor în funcție de numărul de aprecieri sau a vechimii se procedează ca și în cazul postărilor, aceasta realizându-se pe baza a unei valori obținute prin concatenarea a doua sau trei câmpuri, în funcție de tipul sortării. Pentru cele două tipuri de filtrări sunt setate următoare câmpuri la nivelul unui comentariu:

- `comment_hot`

Valoarea acestui câmp este obținută prin concatenarea valorilor câmpurilor `post_id`, `rating` și `timestamp` a comentariului, astfel încât interogarea să obțină în ordine descrescătoare timpului postării, acele comentarii care aparțin unui anumit post, în funcție de *rating*-ul obținut pe baza numărului de aprecieri.

- `comment_new`

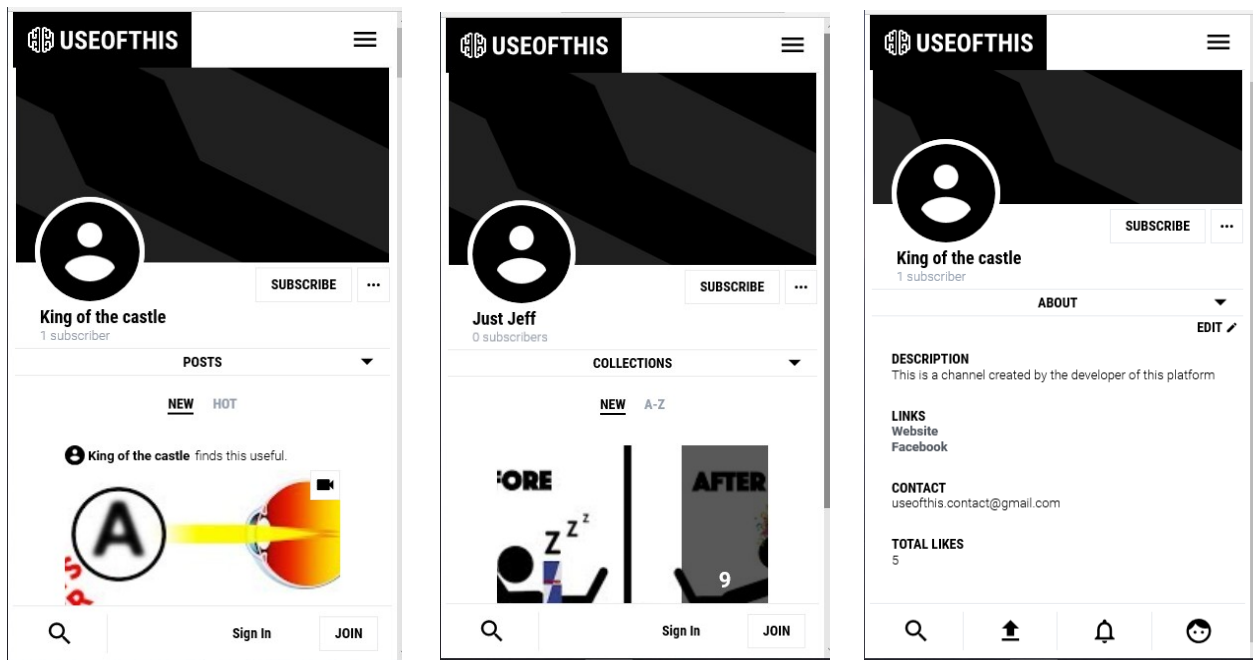
Valoarea acestui câmp este obținută prin concatenarea valorilor câmpurilor `post_id` și `timestamp` a comentariului, astfel încât interogarea să obțină în ordine descrescătoare timpului postării, acele comentarii care aparțin unui anumit post.

Pentru a lăsa un răspuns la un comentariu se procedează la fel ca și în cazul unui comentariu normal, se selectează canalul, se adaugă mesajul, dar de data aceasta, se apasă butonul *reply*. Atunci când se apasă pe unul din cele două butoane de comentarii, acel comentariu este salvat în baza de date la calea `public/comment/[id]`. Acest lucru declanșează funcția din serviciul *Firebase Functions*, și își va urma execuția așa cum a fost descrisă în capitolul dedicat integrării. În cazul în care utilizatorii doresc să nu mai primească notificări sau invers, doresc să primească notificări la fiecare răspuns pentru un anumit comentariu, aceștia au posibilitatea de a se abona la activitatea unui comentariu prin apăsarea butonului *Unfollow* sau *Follow* al

acestuia. Acest lucru îi va adăuga sau șterge din lista de abonați ai comentariului găsită la calea `public/comment/[id]/followers`.

Fiecare pagină de canal are în prima parte a sa o componentă ce prezintă identitatea acestuia. Aici, este prezentă imaginea de copertă, imaginea de profil, numele canalului, numărul de abonați cât și butonul de abonare. Atunci când un utilizator se abonează la un canal, acesta adaugă în baza de date, la calea `private/[uid]/subscribed/[channel_id]` următorul obiect: `{ state: 'on' }`. În cazul în care acest obiect este nou adăugat, cel ce deține canalul va fi înștiințat printr-o notificare că, canalul său a obținut un nou abonat, asta în cazul în care nu a optat să nu mai primească notificări în legătură cu acest lucru. În cazul în care obiectul există deja, nu se va crea o nouă notificare, ci se va seta valoarea câmpului `state` înapoi în `on`. Dacă utilizatorul dorește să se dezaboneze de la canal, prin apăsarea butonului respectiv, valoarea butonului `state` va deveni `off`. În ambele cazuri se va declanșa o funcție de la nivelul serviciului *Firebase Functions* care va modifica numărul de abonați ai canalului cât și va trata notificările necesare operațiunii pentru utilizatorul ce deține canalul.

Pe lângă componenta de post individual din cadrul paginii de canal, platforma pune la dispoziție alte 3 componente: componenta de postări, componenta de colecții, cât și componenta despre.



Componenta de postări a canalului prezintă toate postările acestuia. Ele pot fi ordonate în funcție de vechime sau de numărul de aprecieri. În ambele cazuri se execută interogarea următoare, unica diferență fiind nodul e final. În cazul filtrării pe baza vechimii, se va accesa nodul frunza `channel_new`, iar în cazul filtrării pe baza numărului de aprecieri, nodul `channel_hot`:

```
let ref = firebase.database().ref('/public/posts').
  orderByChild('values/channel_new').
  startAt(urlVars.id + '-').
  endAt(urlVars.id + '-' + "\uf8ff").
  limitToLast(local.postsRetrieved);
ref.once("value").then(function(snapshot){ ... });
```

Această secvență de cod obține ultimele 30 de noduri copil de la calea `public/posts` care încep cu id-ul canalului curent în nodul frunză de la calea `values/channel_new`, după care se procesează rezultatul obținut.

Componenta de colecții afișează toate colecțiile canalului, fiind ordonate în funcție de data creării, sau ordonate alfabetic. Pentru a

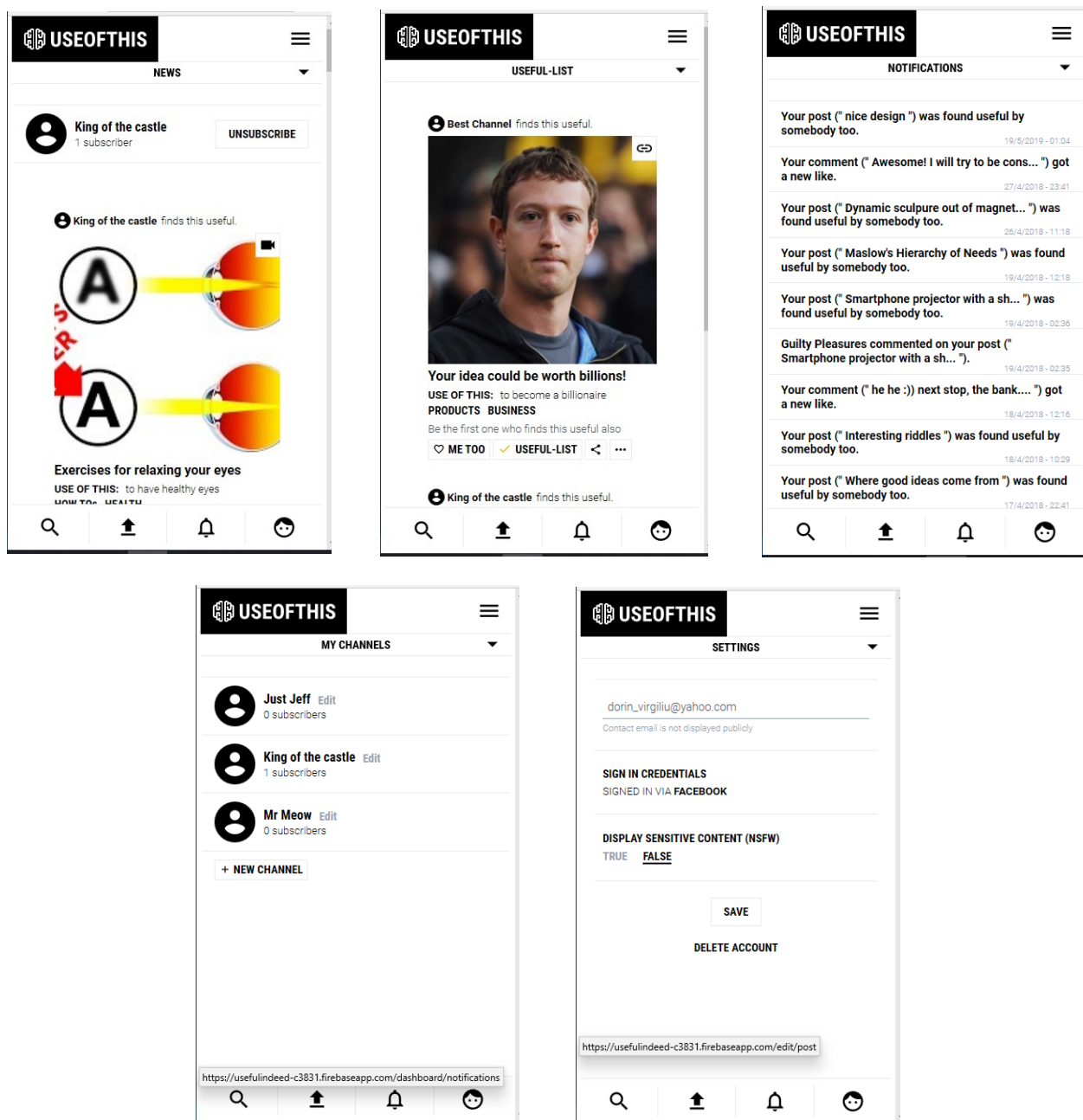
obține toate colecțiile unui canal se urmează un proces asemănător cu cel al obținerii postărilor, însă de această dată, se interoghează pe o valoare completă, nu doar pe o secvență de început. Filtrarea se realizează la nivelul clientului, deoarece se obțin toate colecțiile, nu parțial ca în cazul postărilor.

```
firebase.database().ref("public/collections/").  
  orderByChild('channel').  
  equalTo(scope.index.page.urlVars.id).  
  once("value").then(function(snapshot) { ... });
```

Această secvență de cod obține toate nodurile de la calea `public/collections` care au valoarea nodului frunză `channel` egală cu id-ul canalului pentru care dorim să le obținem.

CAPITOLUL 4 – Pagina panoului de comandă

Fiecare utilizator autentificat are abilitatea de a accesa propriul panou de comandă unde va găsi cinci componente de conținut, fiecare componentă de conținut fiind afișată personalizat în funcție de activitatea utilizatorului:



Componenta de noutăți pune la dispoziția utilizatorului abilitatea de a vedea toate canalele la care s-a abonat. Lista de canale este preluată din baza de date de la calea `private/users/[uid]/subscribed`, iar dacă subscripția este activă, atunci respectivul canal va fi afișat în lista utilizatorului. Această listă conține inclusiv butonul de dezabonare de la canal. În cazul în care este apăsat, are loc același proces că și în cazul butonului de pe pagina dedicată canalului respectiv.

De asemenea, utilizatorului îi este afișat în ordine descrescătoare momentul postării, toate postările canalelor la care a s-a abonat. Pentru asta, la nivelul utilizatorului se aplică un algoritm în care pentru toate canalele la care s-a făcut un abonament se ia ultimul cel mai nou post, din lista obținută se va lua cea mai nouă postare și se va adăuga în lista postărilor afișate utilizatorului. Se va lua următoarea postare ca și noutate de la canalul de la care a fost selectat postul adăugat și se va repeta procesul până când treizeci de postări vor fi afișate utilizatorului, sau până când nu vor mai fi postări disponibile pe niciunul din canalele la care utilizatorul s-a abonat.

În componența de listă utilă se vor găsi toate postările pentru care utilizatorul a apăsat pe butonul dedicat acestui lucru, disponibil mereu când se afișează o postare. Pentru aceasta, se va interoga baza de date la calea `private/users/[uid]/usefulList` pentru a obține lista postărilor și pentru fiecare în parte se va face o nouă interogare

pentru a obține toate postările la care se face referință în interiorul listei.

Panoul de comandă pune la dispoziția utilizatorului și componenta de notificări. În cazul în care utilizatorul are notificări pe care nu le-a văzut încă, în bară de meniu iconița sub formă de clopot va indica acest lucru. Pe această pagină utilizatorul va fi înștiințat printr-un mesaj sugestiv dacă cineva a adăugat un comentariu la postarea să, accesând această notificare va fi redirecționat către pagină postării iar în secțiunea de comentarii va fi pus în evidență acel comentariu, la fel se va întâmpla și în cazul notificării pentru un răspuns la un comentariu propriu. Utilizatorul este înștiințat și atunci când cineva apreciază o anumită postare, redirectarea ducând pe pagină postării, și este notificat și atunci când cineva se abonează la unul din canalele sale, redirectarea făcându-se către pagină respectivului canal.

Pentru aceasta, se interoghează baza de date la calea `private/users/[uid]/notifications` pentru a obține ultimele 30 de notificări. De asemenea, utilizatorul are salvat id-ul ultimei notificări vizualizate. Acest lucru ajută la punerea în evidență a notificărilor noi. Odată ce utilizatorul deschide pagina notificărilor, această valoare se modifică cu id-ul noii ultime notificări, dar nu înainte de a fi folosită vechea valoare pentru a le evidenția pe cele noi.

Componenta canalelor personale permite vizualizarea tuturor acestora cât și abilitatea de a crea unul nou. În vederea canalelor deja existente, ne este afișat numărul de abonați cât și un buton ce ne

permite să edităm câmpurile acestuia. Lista canalelor este obținută prin interogarea bazei de date la calea `private/users/[uid]/user/channels`.

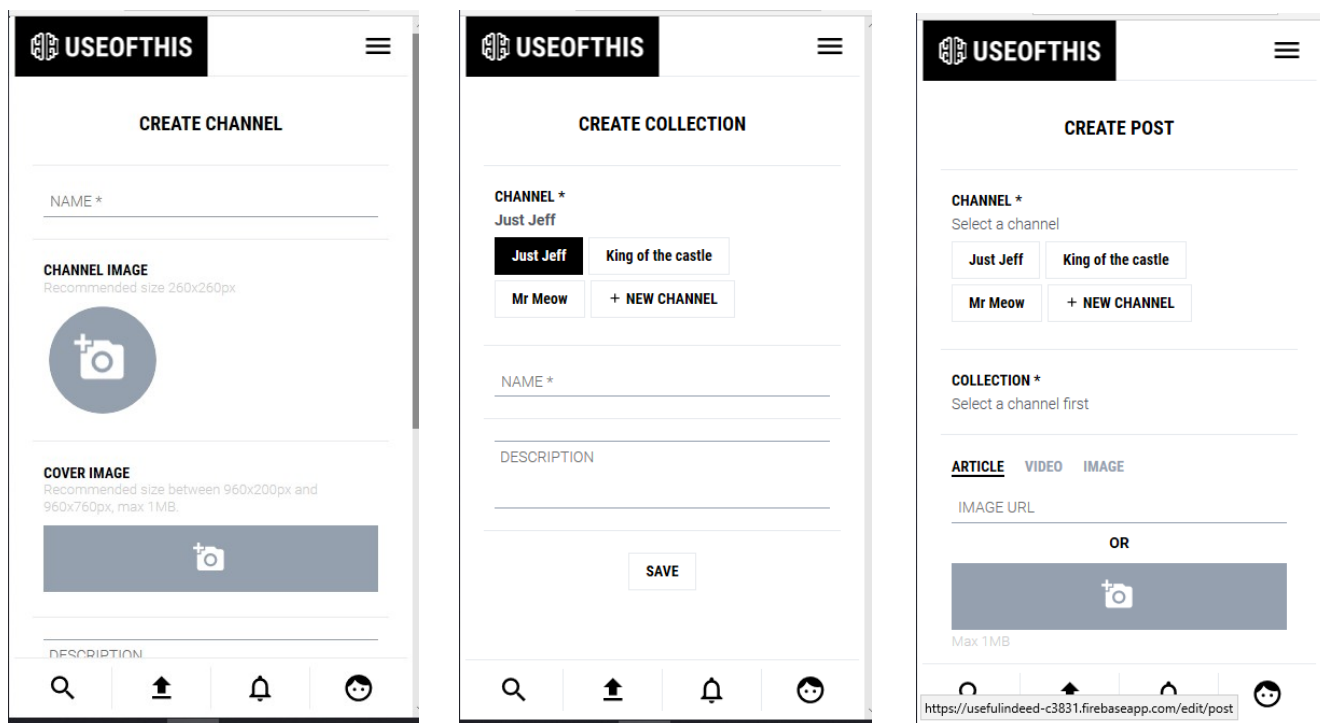
Pagina de setări pune la dispoziția utilizatorului abilitatea de a seta o adresă de email de contact, adresa ce nu va fi făcută publică, abilitatea de a vedea ce metodă de autentificare a folosit, dacă s-a autentificat cu ajutorul unui email și o parolă sau dacă a folosit serviciile unei terțe părți, cât și setarea dacă dorește ca platforma să îi afișeze conținut sensibil sau nu. Atunci când se apasă pe butonul de salvare, noile date sunt introduse la calea `private/users/[uid]/user/settings`.

De asemenea, această componentă pune la dispoziția utilizatorului și butonul de ștergere a contului. Această funcție va șterge nodul de la calea `private/users/[uid]`, lucru ce va declanșa o funcție definită în serviciul *Firebase Functions* ce va iniția o serie de ștergeri consecutive. Ștergerea contului de utilizator va declanșa ștergerea tuturor canalelor utilizatorului. Ștergerea canalelor va presupune ștergerea tuturor colecțiilor din componenta lor. Ștergerea colecțiilor va declanșa ștergerea tuturor postărilor ce aparțin de ele. Ștergerea postărilor va presupune ștergerea tuturor comentariilor, iar acest lucru presupune și ștergerea tuturor răspunsurilor la acele comentarii.

În cazul în care utilizatorul accesează adresa către resursa făcută disponibilă în cadrul notificării, iar aceasta nu mai există, respectiva notificare va fi ștearsă.

CAPITOLUL 5 – Pagini de creare de conținut

În cadrul acestei lucrări de licență putem crea trei lucruri, canale, colecții la nivelul unui canal, cât și o postare. Fiecare din acestea are o pagină individuală acestui scop.



Pentru crearea unui canal, utilizatorul trebuie să meargă pe pagina panoului de comandă personal, să acceseze categoria de canale personale, să apese pe butonul de creare a unui nou canal, lucru ce-l va redirecta pe pagină de pe care se poate realiza acest lucru și să completeze câmpurile puse la dispoziție. Aceste câmpuri sunt: numele canalului, care trebuie să fie unic în cadrul platformei, în caz contrar se va afișa un mesaj de eroare, o imagine de profil pentru canal, cu o rezoluție recomandată de 260x260px, o imagine de copertă, cu o

dimensiune cuprinsă între 960x200px și 960x760px, și cu o dimensiune mai mică de 1MB. În ambele cazuri este nevoie să se pună la dispoziție fișiere de pe mașina locală, altfel, se vor inițializa cele două imagini cu fișiere prestabilite. Utilizatorul are abilitatea și de a oferi o descriere canalului, să ofere adrese către diferite platforme externe, spre exemplu către un *site* sau către o pagină de pe o platforma de socializare, o adresă de contact cât și dacă dorește să primească notificări de la acest canal, fie notificări în legătură cu postările acestuia, precum aprecieri, comentarii, fie notificări ce cuprind noi abonați.

Pentru a crea o nouă colecție de postări, utilizatorul trebuie să acceseze pagină canalului pe care dorește să adauge colecția, să acceseze pagina colecțiilor canalului și să apese pe butonul de creare. Acest lucru îl va direcționa către o pagină dedicată acestui lucru, va avea preselectat canalul de pe care a fost cerută instrucțiunea, însă utilizatorul are posibilitatea de a schimba canalul pentru care crează nouă colecție. Apoi trebuie să specifice numele colecției și o descriere sugestivă. La apăsarea butonului *save* va fi direcționat către pagina colecției nou create.

Pentru a posta un conținut nou, utilizatorul are trei opțiuni, prima fiind prin accesarea butonului de creare de nou conținut din meniu, lucru ce îi va cere să completeze de pe ce canal deja creat dorește să facă postarea publică, a doua fiind din pagină canalului de pe care va dori să facă postarea disponibilă la rubrica postări, lucru ce îi va preselecta canalul ce va deține postarea, sau a treia opțiune, din

interiorul colecției, lucru ce îi va preselecția canalul, cât și colecția din care va face parte.

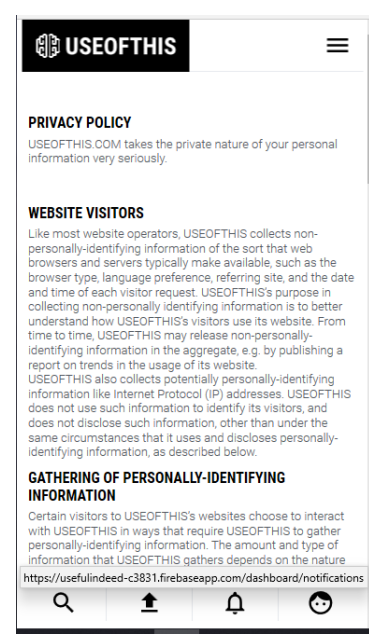
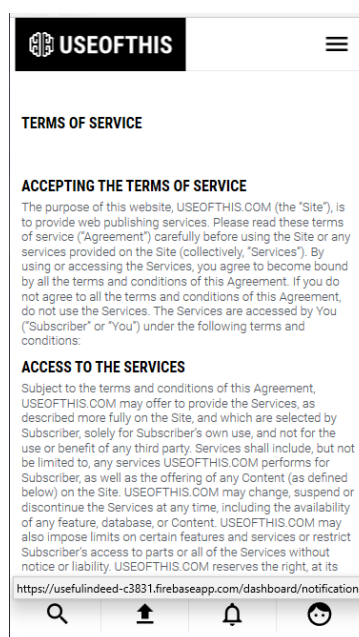
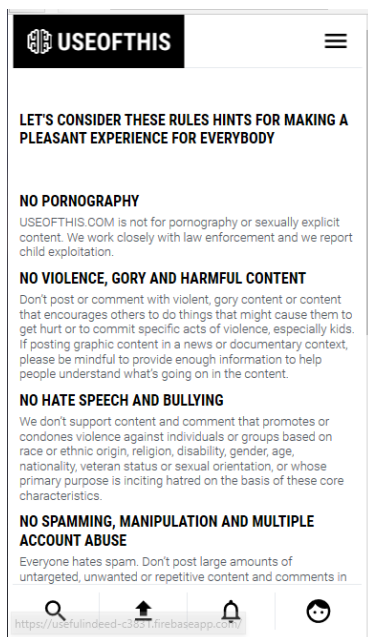
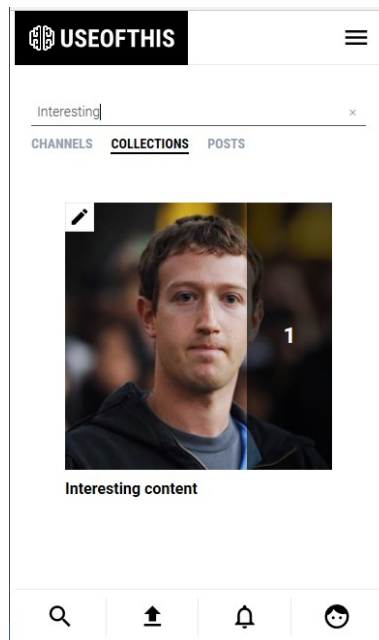
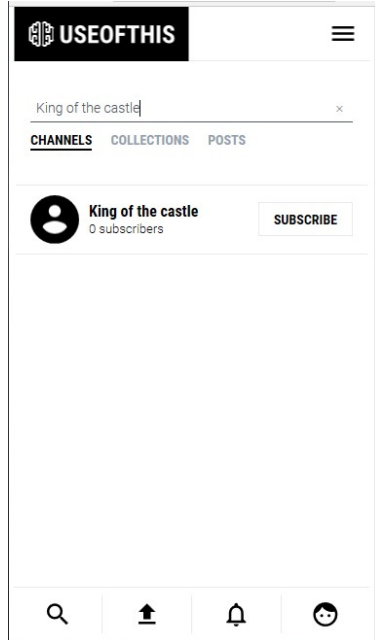
Pagina postării de nou conținut pune la dispoziția utilizatorului câmpul de canal, odată selectat, se va completa următorul câmp, cel al colecție din care va face parte postul, după care utilizatorul va trebui să selecteze tipul conținutului, dacă este un articol, videoclip sau imagine. În cazul unui post de tip articol, se poate seta o imagine, printr-o adresa sau prin selectarea ei de pe mașina locală, titlul articolului și adresa unde poate fi găsit. În cazul unui videoclip, platforma suportă doar adrese de pe platforma *youtube.com* și este deajuns să se introducă adresa la care poate fi găsit videoclipul, imaginea postării va fi extrasă automat după cum este oferită de platforma menționată. În cazul unei imagini, este deajuns să se specifice o adresa la care este disponibilă imaginea sau să se încarce de pe mașina locală. Următoarea dată ce trebuie completată este titlul postării, apoi utilitatea ei, un câmp obligatoriu pentru toate postările, cât și tipul acesteia, dacă este o postare ce învață utilizatorul să creeze un lucru, prezintă un produs, o experiență sau încearcă să învețe utilizatorul o anumită informație. Următorul câmp este selectarea categoriei generalizate, spre exemplu sport sau muzică, urmat de câmpul dacă utilizatorul crede că postarea prezintă conținut capabil să jignească pe cineva, în limitele impuse de regulile platformei, apoi câmpul cu sursa unde a fost găsit conținutul, apoi posibile adrese către diferite pagini și ultimul câmp, dacă dorește să primească notificări în

legătură cu respectivul post în cazul în care este apreciat sau un utilizator adaugă un comentariu acestuia.

Pentru editare sau ștergere de conținut, fie el pagină de canal, colecție sau postare, este nevoie să se acceseze conținutul și să se apese butonul de editare specific fiecăruia dintre ele. Acest lucru va redirecta către paginile descrise anterior, dar cu câmpurile precompletate, iar sub butonul de salvare va apărea inclusiv butonul de ștergere a conținutului. În toate cazurile sunt anumite câmpuri ce nu mai pot fi schimbate la editare. Spre exemplu, în cazul canalului, numele acestuia nu mai poate fi schimbat, în cazul colecțiilor, canalul de care aparțin nu mai poate fi schimbat, iar în cazul postărilor, canalul de care aparțin cât și conținutul pe care îl prezintă.

CAPITOLUL 6 – Alte pagini

Platforma pune la dispoziția utilizatorului și alte pagini, precum pagina de căutare de canale, colecții sau postări, pagina de reguli, pagina de termeni și pagina de confidențialitate a datelor personale.



Pagina de căutare este accesibilă din bara de meniu, prin apăsarea iconiței specifice acestui lucru. Această pagină pune la dispoziția utilizatorului căutarea de canale, colecții, cât și postări. Afișarea rezultatelor se realizează activ, nefiind nevoie să se apese pe un buton după ce se introduce șirul de caractere cu care să înceapă numele resurselor căutate.

Pentru asta, de fiecare dată când se introduce sau se șterge o valoare în câmpul de căutare, se apelează o funcție *javascript* care preia noul șir de caractere, pe baza căruia va interoga baza de date, dar doar după ce a fost returnat un rezultat pentru șirul de caractere precedent. În cazul în care noul șir de caractere este obținut înainte ca rezultatul interogării anterioare să fie obținut, acesta va aștepta ca interogarea să se finalizeze, iar în cazul în care șirul nu a fost modificat, se va interoga baza de date cu acesta, altfel, cu șirul de caractere nou obținut în timpul așteptării unui răspuns.

În cazul interogării pentru obținerea canalelor, șirul de caractere este transformat prin schimbarea tuturor caracterelor mari în caractere mici și eliminarea tuturor caracterelor de tip spațiu din componenta sa. Acesta este mai departe folosit în următoarea structură:

```
firebase.database().ref('public/channels').  
  orderByKey().  
  startAt(input).  
  endAt(input + "\uf8ff").  
  limitToLast(local.searchLimit).  
  once('value').then(function(snapshot){ ... });
```

Id-ul canalului este obținut prin același principiu că și în cazul obținerii șirului de caractere folosit la căutare, prin urmare, putem

caută canale în funcție de id-ul lor. Secvența de cod caută la adresa `public/channels` nodurile cu numele începând cu șirul de caractere obținut anterior, limitându-ne la 12 rezultate.

În cazul interogării pentru obținerea colecțiilor, șirul este transformat în aceeași maniera ca și în cazul canalelor, însă de această dată, secvența de căutare face referire la calea `public/collections/[id]/content/nameSearch`. Valoarea câmpului `nameSearch` este obținută prin transformarea tuturor caracterelor numelui în caractere mici și sunt eliminate spațiile. Următoarea secvență de cod returnează toate canalele ale căror nume începe cu secvența de caractere introdusă de către utilizator:

```
firebase.database().ref('public/collections').  
  orderByChild('content/nameSearch').  
  startAt(input).  
  endAt(input + "\uf8ff").  
  limitToLast(local.searchLimit).  
  once('value').then(function(snapshot){ ... });
```

Pentru obținerea postărilor, se urmăresc aceeași pași ca și în cazul colecțiilor, unica diferență fiind că referința este pentru calea `public/posts`, iar valoarea introdusă în locul `content/nameSearch` este `content/titleSearch`.

Pagina regulilor de postare de conținut specifică regulile ce trebuie urmate atunci când se postează conținut nou. În cazul în care un utilizator consideră că aceste reguli au fost încălcate de o anumită postare, poate să raporteze respectiva postare prin intermediul unui buton atribuit fiecărei postări dedicat acestui lucru.

De asemenea, platforma are o pagină ce specifică garanția către utilizatori că drepturile nu le vor fi încălcate, cât și o pagină de

confidențialitate, în care se specifică cum este folosit conținutul cu caracter personal.

CONCLUZII

Această lucrare de licență a urmărit să pună în evidență avantajele folosirii arhitecturii *serverless* în detrimentul uneia specifică sistemelor *cloud* clasice ce folosesc programe *server* de livrare a conținutului, arhitectură ce necesită utilizarea resurselor chiar și în momente în care aplicația făcută disponibilă nu este interogată de către un utilizator. Avantajele principale sunt: costurile mult mai scăzute de menținere a aplicației active cât și timpul mult mai scăzut pentru a o dezvolta, deoarece se folosesc servicii deja existente pentru diverse operațiuni.

În opinia mea, aceste două avantaje au fost demonstrate de faptul că pentru a menține aplicația activă nu a trebuit să plătesc nimic serviciului *Firebase*, pe când în contextul serviciilor *cloud* clasice ar fi fost necesar să achit o suma de alocare a resurselor în fiecare luna de menținere a aplicației activă, cu toate că nu ar fi fost folosită de către un client, iar din punct de vedere al eficienței dezvoltării, serviciile de bază de date, de autentificare, de stocare, de livrare a fișierelor statice cât și de rulare de funcții la nivelul *serverului* prin apelare de adrese au contribuit semnificativ în livrarea mai rapidă a unei platforme, după părerea mea, complexă.

Din punct de vedere al direcțiilor viitoare în ceea ce privește platforma, procesul de dezvoltare m-a ajutat să-mi dezvolt abilitatea de a privi funcționalitățile aplicației din două perspective, din

perspectiva utilizatorului final cât și a dezvoltatorului ce încearcă să creeze un produs cât mai bun pentru acesta. De aceea, aş spune că paşii următori pentru această lucrare de licenţă ar fi îmbunătăţirea experienţei utilizatorului în interacţiunea sa cu ea cât şi implementarea unor noi funcţionalităţi, în funcţie de nevoile utilizatorului final descoperite ulterior.

Bibliografie

[1] *Tipul de date salvat în serviciul Firebase Database*

<https://firebase.google.com/docs/database/>

Platforme ce oferă servicii serverless

<https://serverless.com/framework/docs/providers/>

Diferențe între aplicații single-page și multi-page

<https://medium.com/@NeotericEU/single-page-application-vs-multiple-page-application-2591588efe58>

Setarea proiectelor ce folosesc serviciile Firebase

<https://firebase.google.com/docs/web/setup>

Setarea serviciului Firebase Hosting

<https://firebase.google.com/docs/hosting/quickstart>

Setarea serviciului Firebase Authentication

<https://firebase.google.com/docs/auth/web/start>

Setarea serviciului Firebase Database

<https://firebase.google.com/docs/database/web/start>

Setarea securității serviciului Firebase Database

<https://firebase.google.com/docs/database/security/>

Setarea serviciului Firebase Storage

<https://firebase.google.com/docs/storage/web/start>

Setarea securității serviciului Firebase Storage

<https://firebase.google.com/docs/storage/security>

Setarea serviciului Firebase Functions

<https://firebase.google.com/docs/functions/get-started>