UNIVERSITATEA TEHNICĂ „Gheorghe Asachi” din IAȘI FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE DOMENIUL: CALCULATOARE ȘI TEHNOLOGIA INFORMAȚIEI SPECIALIZAREA: TEHNOLOGIA INFORMAȚIEI

**Aplicație nor pentru administrare la distanță**

LUCRARE DE DIPLOMĂ

Coordonator științific

conf. dr. ing. Mihai Horia Zaharia

absolvent

Țuțuianu Robert-Constantin

Iași, 2023

Cuprins

[1. Introducere 1](#_Toc139559502)

[2. Documentarea problemei 2](#_Toc139559503)

[2.1 TeamViewer 3](#_Toc139559504)

[I. Prezentare generală 3](#_Toc139559505)

[II. Tehnologii folosite 3](#_Toc139559506)

[III. Argumente pro 3](#_Toc139559507)

[IV. Argumente contra 4](#_Toc139559508)

[2.2 Anydesk 4](#_Toc139559509)

[I. Prezentare generală 4](#_Toc139559510)

[II. Tehnologii folosite 4](#_Toc139559511)

[III. Argumente pro 4](#_Toc139559512)

[IV. Argumente contra 5](#_Toc139559513)

[2.3 LogMeIn 5](#_Toc139559514)

[I.Prezentare generală 5](#_Toc139559515)

[II. Tehnologii folosite 5](#_Toc139559516)

[III. Argumente pro 5](#_Toc139559517)

[IV. Argumente contra 6](#_Toc139559518)

[3. Abordare teoretică 7](#_Toc139559519)

[3.1 Utilitatea aplicațiilor de control la distantă 7](#_Toc139559520)

[3.2 Scenarii de utilizare 7](#_Toc139559521)

[1. Gestionare conexiune cu Kafka 8](#_Toc139559522)

[2. Autentificare și gestionare cont 10](#_Toc139559523)

[3. Control și monitorizare la distanță 12](#_Toc139559524)

[4. Administrare și monitorizare a sesiunilor de control 14](#_Toc139559525)

[4. Metode folosite 16](#_Toc139559526)

[4.1 Proiectarea bazei de date 16](#_Toc139559527)

[4.2 Module generale 17](#_Toc139559528)

[4.3 Deployment 18](#_Toc139559529)

[4.4 Tehnologii folosite 19](#_Toc139559530)

[Python 19](#_Toc139559531)

[Golang 19](#_Toc139559532)

[Ffmpeg 20](#_Toc139559533)

[X Window System (X11) 20](#_Toc139559534)

[PySide6 20](#_Toc139559535)

[Apache Kafka 21](#_Toc139559536)

[MySql 21](#_Toc139559537)

[Docker 22](#_Toc139559538)

[4.5 Descrierea modulelor 22](#_Toc139559539)

[I. Directorul Client 22](#_Toc139559540)

[II. Directorul CloudMicroservices 35](#_Toc139559541)

[5. Rezultate obținute și analiza datelor 41](#_Toc139559542)

[5.1. Hardware: 41](#_Toc139559543)

[5.2 Software: 41](#_Toc139559544)

[5.3 Incarcare resurse pe microservicii: 41](#_Toc139559545)

[5.3.1 AggregatorMicroservice: 41](#_Toc139559546)

[5.3.2 AudioMicroservice: 42](#_Toc139559547)

[5.3.3 Client: 42](#_Toc139559548)

[5.3.4 InputExecutorMicroservice: 42](#_Toc139559549)

[5.3.5 VideoMicroservice: 43](#_Toc139559550)

[5.3.6 MergerMicroservice: 43](#_Toc139559551)

[5.3.7 KafkaMicroservice 43](#_Toc139559552)

[6. Concluzii 44](#_Toc139559553)

[7. Bibliografie 45](#_Toc139559554)

# Introducere

În contextul trecerii din ce în ce mai accelerate la societatea informațională se observa o trecere graduală de la munca la servici (eng. ,,on site”) la cea efectuată de acasă. Interoperarea la distanță utilizând o interfata grafica (eng. ,,remote desktop”) cu un sistem a devenit o tehnică obișnuită de a controla un computer folosind alt dispozitiv prin intermediul internetului, este popular în arii precum munca la distanță la distanță (telemunca), asistența de la distanță, administrare de la distanță.

Această nevoie crește cu cât calculul efectuat în nor este folosit în masa [1] și telemunca devine din ce în ce mai populară. Potrivit [2], aproximativ 3,7 de milioane de angajați muncesc de acasă cel puțin jumătate din timp.

Diversitatea și costul redus al soluțiilor software precum TeamViewer [3] sau Anydesk [4] au contribuit enorm si ele la popularitatea acestei tehnologii. În contextul infrastructurei ca servicu în nor, administratorii folosesc aplicațiile de control la distanță pentru a gestiona o mulțime de computere, în timp ce utilizatorii le folosesc pentru a-și accesa resursele și a încarca propriile aplicații.

O aplicație tipică de administrare de la distantă a computerelor capturează evenimentele mouseului si ale tastaturii tastaturii și le trimite către computer. Dispozitivul aflat la distanta executa comenzile și apoi va trimite către client starea ecranului.

Lucrarea de fata prezinta dezvoltarea unei aplicații pentru controlul de la distanța a computerelor în vederea ușurării administrării acestor dispozitive simplu și rapid, oriunde și oricând. Fluxul de lucru al aplicației începe prin posibilitatea de a controla orice computer de la distanță. Mergând pe exemplul cu telemunca menționat anterior, presupunem că angajatorul se află la computerul de muncă în interiorul sediului în timp ce tu ești în confortul casei în fața computerului personal. În această ordine de idei atât tu cât și angajatorul trebuie să fiți conectați pe conturile aferente.

Primul pas este făcut de către angajator, acesta trebuie să vă comunice atât identificatorul cât și parola de sesiune și să pornească partajarea. Apoi te vei putea alătura sesiunii cu ajutorul datelor primite. În timpul sesiunii respective ești liber să controlezi cursorul și să folosești propria tastatură pentru a utiliza computerul de la sediu.

Sesiunea poate fi oprita oricând de către angajator, caz în care vei fi scos din aceasta. Desigur și tu poți părăsi sesiunea la nevoie și reveni la o dată ulterioară, dacă aceasta nu a fost deja închisă de către angajator.

Odată terminată o sesiune va începe procedura de salvare a acestuia sub formă de videoclip.

Toți participanții care au asistat la sesiunea respectivă vor putea vedea și descarcă videoclipul aferent acesteia.

# Documentarea problemei

Primii pași în dezvoltarea aplicațiilor de administrare la distanță au fost făcuți în anii ‘60, când companiile au început să utilizeze mainframe-uri pentru prelucrarea datelor. Pentru a permite utilizatorilor să acceseze aceste computere de la distanță, s-au dezvoltat primele aplicații de acces la distanță, care permiteau utilizatorilor să se conecteze la mainframe-uri utilizând terminale de la distanță. [5]

În anii ‘70, odată cu apariția calculatoarelor personale, a apărut și nevoia de a accesa și controla aceste calculatoare de la distanță. Primele aplicații de acces la distanță pentru calculatoare personale au fost dezvoltate în această perioadă. Aceste aplicații au permis utilizatorilor să acceseze și să controleze calculatoarele de la distanță, totuși încă sub forma unor terminale asemănător anilor ‘60. [6]

În anii ‘80, odată cu creșterea numărului de computere în rețele de calculatoare, au apărut primele soluții de administrare a rețelelor de calculatoare de la distanță. Aceste soluții au permis administratorilor de rețea să acceseze și să gestioneze computerele și dispozitivele din rețea de la distanță. [6]

În anii ‘90, odată cu popularizarea internetului și a conexiunilor de mare viteză, aplicațiile de acces la distanță au devenit mai ușor de utilizat și mai accesibile. În această perioadă, au apărut aplicații precum PC Anywhere dezvoltat de Symantec și LapLink, care au permis utilizatorilor să acceseze și să controleze calculatoarele de la distanță prin intermediul unei conexiuni internet. [7]

În anii 2000, odată cu apariția tehnologiei cloud și a aplicațiilor web, aplicațiile de acces la distanță au devenit și mai ușor de utilizat și mai accesibile. Acest lucru a permis utilizatorilor să acceseze și să controleze computerele de la distanță prin intermediul unui browser web, fără a fi necesare aplicații specifice de desktop. Exemple de astfel de aplicații au fost, printre altele, și Virtual Network Computing (VNC) [8] și Remote Desktop Protocol (RDP) [9].

În prezent, există o gamă largă de aplicații de administrare a computerelor de la distanță, care utilizează o varietate de tehnologii, inclusiv conexiuni criptate, transfer de fișiere, colaborare în timp real și accesibilitate mobilă. Aceste aplicații sunt utilizate de persoane și companii din întreaga lume pentru a gestiona computerele și dispozitivele de la distanță, pentru a oferi suport tehnic și pentru a lucra în colaborare cu colegii de echipă.

## 2.1 TeamViewer

### I. Prezentare generală

Una dintre cele mai populare aplicații de acest gen, TeamViewer este usor de folosit, oferă o asistentă foarte buna și funcționează foarte bine în administarea de la distanța a dispozitivelor. Cu funcții precum transfer de fișiere, conversatii audio-video, partajarea ecranului, accesarea terminalului de la distanță, trezirea și repornirea computerului, aceasta aplicatie este cea mai completa si versatila aplicație în acest domeniu.

### II. Tehnologii folosite

Modul prin care aplicația face conexiunea între doi utilizatori poate fi găsit în declarația lor de securitate: ,,When establishing a session, TeamViewer determines the optimal type of connection. After the handshake through our master servers, a direct connection via UDP or TCP is established in 70% of all cases (even behind standard gateways, NATs and firewalls). The rest of the connections are routed through our highly redundant router network via TCP or https-tunnelling. You do not have to open any ports in order to work with TeamViewer! As later described in the paragraph "Encryption and Authentication," not even we, as the operators of the routing servers, can read the encrypted data traffic.”. [10]

Astfel putem distinge următorii pași în inițierea unei conexiuni între două computere:

1. Se creează o conexiune pe portul 80 pentru a nu avea probleme cu firewallurile sau NAT urile, pentru a aduna informațiile necesare.
2. Cu detaliile adunate, se folosește probabil o tehnică precum ,,Hole Punching” pentru a se face conexiunea directă între cele două computere, sau dacă acest lucru nu este posibil, se utilizează conexiunea cu serverul lor central.

Pentru compresia și transmiterea video TeamViewer folosește o tehnologie proprie, numită TeamViewer Compression & Fast Data Transmission Engine, care comprimă și optimizează transmisia video pentru a minimiza latenta si a crește performanța. Fast Data Transmission Engine combină transferul de fișiere și video cu o criptare puternică pentru a proteja datele transmise. Limbajul de programare predominant în crearea întregii arhitecturi este folosit C++.

### III. Argumente pro

* Ușurință în folosire
* O gama largă de funcționalități
* Gratuită pentru uzul personal
* Crearea unui cont nu este obligatorie.

### IV. Argumente contra

* Costisitor, cu un preț începând de la 49$/lună.
* Sesiuni instabile la conexiuni mai slabe.

## 2.2 Anydesk

### I. Prezentare generală

Anydesk este altă aplicație care își propune să rezolve problema interacționării la distanță. Nu beneficiază de o popularitate așa de mare precum TeamViewer, însă nu este de ignorat nici aceasta. Are o conexiune mai rapidă și stabilă care este obținută prin propriul algoritm de compresie și decompresie, în special în conexiunile cu lățime de bandă mică.

De asemenea când vine vorba de securitate AnyDesk oferă criptare end-to-end și autentificare în doi pași. Pe partea de specificații, fată de TeamViewer, Anydesk este mai lightweight, poate să ruleze fără probleme pe mașinile cu mai puțină memorie și putere de procesare. Nu în ultimul rând, costul pentru uzul comercial este mai mic față de TeamViewer, în timp ce pentru uzul personal este și el gratis.

### II. Tehnologii folosite

AnyDesk folosește o combinație de tehnologii pentru compresia video, securitatea și transmisia datelor. Compresia video este una de tip H.264, care este un standard de compresie video foarte eficient și larg utilizat. Acest algoritm reduce dimensiunea fișierelor video, fără a compromite semnificativ calitatea imaginii. Securitatea se bazează pe tehnologii de criptare puternice pentru a asigura securitatea transmisiei datelor. Utilizează un protocol de criptare numit TLS 1.2 pentru autentificare și schimbul de chei, iar apoi folosește criptare simetrică AES 256-bit pentru a proteja traficul datelor în timpul sesiunii de control la distanță [15]. Transmisia datelor folosește un protocol propriu numit DeskRT pentru transmisia datelor, care este optimizat pentru performanță și latență redusă. Acest protocol utilizează tehnici de comprimare și deține algoritmi de control al congestionării pentru a asigura o experiență fluidă și rapidă în timpul controlului la distanță [16].

Aplicatia include și o funcționalitate de transfer de fișiere între calculatoarele conectate. Acest transfer se bazează pe protocolul propriu al AnyDesk, care este optimizat pentru transmiterea eficientă și sigură a fișierelor între dispozitive.[16]

### III. Argumente pro

* Conexiune
* Securitate solidă
* Cerințe de sistem scăzute

### IV. Argumente contra

* Lipsa functionalitatilor de tipul de înregistrare a sesiunilor sau transferul de fișiere
* Nu este compatibil cu versiunile mai vechi de sisteme de operare

## 2.3 LogMeIn

### I.Prezentare generală

LogMeIn este o platformă de control la distanță care oferă utilizatorilor acces și control asupra computerelor de la distanță prin intermediul unei conexiuni securizate. Această soluție permite utilizatorilor să acceseze și să gestioneze resursele computerelor lor sau ale altor computere de pe orice locație și dispozitiv. Indiferent dacă este vorba de accesarea desktopului, de utilizarea aplicațiilor sau de transferul de fișiere, LogMeIn oferă o soluție flexibilă și convenabilă pentru controlul la distanță.

### II. Tehnologii folosite

LogMeIn se bazează pe o serie de tehnologii avansate pentru a oferi un control la distanță sigur și eficient. Acestea include SSL/TLS pentru a securiza transmisia datelor între dispozitivul local și computerul de la distanță. Această criptare asigură confidențialitatea și integritatea datelor transmise, protejându-le împotriva accesului neautorizat. LogMeIn oferă opțiuni puternice de autentificare și securitate pentru a proteja accesul la computerul de la distanță. Utilizatorii pot beneficia de autentificarea în doi pași, gestionarea permisiunilor de utilizator și restricțiile de acces, precum și posibilitatea de a crea parole unice și complexe pentru fiecare sesiune de control la distanță. [17]

### III. Argumente pro

* Interfață intuitivă și ușor de utilizat: Platforma LogMeIn oferă o interfață prietenoasă și intuitivă, care facilitează controlul la distanță pentru utilizatorii de toate nivelurile de experiență
* Funcționalitate extinsă: În plus față de controlul la distanță, LogMeIn oferă și alte funcționalități valoroase, cum ar fi transferul de fișiere, chat-ul integrat, colaborarea în timp real și administrarea utilizatorilor și a permisiunilor
* Securitate puternică: Cu criptarea SSL/TLS și opțiunile avansate de autentificare, LogMeIn oferă un nivel ridicat de securitate pentru a proteja datele și accesul la computerul de la distanță.

### IV. Argumente contra

* Anumite funcționalități avansate pot fi disponibile numai în planurile de abonament superioare, ceea ce poate implica costuri suplimentare.
* Performanța și responsivitatea pot varia în funcție de calitatea conexiunii la internet și de lățimea de bandă disponibilă.

# Abordare teoretică

## 3.1 Utilitatea aplicațiilor de control la distantă

Aplicațiile de control la distanță oferă oportunități nelimitate pentru a monitoriza și controla dispozitive și sisteme de la distanță, sporind eficiența și comoditatea într-o varietate de domenii. Iată câteva exemple de utilizări potențiale pentru o astfel de aplicație:

**Monitorizarea** – O astfel de aplicație poate fi folosită pentru a monitoriza activitățile de pe computerul unui utilizator. Aceasta poate fi utilă pentru a monitoriza angajații și pentru a evita utilizarea neautorizată a computerelor. De exemplu, un angajator poate utiliza această aplicație pentru a monitoriza utilizarea computerelor de către angajați în timpul orelor de lucru.

**Gaming** -Unele aplicații de control la distanță permit utilizatorilor să controleze un computer de la distanță și să joace jocuri. Acest lucru poate fi util pentru a juca jocuri care necesită o putere mare de procesare sau pentru a juca jocuri care nu sunt disponibile pe dispozitivele mobile. De exemplu, un jucător poate controla un computer puternic de la distanță pentru a juca jocuri cu cerințe ridicate de hardware sau pentru a juca jocuri care nu sunt disponibile pe dispozitivele mobile.

**Asistență tehnică** - O aplicație de control la distanță poate fi utilizată pentru a oferi asistență tehnică la distanță pentru problemele de pe computerul unui utilizator. Această abordare permite unui specialist să acceseze și să controleze computerul utilizatorului pentru a remedia problemele. De exemplu, dacă un utilizator întâmpină probleme cu instalarea unui program, poate invita un specialist să controleze computerul său prin intermediul aplicației de control la distanță pentru a remedia problema.

**Predare la distanță** -O aplicație de control la distanță poate fi utilizată pentru a oferi instruire la distanță, predare și prezentări. Prezentatorul poate controla computerul gazdă pentru a prezenta diapozitive sau pentru a efectua demonstrații pentru publicul său. De exemplu, profesorii pot utiliza această aplicație pentru a preda studenților de la distanță, prezentatorii pot utiliza această aplicație pentru a organiza webinarii sau prezentări virtuale.

**Accesarea computerului de la distanță** -O aplicație de control la distanță poate fi folosită pentru a accesa fișiere, programe și setări de pe un computer la distanță. Acest lucru este util pentru accesarea documentelor personale sau profesionale de pe un alt computer sau pentru a controla un computer de la distanță. De exemplu, dacă utilizatorul se află într-o călătorie de afaceri și are nevoie să acceseze un document de pe computerul său de acasă, poate utiliza o astfel de aplicație pentru a accesa computerul de la distanță și a descărca documentul.

## 3.2 Scenarii de utilizare

O diagramă de use case (fig 3.2.1) este o diagramă în UML (Unified Modeling Language) care modelează interacțiunile între actori și sistemul software într-un mod intuitiv și accesibil. Scopul principal al unei diagrame de use case este de a documenta și comunica comportamentul funcțional al unui sistem într-un format ușor de înțeles de către toate părțile interesate.

Elementele cheie ale unei astfel diagrame de use case sunt actorii(ex. User loged in), cazurile de utilizare (ex. Download video), și relațiile dintre aceștia .

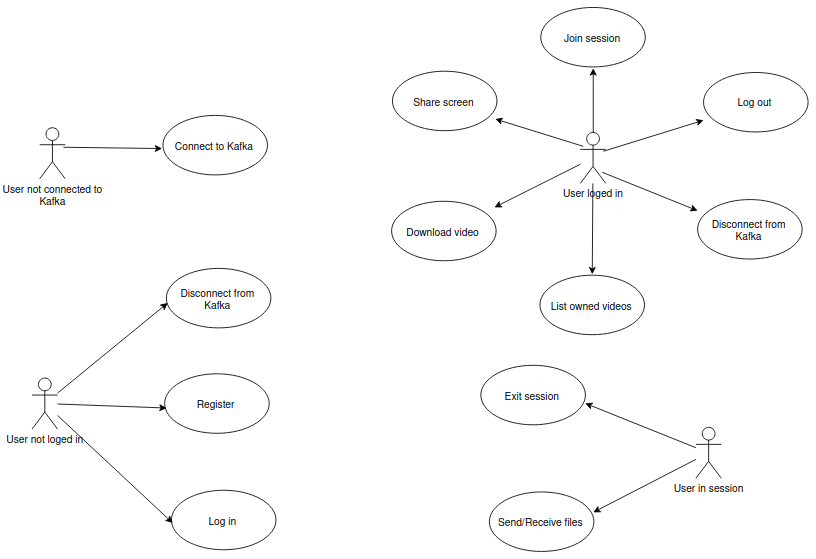


Fig 3.2.1

### 1. Gestionare conexiune cu Kafka

1. **Conectare la brokerul Kafka**

**Actor**

Utilizator

**Precondiții**

Utilizatorul nu este conectat la brokerul Kafka si dorește să se conecteze la acesta.

**Scenariu**

Utilizatorul accesează secțiunea ,,Kafka”, introduce adresa IP a brokerului Kafka în câmpul de input corespuzator, apoi apasă buton ,,Connect”. Aplicația verifică dacă adresa IP introdusă este validă și în caz afirmativ utilizatorul primește notificare că s-a conectat cu success la brokerul Kafka. Utilizatorul poate începe să se autentifice în aplicație sau să își creeze un cont nou(fig 3.2.2).

**Excepții**

Dacă adresa IP introdusă nu este validă sau nu poate fi accesibilă, aplicația afișează un mesaj de eroare și solicită introducerea unei adrese IP valide.

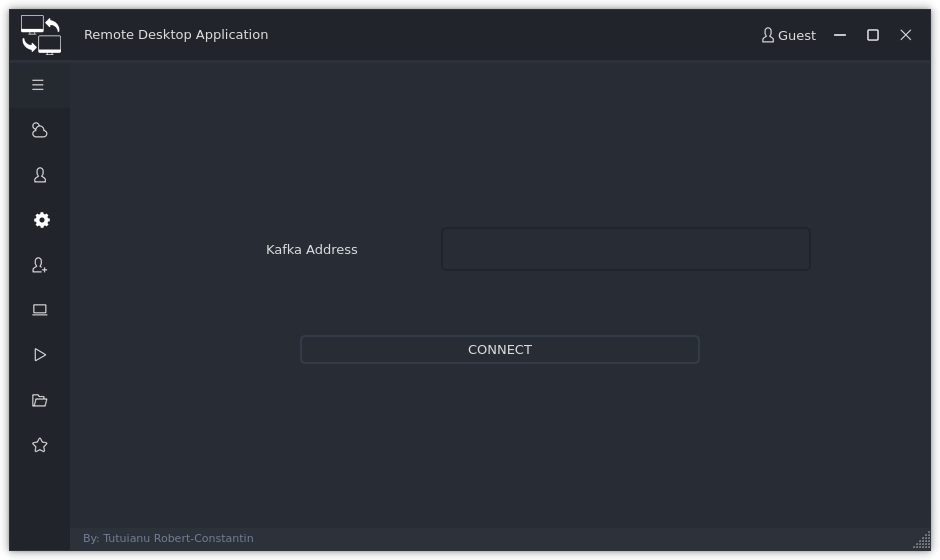


Fig 3.2.2

1. **Deconectare de la brokerul Kafka**

**Actor**

Utilizator.

**Precondiții**

Utilizatorul este conectat la brokerul Kafka prin intermediul aplicației și dorește să se deconecteze.

**Scenariu**

Utilizatorul accesează secțiunea ,,Kafka”, apasă butonul ,,Disconnect” pentru a se deconecta de la brokerul Kafka. Aplicația încheie conexiunea cu brokerul și dezactivează funcționalitățile legate de Kafka.(fig 3.2.3)

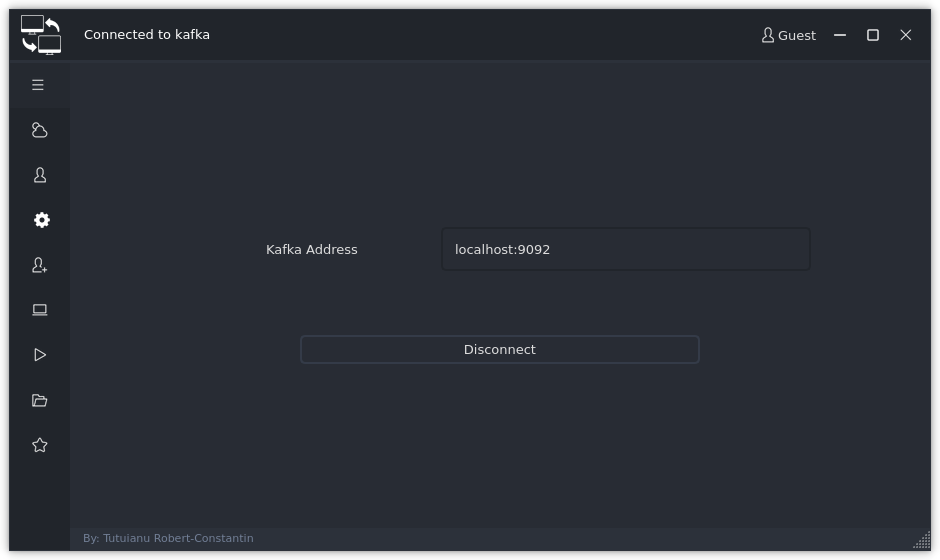


Fig 3.2.3

### 2. Autentificare și gestionare cont

1. **Autentificare utilizator**

**Actor**

Utilizator

**Precondiții**

Utilizatorul este conectat la brokerul Kafka și are un cont valid în aplicație.

**Scenariu**

Utilizatorul deschide aplicația și selectează secțiunea ,,Login”, introduce datele sale (nume de utilizator și parolă), iar mai apoi apasă butonul ,,Login” (fig 3.2.4). Sistemul verifică informațiile introduse și autentifică utilizatorul.

**Excepții**

Dacă credențialele introduse sunt incorecte, sistemul afișează un mesaj de eroare și utilizatorul este invitat să reintroducă informațiile de autentificare.

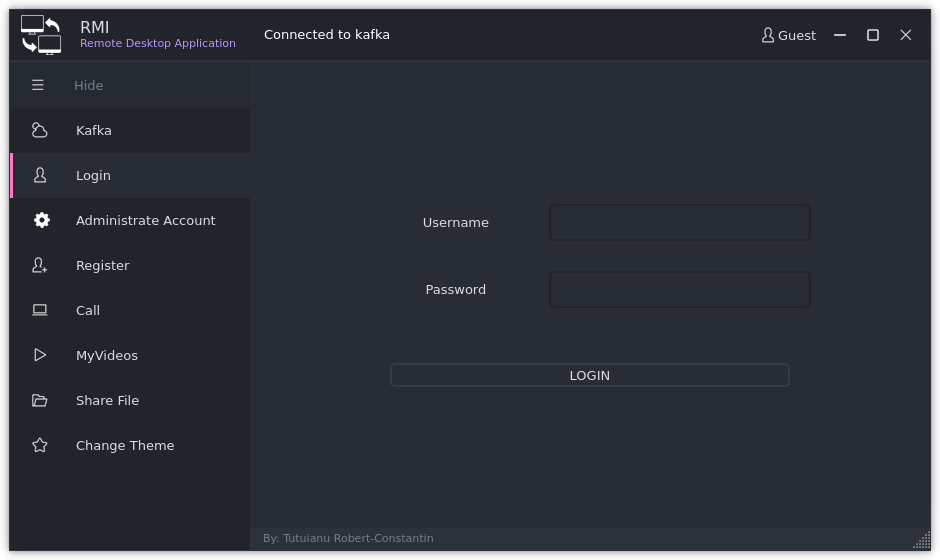


Fig 3.2.4

1. **Creare cont**

**Actor**

Utilizator

**Precondiții**

Utilizatorul este conectat la brokerul Kafka și dorește să își creeze un cont în aplicație.

**Scenariu**

Utilizatorul deschide aplicația, accesează secțiunea ,,Register” din meniul din partea stângă, introduce datele necesare creării unui cont nou (nume de utilizator, parolă, confirmare parolă) și apasă butonul ,,Register” (fig 3.2.5). Aplicația va verifica datele introduse iar mai apoi creează contul.

**Excepții**

Numele de utilizator exista deja în baza de date.

Parola și confirmarea acesteia nu coincid.

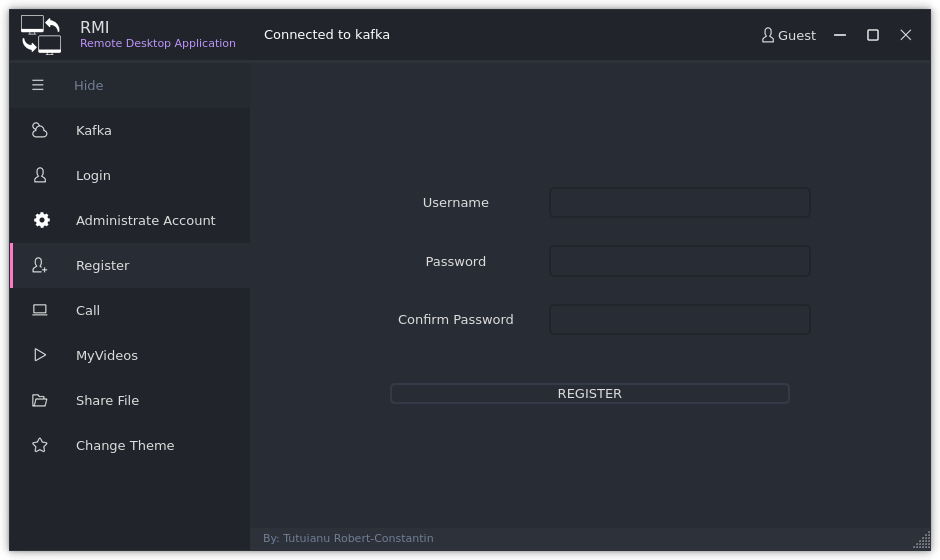


Fig 3.2.5

1. **Schimbare parola**

**Actor**

Utilizator

**Precondiții**

Utilizatorul este conectat la brokerul Kafka și are un cont valid în aplicație.

**Scenariu**

Utilizatorul deschide aplicația, accesează secțiunea ,,Account management” din meniul din partea stângă, introduce datele aferente schimbării parolei (parola veche, parolă nouă , confirmare parolă nouă) și apasă butonul ,,Change” (fig 3.2.6). Aplicația va verifica datele introduse iar mai apoi va face setările actuale.

**Excepții**

Parola nouă și confirmarea acesteia nu coincid.

Parola veche nu concide cu ce introdusă.

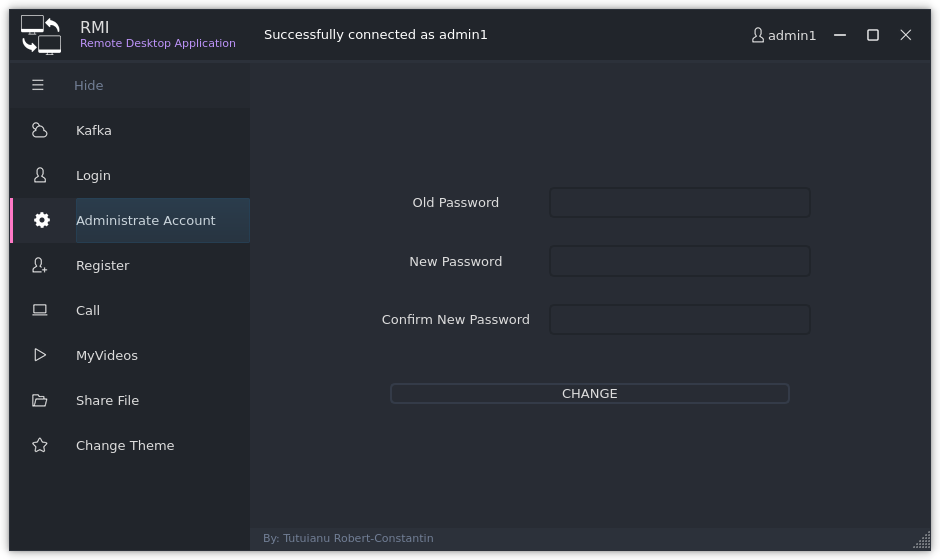


Fig 3.2.6

1. **Deconectare**

**Actor**

Utilizator.

**Precondiții**

Utilizatorul este conectat la brokerul Kafka și autentificat și dorește să se deconecteze de la acesta.

**Scenariu**

Utilizatorul accesează secțiunea ,,Login”, apasă butonul ,,Disconnect” pentru a se deconecta de la contul acestuia (fig 3.2.7). Utilizatorul primește o notificare cum că s-a deconectat cu succes de la contul sau, nemaiavand acces la funcționalitățile specifice contului pand când se autentifică din nou.

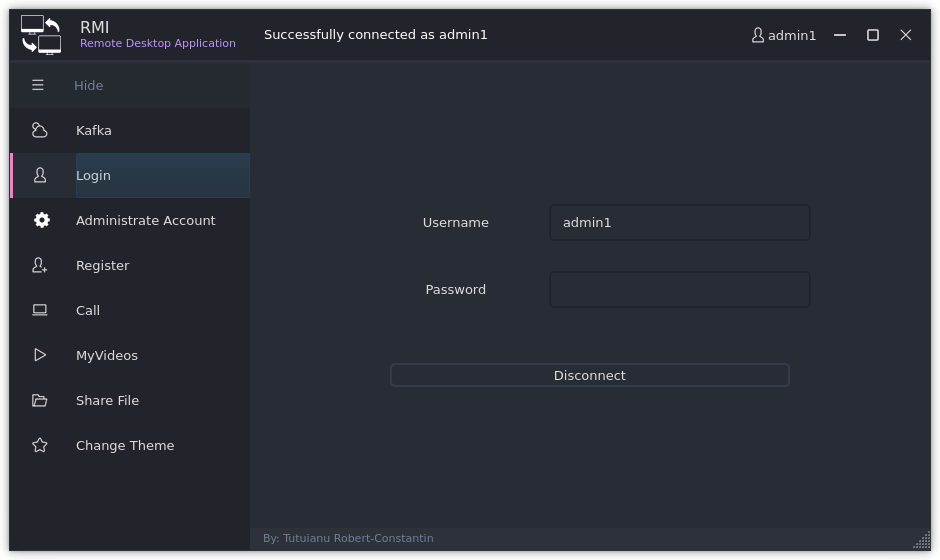


Fig 3.2.7

### 3. Control și monitorizare la distanță

1. **Participare la o sesiune de control la distanță**

**Actor**

Utilizator

**Precondiții**

Utilizatorul este autentificat în aplicație și există o sesiune activă.

Utilizatorul are cheia și parola de sesiune.

**Scenariu**

Utilizatorul accesează meniul ,,Remote Control”, introduce identificatorul și parola furnizate de partener (fig 3.2.8). Dacă partenerul are o sesiune activă, utilizatorul va fi redirecționat către o fereastră de control pentru a gestiona computerul acestuia.

**Excepții**

Dacă partenerul nu are o sesiune activă, utilizatorul primește o notificare corespunzătoare și nu se va deschide o fereastra de gestiune.

Cheia și parola nu se potrivesc cu nici o sesiune activa.

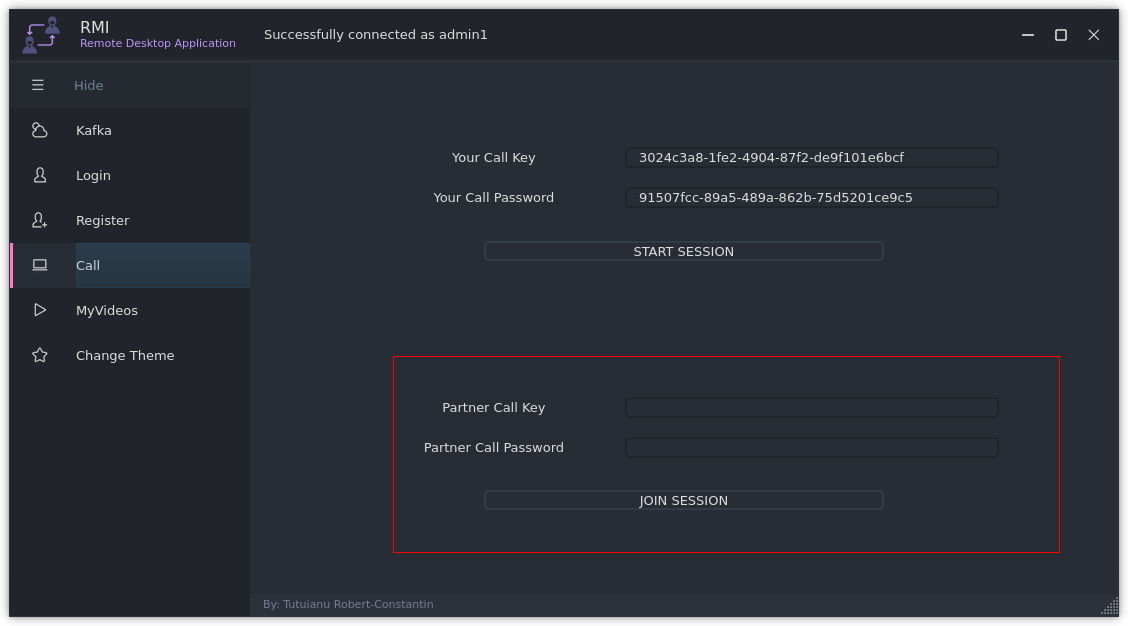


Fig 3.2.8

1. **Inițierea unei sesiuni de control la distanță**

**Actor**

Utilizator

**Precondiții**

Utilizatorul este autentificat în aplicație.

**Scenariu**

Utilizatorul accesează meniul "Remote Control", apasă butonul "Start Sharing" pentru a iniția o sesiune de control la distanță și comunica identificatorul și parola sesiunii partenerului cu care dorește să se conecteze. Partenerul poate introduce cheia și parola de sesiune pentru a se conecta. (fig 3.2.9)

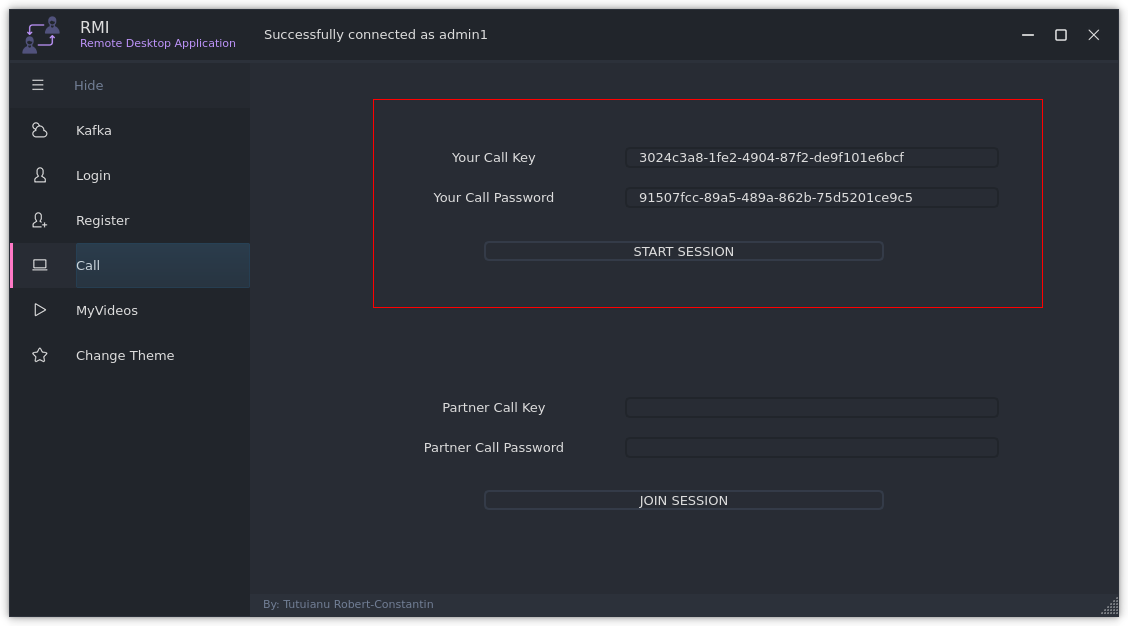


Fig 3.2.9

1. **Trimiterea de fișiere în cadrul sesiunii de control**

**Actor**

Utilizator

**Precondiții**

Utilizatorul este conectat la o sesiune de control și are un fișier pregătit pentru transfer.

**Scenariu**

Utilizatorul accesează meniul "File Transfer" în cadrul sesiunii de control, selectează fișierul dorit prin apăsarea butonului ,,Select File”. Calea către fișier va apărea în centrul ferestrei. Utilizatorul apasă mai apoi butonul ,,Upload” pentru a trimite fișierul către toți participanții la sesiunea de control (fig 3.2.10). Dacă participanții acceptă transferul, fișierul este livrat cu succes către toți utilizatorii conectați.

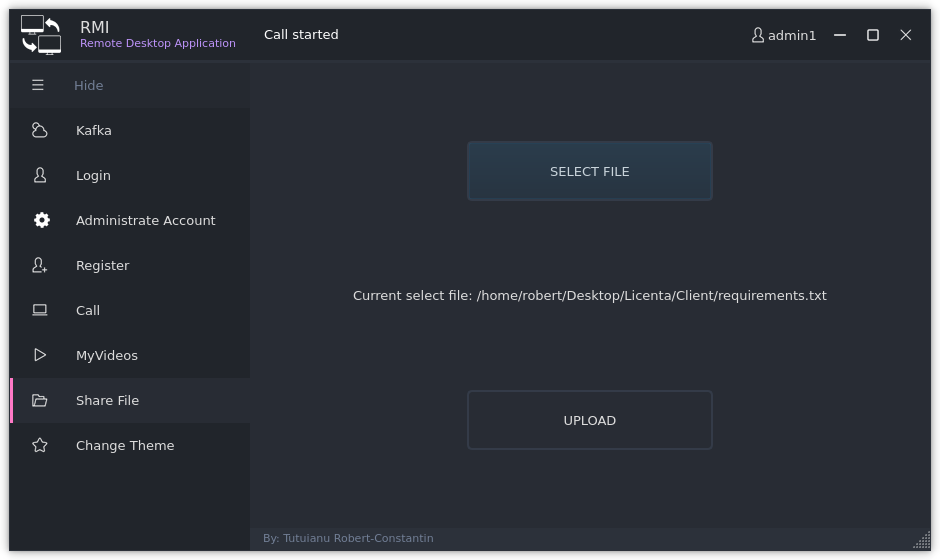


Fig 3.2.10

### 4. Administrare și monitorizare a sesiunilor de control

1. **Listare și gestionare a sesiunilor anterioare**

**Actor**

Utilizator

**Precondiții**

Utilizatorul este autentificat în aplicație și are dreptul de acces la sesiunile anterioare.

**Scenariu**

Utilizatorul accesează secțiunea "My Videos" pentru a vedea lista sesiunilor anterioare. Aplicația afișează o listă cu toate sesiunile la care utilizatorul a participat. Utilizatorul poate vizualiza informații suplimentare despre fiecare sesiune, cum ar fi data și ora desfășurării. Utilizatorul are opțiunea de a șterge sau descărca sesiunile anterioare, utilizând butoanele corespunzătoare. (fig 3.2.11)

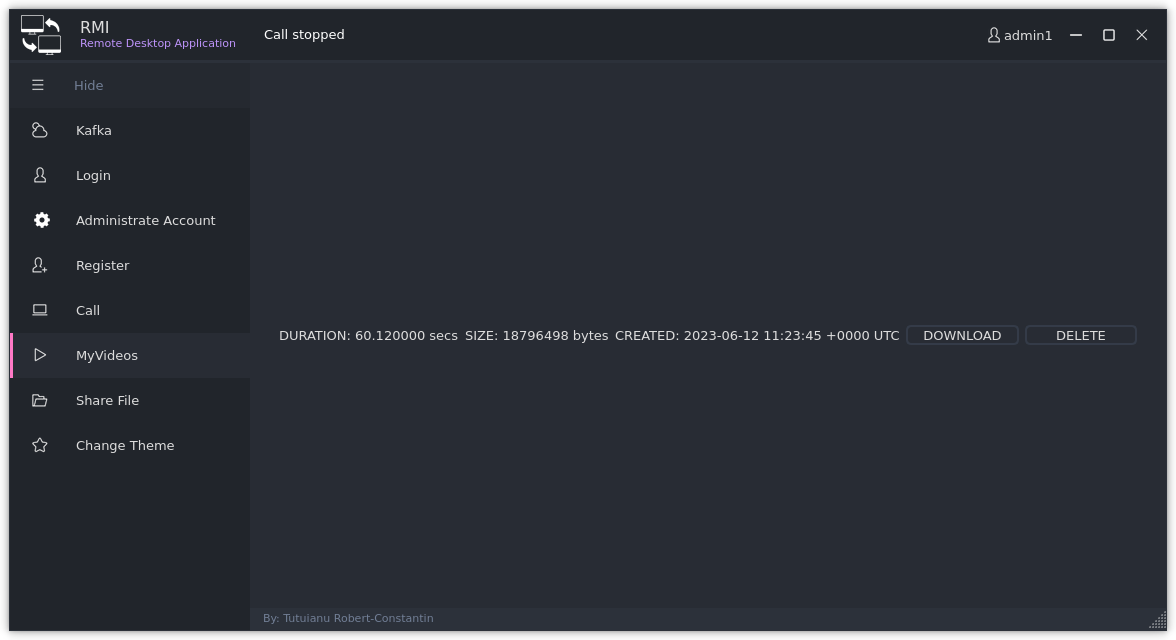


Fig 3.2.11

# Metode folosite

## 4.1 Proiectarea bazei de date

Capitolul "Proiectarea bazei de date" are ca scop prezentarea structurii și relațiilor dintre tabelele utilizate în aplicația de control la distanță. Baza de date este responsabilă pentru stocarea și gestionarea informațiilor legate de utilizatori, sesiuni și înregistrări video. Această proiectare a bazei de date asigură un sistem coerent și eficient de gestionare a datelor, facilitând funcționalitatea corectă și scalabilitatea aplicației.

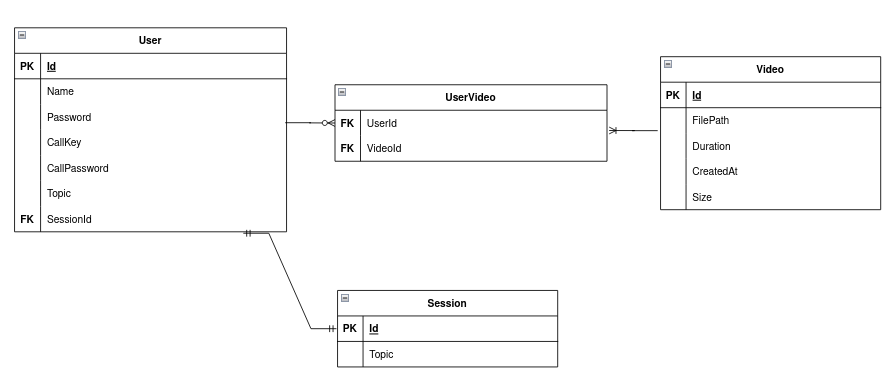


Fig 4.1.1

Tabela "User" (fig 4.1.1) reprezintă entitatea utilizatorului în aplicație. Aceasta conține informațiile necesare pentru identificarea și autentificarea utilizatorului. Fiecare înregistrare din tabel reprezintă un utilizator și este identificată unic prin intermediul câmpului "Id". Câmpurile tabelului "User" includ:

* "Id": Un identificator unic auto-incremental pentru fiecare înregistrare din tabel.
* "Name": Numele utilizatorului, care este un câmp obligatoriu și trebuie să fie unic.
* "Password": Parola utilizatorului pentru autentificare.
* "CallKey" și "CallPassword": Informații necesare pentru ca un partener să se conecteze la utilizator.
* "Topic": Topicul Kafka asociat utilizatorului, utilizat pentru transmiterea datelor.
* "SessionId": Cheie externă care indică sesiunea la care utilizatorul este conectat.

Tabela "Video" (fig 4.1.1) stochează informații despre videoclipurile salvate aferente sesiunilor. Fiecare înregistrare din tabel reprezintă un videoclip și este identificată unic prin intermediul câmpului "Id". Câmpurile tabelului "Video" includ:

* "Id": Un identificator unic auto-incremental pentru fiecare înregistrare din tabel.
* "FilePath": Calea către fișierul video pe computer, care permite găsirea și descărcarea ulterioară a videoclipului.
* "Duration": Durata videoclipului, exprimată în secunde.
* "CreatedAt": Data și ora creării videoclipului.
* "Size": Mărimea videoclipului pe disk.

Tabela "Session" (fig 4.1.1) reprezintă o sesiune activă în aplicație. Aceasta este creată atunci când un utilizator pornește o sesiune de control la distanță. Fiecare înregistrare din tabel reprezintă o sesiune și este identificată unic prin intermediul câmpului "Id". Câmpurile tabelului "Session" includ:

* "Id": Un identificator unic auto-incremental pentru fiecare înregistrare din tabel.
* "Topic": Topicul Kafka asociat sesiunii, care permite partenerului să asculte și să primească datele de ecran și să trimită intrările de la tastatură și mouse.

## 4.2 Module generale

Aplicația este compusă din mai multe module funcționale (fig 4.2.2) care lucrează împreună pentru a oferi o experiență fluidă și eficientă. Aceste module lucrează sinergic pentru a oferi utilizatorilor o experiență de control la distanță flexibilă, eficientă și securizată. Prin capturarea audio-video, înregistrarea comenzilor și gestionarea sesiunilor, aplicația permite utilizatorilor să controleze și să monitorizeze dispozitivele remote în timp real, facilitând colaborarea și accesul la resursele necesare pentru angajați sau studenți.

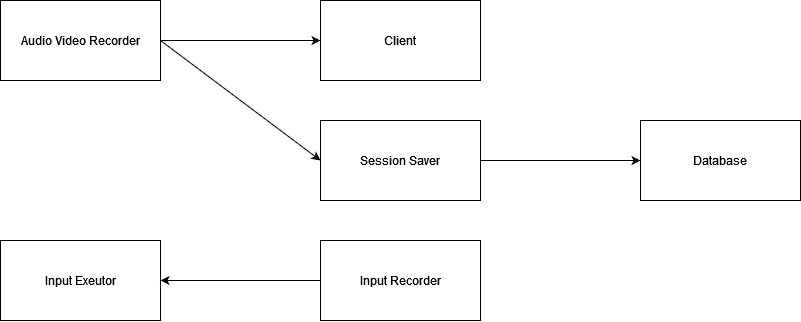


Fig 4.2.1

**AudioVideo Recorder** se ocupă cu înregistrarea și transmiterea ecranului și a microfonului utilizatorului. Acest modul permite capturarea și înregistrarea activității audio-video a dispozitivelor remote. Utilizatorii pot vedea și auzi tot ceea ce se întâmplă pe ecranul și în jurul dispozitivului remote în timp real.  
 Modulul **Client** reprezintă interfața prin care utilizatorii interacționează cu aplicația. Acesta oferă opțiuni și funcționalități pentru a controla și monitoriza dispozitivele remote, inclusiv conectarea la sesiuni, navigarea prin sesiuni salvate și transmiterea comenzilor de la tastatură și mouse.  
 **Session Saver** creează și salvează înregistrarea aferentă unei sesiuni în baza de date. Acest modul se ocupă de gestionarea sesiunilor de control la distanță. După finalizarea unei sesiuni, Session Saver creează și salvează înregistrarea sesiunii respective în baza de date, păstrând astfel un istoric al activității.  
 **Input Recorder** înregistrează comenzile primite de la mouse și tastatură. Acest modul monitorizează și înregistrează comenzile de la tastatură și mouse trimise de utilizatorul local. Comenzile înregistrate sunt apoi trimise către modulul Input Executor pentru a fi executate pe dispozitivul remote.  
 **Input Executor** execută comenzile primite de la Input Recorder. Acest modul primește comenzile înregistrate de la Input Recorder și le execută pe dispozitivul remote. Comenzile pot include acțiuni precum deplasarea cursorului, clicuri, taste apăsate și altele, permițând utilizatorului să controleze dispozitivul remote în mod eficient.  
 **Database** interacționează cu baza de date. Modulul Database este responsabil de interacțiunea cu baza de date a aplicației. Acesta gestionează stocarea și recuperarea datelor referitoare la sesiuni, înregistrări și alte informații relevante pentru funcționarea aplicației.

## 4.3 Deployment

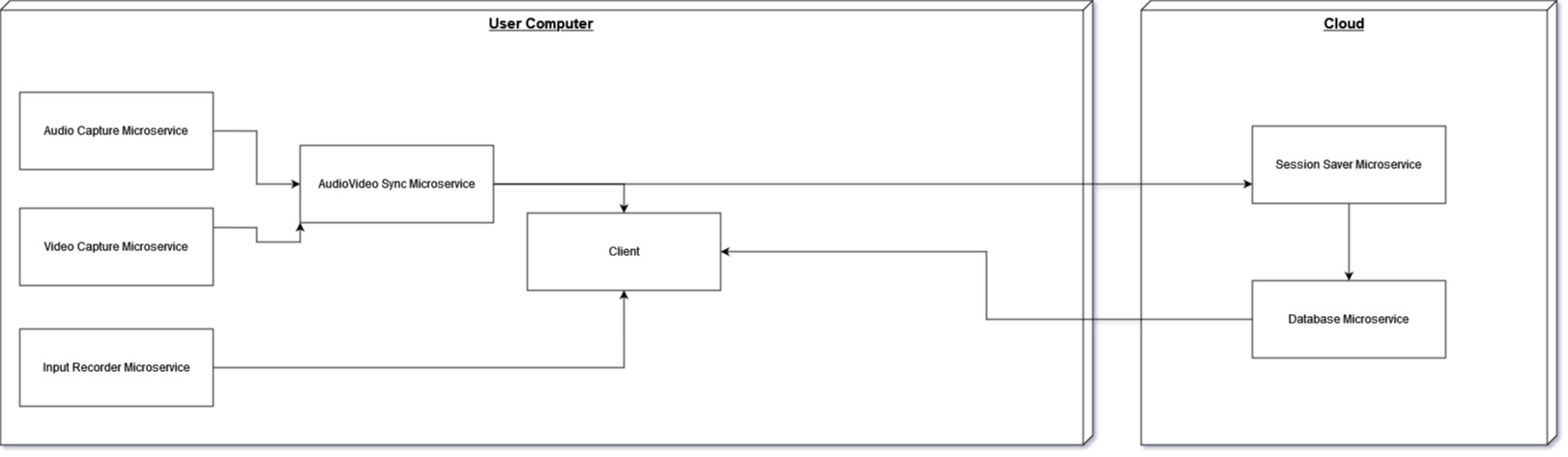


Fig 4.3.1 Diagrama de deployment

1. Video Capture Microservice -> instalat pe computerul utilizatorului întrucât are nevoie de acces la ecranul acestuia pentru a face capturile.
2. Audio Capture Microservice -> instalat pe computerul utilizatorului deoarece are nevoie de acces la porturile de intrare pentru a înregistra microfonul.
3. Input Recorder Microservice -> instalat pe computerul utilizatorului deoarece are nevoie de acces la perifericele de intrare pentru a captura comenzile primite de la mouse și tastatura.
4. AudioVideo Sync Microservice -> instalat pe computerul utilizatorului pentru a crește performanța. Nu are nevoie de acces direct la computer dar pentru a reduce latentă semnificativ, microserviciile de captură trimit acestuia doar căile de pe disk către fișierele care conțin conținutul video respectiv audio care să fie sincronizat.
5. Client -> instalat pe computerul utilizatorul pentru a pune la dispoziție interfața grafică.
6. Session Server Microservice -> instalat în nor deoarece nu are nevoie de acces la computerul utilizatorului. Toate videoclipurile care compun sesiunea sunt concatenate iar apoi salvate în baza de date.
7. Database Microservice -> instalat în nor deoarece nu are nevoie de acces la computerul utilizatorului. Acesta interacționează cu baza de date.

Calculul (captură, sincronizare și comprimare) se realizează predominant la client deoarece este ce mai eficientă metodă din punct de vedere al latentei și a cantității de date vehiculate. O captură de ecran la o rezoluție de 1920x1080 are aproximativ 8 MO înainte de a fi procesată. Pentru o experiență decentă, capturile ar trebui să fie făcute odată la 0.0333 secunde (30 pe secundă). Deci cea mai eficientă metoda de a putea trimite videoclipurile rapid este de a sincroniza și de a face compresia la nivel de client, înainte ca videoclipul să fie transmis.

## 4.4 Tehnologii folosite

### Python

Python este un limbaj de programare interpretat folosit pentru a creea aplicații web, a face scripturi de automatizare și de a analiza date. Este un limbaj general, însemnând că poate fi utilizat în o mulțime de domenii și sarcini, nefiind specializat pe o anumită nișa. Aceasta versatilitate, împreună cu ușurință de a fi învățat, l-a ajutat să devină unul dintre cele mai populare limbaje de la momentul actual.   
 Potrivit studiilor [13] și [14], făcute în februarie 2020 respectiv iunie 2021, Python era pe locul 2 în cele mai populare limbaje.   
Cele mai populare utilizare ale limbajului Python sunt:

* Analiza datelor și învățare automată.
* Automatizarea sarcinilor repetitive
* Dezvoltarea aplicațiilor web
* Testarea aplicațiilor software

Acesta este folosit în aplicație la dezvoltarea microserviciilor Audio și Merger dar și a clientului, prin librăria PySide6.

### Golang

Go este un limbaj de programare open-source, dezvoltat inițial de către Google, cu un accent puternic pe performanță, eficiență și gestionarea sarcinilor concurente. Go oferă o sintaxă simplă și ușor de înțeles, ceea ce îl face ideal pentru dezvoltatorii de toate nivelurile de experiență. Limbajul a fost proiectat să permită dezvoltarea de aplicații mari și complexe, cu un număr redus de erori.  
 Limbajul are un model de concurență unic, care se bazează pe canale și go-routines. Aceasta permite dezvoltatorilor să creeze aplicații care rulează mai rapid și mai eficient, fără a fi nevoie de gestionarea manuală a sarcinilor concurente. Go dispune, de asemenea, de o colecție largă de biblioteci standard, care simplifică dezvoltarea de aplicații, precum și o comunitate activă de dezvoltatori care oferă suport și îmbunătățesc continuu limbajul.  
 Acesta este proiectat să fie un limbaj de programare sigur și robust, cu o sintaxă strictă, gestionare automată a memoriei și evitare a vulnerabilităților de securitate comune, cum ar fi pointerii nuli. Go este un limbaj foarte util pentru dezvoltarea de aplicații critice din punct de vedere al securității, precum cele care implică procesarea datelor sensibile sau tranzacții financiare. [19].  
 In aplicatie limbajul este folosit în crearea microserviciilor care necesită performantă crescută, de exemplu micoserviciul care se ocupă cu înregistrarea ecranului prin efectuarea a multiple capturi ale acestuia pe secundă, la microserviciul de agregare care sincronizează și creează un singur fișier din din canalele video și audio.

### Ffmpeg

FFmpeg este un set de instrumente open-source pentru înregistrarea, convertirea și redarea fișierelor multimedia, cum ar fi video și audio. Este disponibil pentru o varietate de sisteme de operare, inclusiv Windows, Linux și macOS, și este folosit într-o gamă largă de aplicații, de la producția video până la redarea de conținut multimedia pe dispozitive mobile.  
 Cu ajutorul FFmpeg, utilizatorii pot converti fișiere multimedia între diferite formate și compresii, precum și să modifice diverse proprietăți, cum ar fi rezoluția, ratele de cadre și bitrate-ul. Această flexibilitate face posibilă manipularea fișierelor multimedia pentru o varietate de aplicații, cum ar fi redarea pe dispozitive mobile sau streaming-ul de conținut video pe internet.  
 FFmpeg este un instrument puternic și versatil, care vine cu o serie de filtre și efecte incluse, cum ar fi înclinarea, rotația și ajustarea culorilor. Aceste funcții permit utilizatorilor să îmbunătățească și să optimizeze fișierele multimedia, astfel încât acestea să fie cât mai atractive pentru publicul lor. În plus, utilizatorii avansați pot personaliza și extinde funcționalitatea FFmpeg prin intermediul unor plugin-uri și scripturi personalizate. [20]  
 Acesta este folosit direct la îmbinarea și compresia videoclipului audio și cel video dar și în mod indirect în decodarea fișierelor la client, pentru a separa canalele audio și video.

### X Window System (X11)

X Window System, cunoscut și sub numele de X11, este un sistem de afișare grafică open-source, care oferă o platformă standardizată pentru dezvoltarea aplicațiilor grafice pe sisteme de operare Unix și Linux. X11 a fost dezvoltat inițial în anii '80 și este încă utilizat pe scară largă astăzi.  
 X11 oferă o arhitectură client-server, în care aplicațiile grafice (numite și clienți X) rulează într-un proces client separat, iar sistemul de afișare grafică (server X) rulează pe sistemul de operare gazdă. Această arhitectură permite utilizatorilor să ruleze aplicații grafice de la distanță, ceea ce este util în medii de rețea și distribuite.  
 Prin intermediul X11, utilizatorii pot crea interfețe grafice complexe și personalizate pentru aplicațiile lor, folosind o varietate de widget-uri și ferestre. X11 oferă, de asemenea, suport pentru o varietate de dispozitive de intrare, cum ar fi tastatura, mouse-ul și dispozitivele touchscreen, precum și pentru afișaje multiple și afișaje cu rezoluții diferite. Aceste funcții permit utilizatorilor să creeze aplicații grafice de înaltă calitate și eficiente din punct de vedere al resurselor.[12]  
 Sistemul este folosit pentru captura efectivă a ecranului, prin multiple apeluri la biblioteca xcb din limbajul C.

### PySide6

PySide6 este o bibliotecă Python puternică și populară, care oferă facilități pentru dezvoltarea de aplicații desktop cu interfață grafică. Similar cu PyQt5, PySide6 este construită pe baza funcționalităților oferite de Qt, un framework C++ extensibil și flexibil pentru dezvoltarea de aplicații cross-platform. PySide6 oferă posibilitatea de a crea aplicații cu o interfață utilizator modernă și atractivă, compatibile cu diverse sisteme de operare.  
 Biblioteca oferă o gamă largă de widgeturi și instrumente grafice pentru a crea interfețe flexibile și personalizate. Dezvoltatorii au acces la controalele de bază, cum ar fi butoane, casete de text și liste, dar și la widgeturi mai avansate, precum tab-uri, arbori de direcții, diagrame sau grafice. Această versatilitate îi permite dezvoltatorului să creeze interfețe bogate și interacțiuni complexe în aplicații.  
 Pe lângă interfața grafică, PySide6 oferă și funcționalități adiționale, precum accesul la baze de date, suport pentru multimedia, manipularea imaginilor și multe altele. Astfel, dezvoltatorii pot crea aplicații complete, care nu doar afișează informații și permit interacțiunea cu utilizatorul, ci și gestionează date, rulează procese în background și oferă funcționalități avansate.[21]  
 PySide6 este folosit la întreaga dezvoltarea a interfeței cu utilizatorul.

### Apache Kafka

Apache Kafka este o platformă open-source de streaming de date, care permite transmiterea în timp real a datelor între aplicații și sisteme. Este proiectat pentru a face față volumelor mari de date și pentru a asigura livrarea sigură a acestora. Kafka a fost inițial dezvoltat de către compania LinkedIn, iar acum este un proiect Apache Software Foundation.  
 Kafka utilizează o arhitectură de tip publish-subscribe, în care producătorii de date trimit mesaje la un set de topicuri, iar consumatorii se abonează la aceste topicuri pentru a primi mesajele respective. Această arhitectură permite utilizatorilor să transmită și să prelucreze datele în timp real, fără a fi nevoie să stocheze toate datele într-un singur loc.  
 O altă caracteristică importantă a Apache Kafka este scalabilitatea. Kafka poate fi scalat orizontal, adică se pot adăuga mai multe instanțe Kafka pentru a gestiona volumul crescut de date. Acest lucru permite utilizatorilor să facă față cu ușurință creșterii volumului de date și să asigure performanțe ridicate. [23]  
 Platforma este responsabilă de întreagă comunicație între microservicii fiind unicul canal de transmisie a informațiilor prezent în aplicație.

### MySql

MySQL este folosit la dezvoltarea bazei de date în aplicație fiind un sistem de gestionare a bazelor de date relaționale open-source, utilizat în mod obișnuit în dezvoltarea de aplicații web și mobile. Este dezvoltat și susținut de Oracle Corporation și este disponibil sub licența GPL (General Public License).  
 Acesta este un sistem de gestiune a bazelor de date relaționale (RDBMS) care utilizează limbajul de interogare structurat (SQL) pentru a efectua operațiile CRUD (Create, Read, Update, Delete) pe bazele de date. MySQL poate fi utilizat cu multe limbaje de programare, cum ar fi PHP, Python, Java și C#, prin intermediul driverelor de conectivitate disponibile.  
 Sistemul oferă o varietate de funcții, cum ar fi suportul pentru tranzacții, chei străine și indexare. Aceste caracteristici permit dezvoltatorilor să implementeze baze de date puternice, rapide și fiabile. În plus, MySQL este scalabil și poate gestiona volume mari de date. [23]

### Docker

Docker este o platformă de virtualizare de tip container care permite dezvoltatorilor să izoleze și să ruleze aplicații într-un mediu de producție, fără a afecta restul sistemului. Fiecare container Docker conține toate componentele necesare pentru a rula o aplicație, inclusiv codul sursă, bibliotecile și dependințele, ceea ce face ca distribuirea aplicațiilor să fie rapidă și ușoară.  
 Platforma se bazează pe tehnologia de containerizare, care permite izolarea componentelor unei aplicații într-un mediu de rulare independent. Acest lucru face posibilă rularea mai multor containere pe același sistem de operare, fără a se interfera între ele. Containerul este o entitate portabilă care poate fi distribuită pe orice sistem care rulează Docker, ceea ce face distribuirea și gestionarea aplicațiilor mult mai ușoară.  
 Acesta oferă și un sistem de gestiune a imaginilor, care permite dezvoltatorilor să creeze, să stocheze și să distribuie imagini Docker. Imaginile sunt utilizate pentru a crea containere și pot fi personalizate în funcție de necesitățile aplicației. Docker Hub este un registru public de imagini Docker care conține o varietate de imagini, cum ar fi cele pentru baze de date, servere web și multe altele. [24]  
 Prin intermediul platformei Docker microserviciile Database și Merger sunt încărcate în nor.

## 4.5 Descrierea modulelor

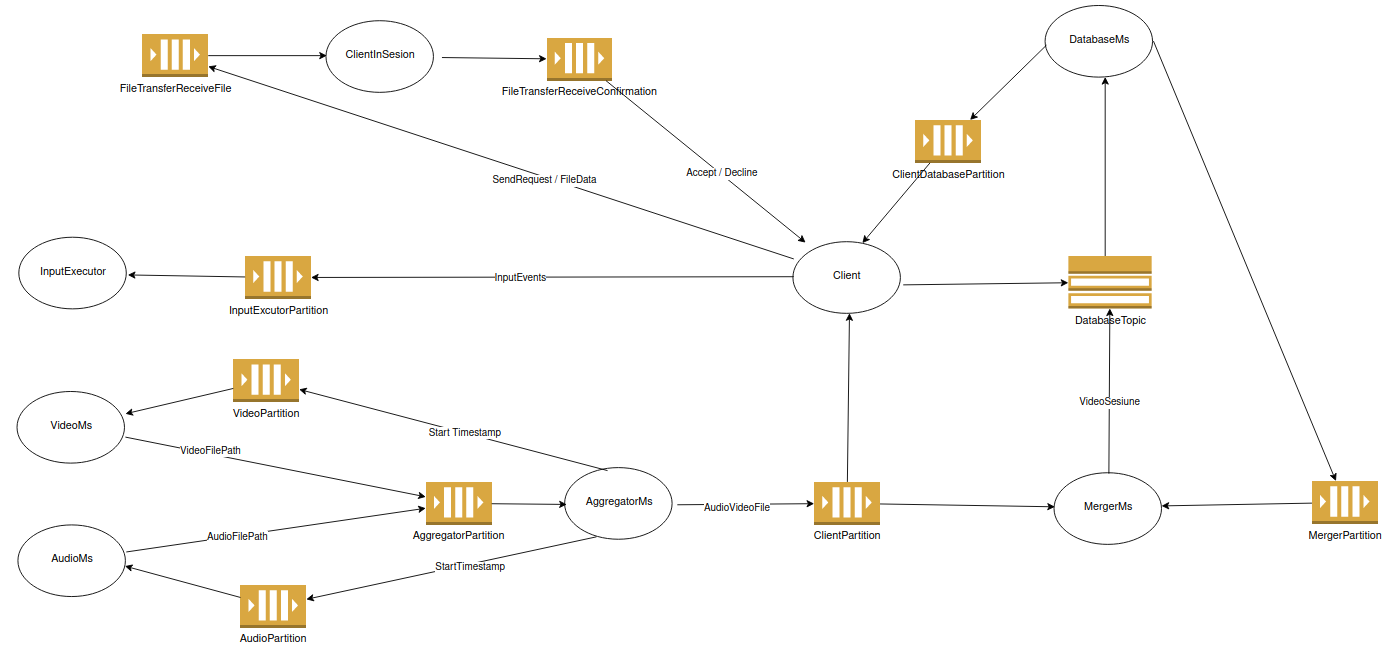


Fig 4.5.1

### Directorul Client

**Directorul client** conține elementele necesare pentru executarea atât a interfeței cu utilizatorul, cât și a componentelor indispensabile de care are nevoie aplicația pentru a începe partajarea resurselor proprii sau pentru a prelua controlul altui computer. Acesta este, la rândul lui, compus din următoarele pachete:

1. Pachetul Kafka - reprezinta stratul de abstractizare intre libraria cofluent\_kafka si microservicii.
2. VideoMicroservice - desemnat cu înregistrarea video a ecranului, folosit pentru partajarea resurselor proprii.
3. AudioMicroservice - Are rolul de a înregistra audio informația de la microfonul clientului, utilizat la partajarea resurselor proprii.
4. AggreagatorMicroservie - Îmbină informațiile (fișierele) primite de la microserviciile video și audio și este folosit la partajarea resurselor proprii
5. InputExecutorMicroservice - fiind folosit la partajarea resurselor proprii, acesta primește informațiile de la perifericele computerului partener, cum ar fi butoanele apăsate pe tastatură sau mouse, pentru a le putea executa pe computerul propriu
6. User Interface Client - este pachetul care pune la dispoziție interfața grafică cu utilizatorul.

**Pachetul Kafka**

1. **Python**

Clasa **CustomKafkaMessage** (fig. 4.5.2) moștenește clasa Message din biblioteca confluent\_kafka pentru a crea mesaje personalizate în cadrul clasei KafkaConsumerWrapper și a se integra cu obiectele de tip Message primite de la consumatorii Kafka implicit.  
 Clasa **KafkaProducerWrapper** (fig. 4.5.2) moștenește clasa Producer din confluent\_kafka. Sistemul Kafka este optimizat pentru a trimite mesaje de maxim 1 MO, astfel este nevoie de un obiect prin care putem trimite mesaje mai mari, de exemplu un fișier audio/video.  
 Clasa **KafkaConsumerWrapper** (fig. 4.5.2) este proiectată în vederea recepționării mesajelor personalizate trimise de un obiect de tipul KafkaProducerWrapper, în scopul primirii mesajelor mai mari de 1 MO.  
 Clasa **Partitions** (fig. 4.5.2) conține numărul de partiție la care fiecare microserviciu trebuie să trimită mesajele.

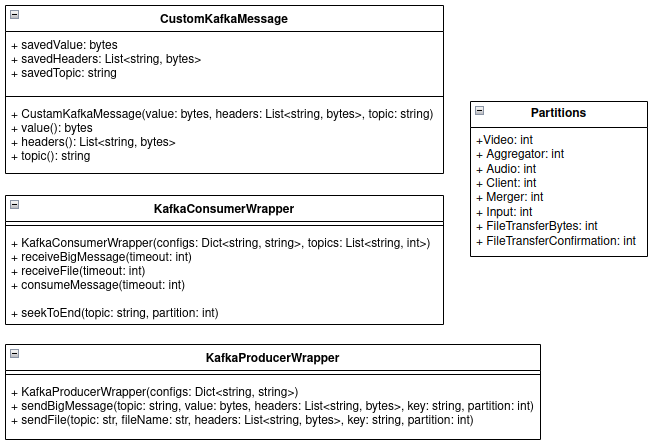


Fig 4.5.2

1. **Golang**

Clasa **Producer** (fig. 4.5.3) moștenește clasa Producer din confluent\_kafka. Sistemul Kafka este optimizat pentru a trimite mesaje de maxim 1 MO, astfel este nevoie de un obiect prin care putem trimite mesaje mai mari, de exemplu un fișier audio/video.  
 Clasa **Consumer** (fig. 4.5.3) este proiectată în vederea recepționării mesajelor personalizate trimise de un obiect de tipul KafkaProducerWrapper, în scopul primirii mesajelor mai mari de 1 MO.  
 Clasa **Partitions** (fig. 4.5.3) conține numărul de partiție la care fiecare microserviciu trebuie să trimită mesajele.

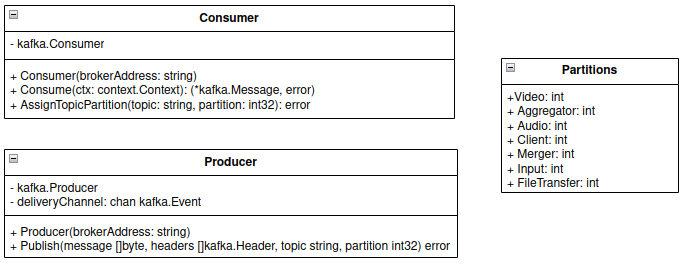


Fig 4.5.3

**Pachetul VideoMicroservice**

**VideoMicroservice**  este microserviciul care se ocupă de înregistrarea video a ecranului. Acesta este conceput să primească un mesaj de start de la AggregatorMicroservice, care să îi indice momentul exact în timp când trebuie să înceapă înregistrarea. De exemplu, mesajul de start poate să indice ora și data la care înregistrarea trebuie să înceapă. Odată ce primește acest mesaj, VideoMicroservice pornește obiectul de tip Recorder, care se ocupă de înregistrarea efectivă a video-ului. Recorder va capta tot ceea ce se întâmplă pe ecranul dispozitivului în acel moment, înregistrând în timp real. Pe măsură ce Recorder înregistrează, VideoMicroservice va capta de la canalul de ieșire al obiectului toate videoclipurile realizate. Acesta le va colecta și le va transmite mai departe la microserviciul de agregare.

Clasa **ByteImage** (fig. 4.5.4) este o clasă folosită pentru a reprezenta o imagine în format binar sau în format de octeți. Aceasta permite crearea, parsarea și modificarea imaginilor prin manipularea datelor binare care reprezintă imaginea. De asemenea, clasa poate fi folosită și pentru a comprima imaginile, ceea ce poate duce la reducerea dimensiunii fișierelor de imagine și a timpului de transfer al acestora prin rețea. Pentru a fi mai specific, clasa ByteImage este utilă în situațiile în care este necesară manipularea datelor brute ale imaginilor, cum ar fi extragerea unui anumit pixel sau grup de pixeli dintr-o imagine, conversia imaginii dintr-un format în altul sau aplicarea unor filtre de imagine pentru a îmbunătăți calitatea acesteia.

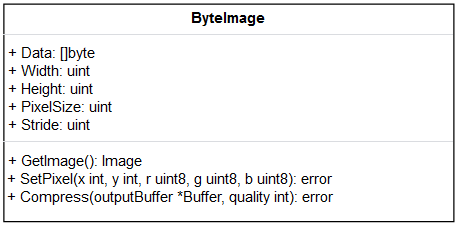


Fig 4.5.4

Clasa **Recorder** (fig. 4.5.5) reprezintă un element cheie în cadrul microserviciului de captură video, fiind utilizată pentru crearea și transmiterea fișierelor video. Această clasă este instanțiată și apelată prin intermediul metodei Start, care inițiază un proces complex de capturare și înregistrare a imaginilor. Procedura de funcționare a clasei începe cu calcularea intervalului de timp la care trebuie făcută o captură de ecran, astfel încât să se obțină un flux video continuu și de înaltă calitate. Apoi, prin intermediul unui loop de capturare a imaginilor, acestea sunt adăugate în coada imageBuffer pentru a fi procesate ulterior. În același timp, se creează un fișier video nou și se adaugă fiecare cadru din coada menționată anterior, astfel încât să se formeze un videoclip complet și funcțional. În final, calea către acest videoclip este adăugată în coada VideoBuffer, pentru a fi transmisă mai departe în cadrul microserviciului.

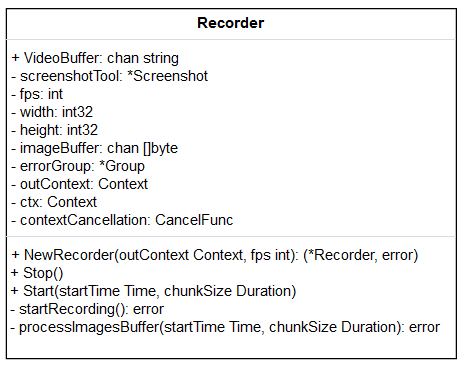


Fig 4.5.5

Clasa **Screen** (fig. 4.5.6) este o clasă utilitară utilizată pentru a captura ecranul computerului. Aceasta oferă o metodaa metoda care permite obținerea capturilor de ecran. Atunci când clasa Screen este inițializată, aceasta detectează automat dimensiunile ecranului și instantiază obiectul prin care se realizează copierea octeților care conțin informația stării ecranului la acel moment.. Atunci când este apelată metoda Get, aceasta copiază octeții din buffer și îi returnează sub forma unui obiect octet. Acești octeți pot fi utilizați apoi pentru a afișa captura de ecran sau pentru a fi procesați în alte moduri.

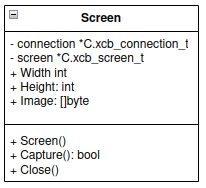


Fig 4.5.6

Clasa **Screenshot** (fig. 4.5.6) folosește clasa Screen pentru a capta ecranul computerului iar mai apoi procesează imagina obținută prin a adăuga cursorul la poziția lui curentă pe ecran. Aceasta clasă este direct folosita de către obiectele de tip Recorder pentru a obține imaginile aferente ecranului la acel moment de timp.

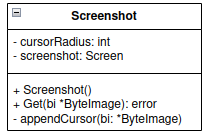


Fig 4.5.6

**KafkaConnection** (fig. 4.5.7)este clasa care se ocupă cu tot ce înseamnă interacțiunea microserviciului cu librăria Kafka. Este compus dintr-un Consumer și Producer Kafka. Prin intermediul metodelor sale microserviciul trimite mesaje către microserviciile adiacente și receptioneza mesaje de la acestea.

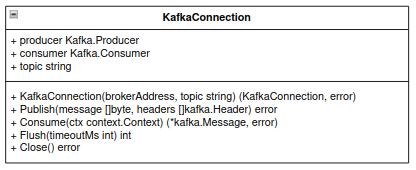


Fig 4.5.7

**Pachetul AudioMicroservice**

Clasa **Recorder** (fig. 4.5.8) este cea care înregistrează efectiv semnalele primite de la difuzor. Este instantiat de care punctul de start al microserviciului audio primind coada unde vor fi trimise blocurile de mărimi specificate în mod asincron. Folosind librăria portaudio acesta setează o funcție de callback care va fi apelată de către sistem în momentul în care a înregistrat o eșantion audio de mărimi reduse, pentru a avea o latentă minimă, care este adăugat într-un buffer. Acest buffer este procesat în paralel în următoarea maniera. Se așteaptă intervalul de timp specificat, mai apoi se scoate din bufferul audio ce s-a înregistrat, se adaugă sau se scoate informație în minus sau în plus, iar mai apoi este scris într-un fișier audio, al cărui cale este pusă în coada.

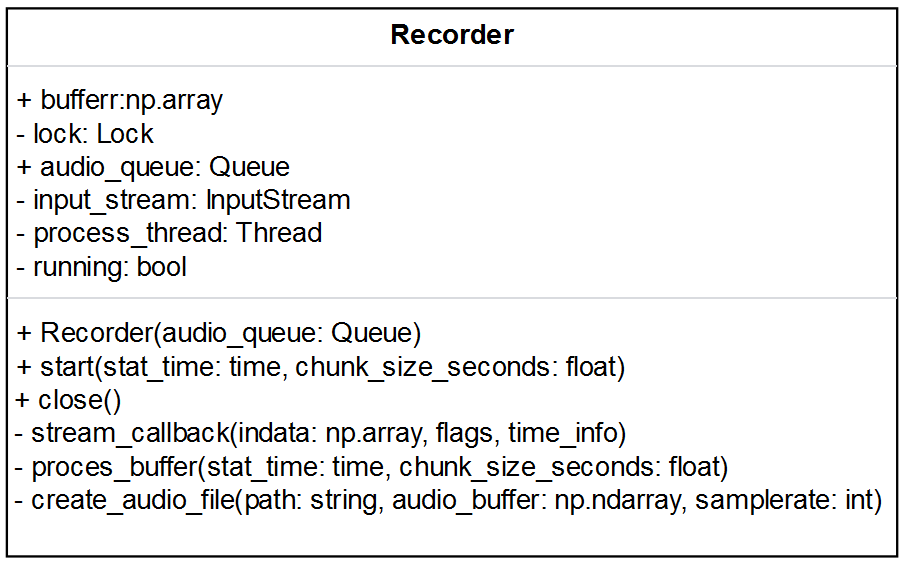


Fig 4.5.8

Fisierul **AudioMicroservice** reprezinta punctul de intrare al microserviciului audio, având ca principală funcționalitate înregistrarea audio. La pornire, acesta așteaptă un mesaj de la microserviciul AggregatorMicroservice care conține momentul de timp la care trebuie să înceapă înregistrarea (acest lucru este necesar pentru sincronizarea cu microserviciul video). Ulterior, se instantiază și pornește clasa Recorder, furnizând momentul de începere și durata în secunde a unui fișier audio. În cele din urmă, se pornește o buclă în care se ascultă coada furnizată de obiectul de tip Recorder, în vederea obținerii și trimiterii către microserviciul AggregatorMicroservice a căii către fișierul audio nou creat.

**Pachetul AggregatorMicroservice  
 AggregatorMicroservice** (fig. 4.5.9) este elementul central al sistemului de partajare deoarece are rolul de a aduna și a coordona datele primite de la cele două microservicii, AudioMicroservice și VideoMicroservice. Odată ce AggregatorMicroservice primește semnalul de la client pentru a începe înregistrarea, acesta devine responsabil pentru gestionarea tuturor datelor primite, asigurându-se că acestea sunt sincronizate și combinate în mod corespunzător. Pentru a realiza acesta, AggregatorMicroservice utilizează funcția CollectAudioAndVideoFiles, care face apeluri repetate la funcția GetNextSyncedAudioAndVideo pentru a sincroniza și combina datele audio și video. În plus, AggregatorMicroservice utilizează și funcția GetNextAudioAndVideo pentru a prelua datele primite de la microservicii și pentru a le procesa înainte de a le trimite mai departe către destinatarul final. Pe măsură ce înregistrarea progresează, AggregatorMicroservice monitorizează continuu datele pentru a se asigura că totul funcționează fără probleme și că nu există erori. Astfel, AggregatorMicroservice joacă un rol crucial în succesul întregului sistem, asigurându-se că datele sunt gestionate într-un mod eficient și precis.

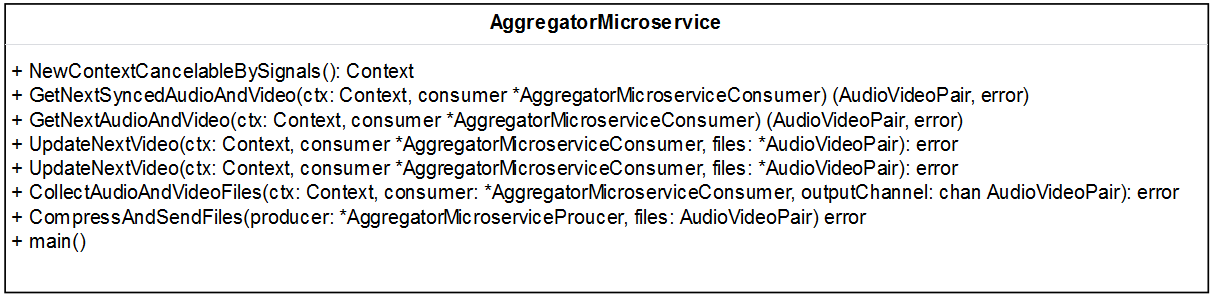


Fig 4.5.9

**AudioVideoPair** (fig. 4.5.10) încapsulează o pereche de fișiere audio și video. Aceasta conține logica necesară pentru combinarea și comprimarea acestora, precum și pentru ștergerea și inspectarea momentului în care au fost create.

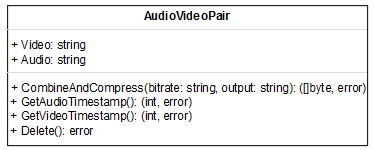


Fig 4.5.10

**KafkaConnection** (fig. 4.5.11) este clasa care se ocupă cu tot ce înseamnă interacțiunea microserviciului cu librăria Kafka. Este compus dintr-un Consumer și Producer Kafka. Prin intermediul metodelor sale microserviciul trimite mesaje către microserviciile adiacente și recepționă mesaje de la acestea. Clasa mai oferă și metode auxiliare pentru parsarea mesajelor Kafka primite (GetTypeFromMessage).

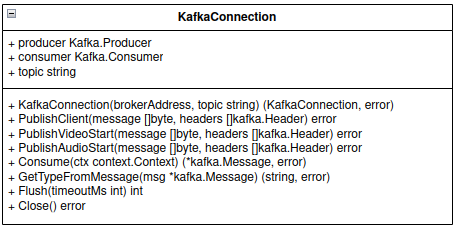


Fig 4.5.11

Scriptul **CombineAndCompress** reprezintă o unealtă utilă în procesul de manipulare și prelucrare a fișierelor video și audio, oferind capacitatea de a combina și comprima două fișiere separate într-unul singur, cu un bitrate personalizabil. Parametrii care trebuie dati scriptului sunt calea catre fisierul video, calea catre fisierul audio, factorul de compresie si calea catre fisierul de iesire.

**Pachetul InputExecutorMicroservice**

**InputExecutorMicroservice** (fig. 4.5.12) reprezintă punctul de intrare în microserviciu. Acesta ascultă pentru mesaje care conțin numeroase instrucțiuni ce trebuie executate și pentru timpul de așteptare între fiecare instrucțiune. Odată ce a primit un mesaj acesta îl descompune, îl parseaza și execută fiecare instrucțiune primită de la tastatura clientului.

Cu ajutorul clasei **KeyTranslator** (fig. 4.5.12), microserviciul de executare a comenzilor de intrare de la tastatură este capabil să parseze și să analizeze mesajele primite de la client. Această clasă permite maparea codurilor asociate butoanelor tastaturii cu un obiect corespunzător care conține informații necesare pentru interpretarea mesajelor primite. În momentul în care microserviciul primește un mesaj de la client, acesta poate apela metoda translate a clasei KeyTranslator, oferindu-i ca argument codul butonului apăsat pe tastatură. Aceasta va returna un obiect care conține informațiile necesare pentru interpretarea mesajului, cum ar fi comanda asociata butonului respectiv.

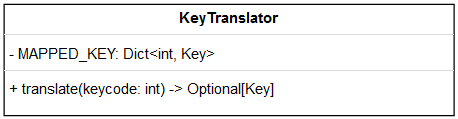


Fig 4.5.12

**User Interface Client**

**VideoWindow** (fig. 4.5.13) este elementul principal în afișarea informațiilor primite de la calculatorul care face partajarea. Acesta funcționează ca un singur element în interiorul aplicației PyQt pentru a putea fi folosit și administrat mai ușor. Clasa facilitează primirea informației de la calculatorul care face partajarea ecranului și sunetului, afișarea acestora în mod sincronizat și înregistrarea, salvarea și trimiterea tuturor comenzilor de la tastatură și mouse către partener, după cum urmează.  
 Prin intermediul mediului de comunicare Kafka, aplicația primește un fișier audio-video care este descompus în informație video și informație audio, fiind trimise în cozi specifice pentru a fi procesate mai departe. Afișarea video în interfață se face prin schimbarea repetată, rapidă a imaginilor în elementul pyqt, distanța așteptată între fiecare afișare este calculată din metadatele fișierului primit. Pe partea de audio, fiecare bloc audio este stocat într-o coadă specifică și scos de către un callback care trimite informația către difuzor.  
 Informația de la periferice (mouse și tastatură) este captată utilizând funcțiile de callback puse la dispoziție de către pyqt. La fiecare callback declanșat se adaugă într-un obiect de tip InputBuffer informația necesară executării comenzilor.  
 Clasele **StreamReceiverThread**, **DisplayContentThread**, **SendInputsThread** (fig. 4.5.36) sunt folosite pentru a primi și descompune fluxul de date audio-video de la partenerul de la celalalt capat, de a afișa informația video primită într-o maniera sincronizata și respectiv pentru a trimite evenimentele de la mouse si tastatura catre partener.

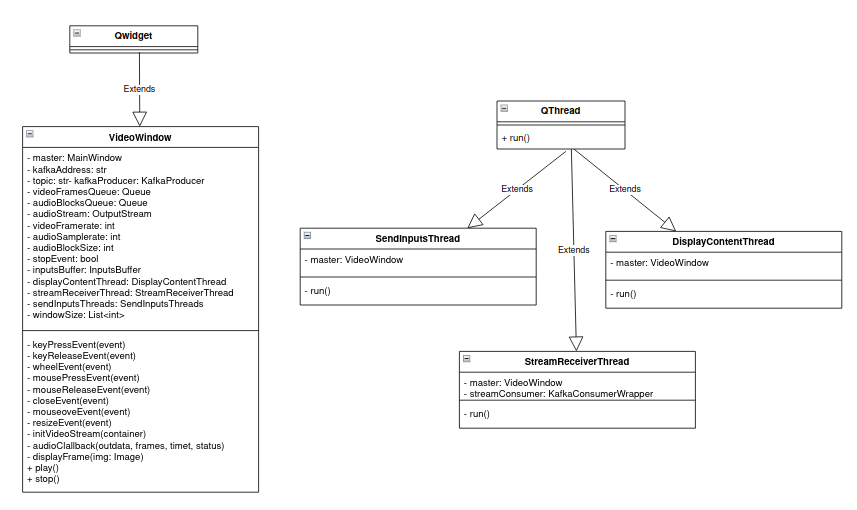


Fig 4.5.13

Clasa **InputBuffer** (fig. 4.5.14) a fost creată din nevoia de a avea un buffer thread safe pentru a putea stoca în ordine instrucțiunile primite de la perifericele computerului de pe care este executat.

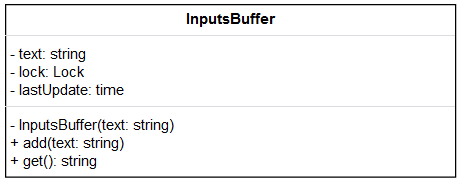


Fig 4.5.14

**KafkaContainer** (fig. 4.5.15) este clasa care se ocupa cu tot ce inseamna interactiunea aplicatiei client cu libraria Kafka. Este compus dintr-un consumer si producer kafka. Prin intermediul metodelor sale aplicatia trimite mesaje catre baza de date si receptiona mesaje de la aceasta. Clasa mai ofera si metode auxiliare pentru parsarea mesajelor Kafka primite (getStatusFromMessage).

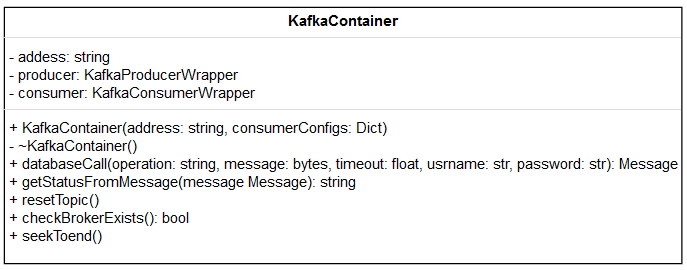


Fig 4.5.15

Clasa **MainWindow** (fig. 4.5.16) reprezinta punctul de intrare in aplicatia client. Aceasta clasa conecteaza modulele “Backend” si “Frontend” prin manipularea elementelor grafice din “Frontend”, de exemplu blocare elementelor din meniu in momentul cand utilizatorul nu este autentificat, si apeland functionalitatile din “Backend”, cum ar fi interogarea bazei de date.

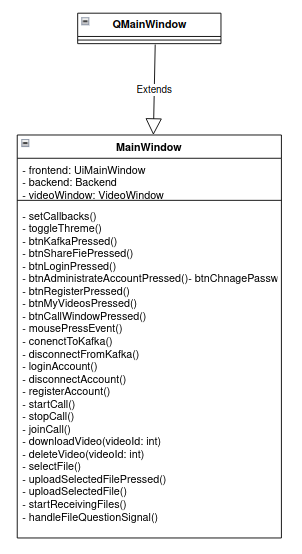


Fig 4.5.16

Clasa **Frontend** (fig. 4.5.17) face parte din componenta de frontend a interfeței cu utilizatorul. Acesta construiește ferestrele principale cu care clientul urmează să interacționeze. Ferestrele puse la dispoziție de componentă sunt cele de conectare la brokerul Kafka, înregistrarea unui nou utilizator, autentificarea, listarea tuturor videoclipurilor disponibile pentru descărcare și fereastra care facilitează conectarea la un partener sau partajarea resurselor unui alt computer.

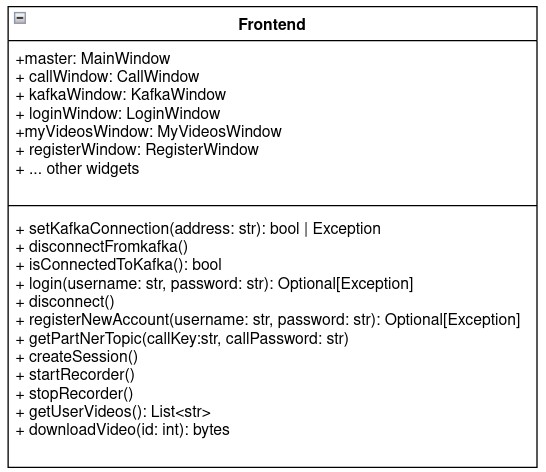


Fig 4.5.17

Clasa **Backend** (fig. 4.5.18) reprezinta inima functionalitatii aplicatiei intalate pe computerul clientului. Aceasta pune la dispozitie metode precum conectarea/deconectarea la kafka, autentificarea/deconectarea la cont, inregistrarea unui nou utlizator, creearea unei noi sesiuni, listarea si descarcarea sesiunilor disponibile si nu in ultimul rand posibilitatea de a te alatura unei sesiuni active.

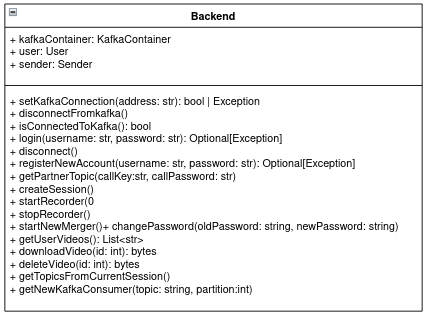


Fig 4.5.18

Clasa **User** (fig. 4.5.19) este utilizată pentru a reprezenta un utilizator autentificat în aplicație și pentru a administra informațiile asociate cu acesta. Clasa User poate fi utilizată pentru a stoca informații precum numele, parola și multe altele. Este folosită în contextul interfeței cu utilizatorul.

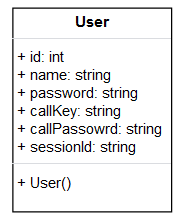


Fig 4.5.19

Clasa **Merger** (fig. 4.5.20) este folosită în vederea instantierii uniu microserviciu de tip MergeMicroservice. Microserviciul este desemnat cu rolul de a îmbina și salva toate piesele de fisere audio-video primite de la computerul care partajează.

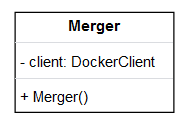


Fig 4.5.20

Clasa **Sender** (fig. 4.5.21) se ocupă cu administrarea microserviciilor de înregistrare a ecranului (VideoMicroservice, AudioMicroservice și AggregatorMicroservice). La apelarea metodei start acesta porteste fiecare microserviciu în ordinea corectă, iar în momentul în care clientul apleleaza oprirea acesta le oprește în mod sigur.

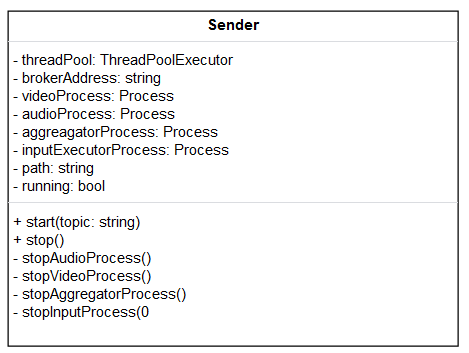


Fig 4.5.21

### II. Directorul CloudMicroservices

Directorul CloudMicroservices conține componentele în cloud necesare aplicației pentru a rula. Acesta este compus din:

1. MergerMicroservice - este instanțiat unul pentru fiecare sesiune. Acesta recepționează toate pachetele trimise de către AggregatorMicroservice și le combina într-un singur fișier care o sa fie salvat în baza de date.
2. DatabaseMicrosrvice - exista o singura instanța a acestui microserviciu în întreaga aplicație. Mentine o conexiune cu baza de date efectiva si acesta interactioneaza in mod direct cu ea. Restul de microservicii vor comunica cu acesta pentru a obține diferite informații (sesiuni, date conturi etc.).

**Pachetul DatabaseMicroservice**

Structura **ConsumerWrapper** (fig. 4.5.22) este o clasă care moștenește clasa Consumer din biblioteca confluent-kafka-go. Cea din urma este optimizată pentru a trimite mesaje de maxim 1 MB, iar în situația în care este necesară primirea unor mesaje mai mari, cum ar fi fișiere audio sau video, este necesar un obiect suplimentar care să permită trimiterea acestor mesaje. Acest obiect suplimentar este reprezentat de ConsumerWrapper, care îi permite utilizatorului să primeasca mesaje de dimensiuni mai mari prin împărțirea lor în bucăți mai mici și trimiterea acestora separat, astfel încât să poată fi procesate de Kafka. Această capacitate de trimitere a mesajelor mai mari poate fi utilă în situațiile în care este necesară transmiterea unor date voluminoase, cum ar fi fișiere multimedia sau baze de date mari.

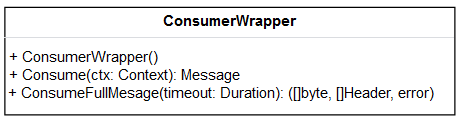


Fig 4.5.22

**DatabaseConsumer** (fig. 4.5.23) mosteneste structura ConsumerWrapper pentru a adauga functionalitati specifice de care are nevoie microserviciul care interactioneaza cu baza de date.

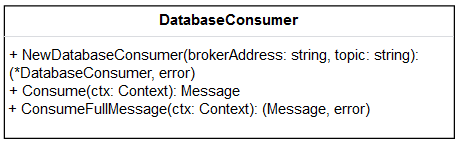


Fig 2.5.23

Structura **ProducerWrapper** (fig. 4.5.24) este o clasă care moștenește clasa Producer din biblioteca confluent-kafka-go. Cea din urma este optimizată pentru a trimite mesaje de maxim 1 MB, iar în situația în care este necesară trimiterea unor mesaje mai mari, cum ar fi fișiere audio sau video, este necesar un obiect suplimentar care să permită trimiterea acestor mesaje. Acest obiect suplimentar este reprezentat de ProducerWrapper, care îi permite utilizatorului să trimită mesaje de dimensiuni mai mari prin împărțirea lor în bucăți mai mici și trimiterea acestora separat, astfel încât să poată fi procesate de Kafka. Această capacitate de trimitere a mesajelor mai mari poate fi utilă în situațiile în care este necesară transmiterea unor date voluminoase, cum ar fi fișiere multimedia sau baze de date mari.

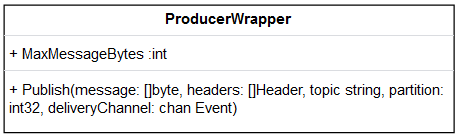


Fig 2.5.24

Clasa **DatabaseProducer** (fig. 4.5.25) este o subclasă a clasei ProducerWrapper, care a fost extinsă pentru a adăuga funcționalitățile specifice necesare pentru producerea de evenimente Kafka utilizate de microserviciul care gestionează baza de date. Această clasă permite crearea unei soluții scalabile și robuste pentru producerea de mesaje și comunicarea cu brokerul Kafka.

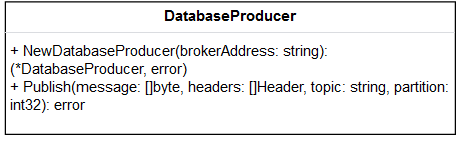


Fig 2.5.25

Clasa **Tools** (fig. 4.5.26) a fost creeata pentru functionalitatile legate de kafka dar care nu sunt legate de comunicatie.

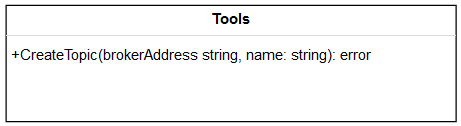


Fig 2.5.26

Structura **DatabaseManager** (fig. 4.5.27) este un obiect important în cadrul arhitecturii DatabaseMicroservice, fiind utilizat pentru a parsa mesajele primite de la celelalte microservicii care doresc să interacționeze cu baza de date. Odată ce mesajele sunt primite, DatabaseManager le procesează și aplică acțiunile corespunzătoare, fie că este vorba despre interogări sau actualizări ale bazei de date.  
După ce acțiunile sunt finalizate, DatabaseManager trimite înapoi un răspuns către microserviciul solicitant, care poate fi un răspuns de succes sau unul de eroare, în funcție de rezultatul acțiunilor aplicate. Capacitatea de a gestiona comunicarea între microservicii și baza de date prin intermediul obiectului DatabaseManager este esențială pentru un flux de lucru eficient și securizat în cadrul arhitecturii microserviciilor.

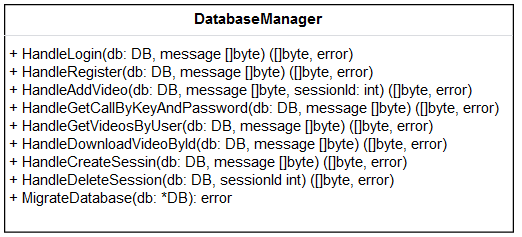


Fig 4.5.27

Clasa **FileOperations** (fig. 4.5.28) este folosită ca un utilitar pentru lucrul cu fișierele de către DatabaseMicroservice. Aceasta pune la dispoziție funcționalități precum verificarea existenței unui fișier și crearea unui nou fișier.

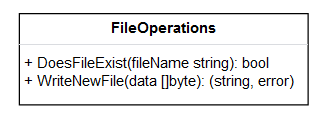


Fig 4.5.28

Clasa **PasswordOpeartions** (fig. 4.5.29) este folosită că un utilitar pentru lucrul cu fișierele către DatabaseMicroservice. Aceasta pune la dispoziție funcționalitatea de a face hash la orice string furnizat la intrarea metodei Hash.

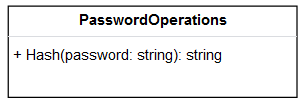


Fig 4.5.29

Punctul de intrare în microserviciul delegat de a interacționa cu baza de date este reprezentat de un obiect esențial numit **DatabaseMicroservice** (fig. 4.5.30), care este responsabil pentru crearea și gestionarea tuturor interacțiunilor cu baza de date. La pornire, DatabaseMicroservice efectuează verificarea existenței tabelelor în baza de date, creându-le în cazul în care acestea nu există. De asemenea, obiectul creează și topicul prin intermediul căruia se vor realiza toate comunicările între microservicii și baza de date.  
După ce toate aceste operațiuni sunt finalizate, DatabaseMicroservice începe să asculte pentru solicitațiile primite de la celelalte microservicii și să le proceseze în consecință. Prin intermediul acestui flux de lucru eficient, microserviciile pot interacționa cu baza de date fără a fi necesară o intervenție manuală constantă, ceea ce duce la o creștere a eficienței și a performanței întregului sistem.

**Pachetul MergerMicroservice**

Clasa **CustomKafkaMessage** (fig. 4.5.30) moștenește clasa Message din biblioteca confluent\_kafka pentru a crea mesaje personalizate în cadrul clasei KafkaConsumerWrapper și a se integra cu obiectele de tip Message primite de la consumatorii Kafka implicit.

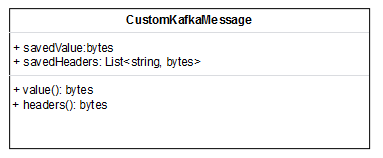


Fig 4.5.30

Clasa **KafkaProducerWrapper** (fig. 4.5.31) moștenește clasa Producer din confluent\_kafka. Sistemul Kafka este optimizat pentru a trimite mesaje de maxim 1 MO, astfel este nevoie de un obiect prin care putem trimite mesaje mai mari, de exemplu un fișier audio/video.

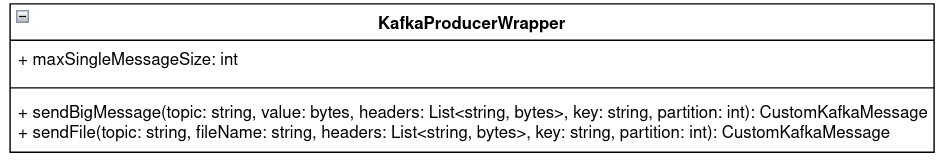


Fig 4.5.31

Clasa **KafkaConsumerWrapper** (fig. 4.5.32) este proiectată în vederea recepționării mesajelor personalizate trimise de un obiect de tipul KafkaProducerWrapper, în scopul primirii mesajelor mai mari de 1 MO.

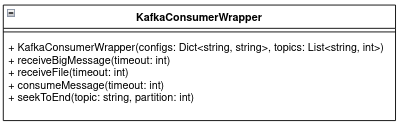


Fig 4.5.32

**Merger** (fig. 4.5.33) reprezintă punctul de intrare în microserviciul de îmbinare a videoclipurilor. Acest microserviciu are rolul de a captura toate piesele audio-video primite de la computerul care face partajarea, pentru a le salva ulterior în baza de date, prin intermediul unui apel la microserviciul corespunzător.

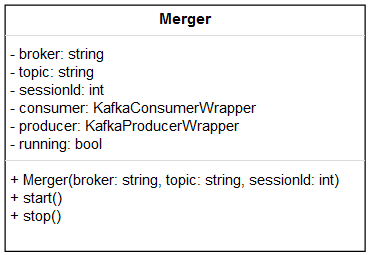


Fig 4.5.33

Clasa **Tempfile** (fig. 4.5.34) s-a născut din nevoia de a crea și administra fișiere temporare în mod rapid, intuitiv și simplu. Astfel, prin această clasă putem să creăm fișiere temporare cu opțiunea de ștergere automată în momentul în care obiectul este dealocat sau doar ștergere manuală. Aceste fișiere sunt stocate în folderul /tmp. Cu metodele de scriere și citire de text sau octeți, MergerMicroservice administrează fișierele primite de la computerul care partajeaza resursele prin salvarea lor temporară, apoi combinarea tuturor și salvarea în baza de date.

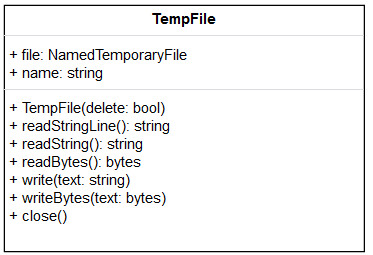


Fig 4.5.34

Clasa **VideoAggregator** (fig. 4.5.35) este un instrument creat pentru a îndeplini funcția de a combina mai multe fișiere într-unul singur. Metoda care face acest lucru primește ca parametri două șiruri de caractere. Primul reprezintă un text care descrie ce fișiere să fie combinate, iar al doilea reprezintă calea către fișierul de ieșire.



Fig 4.5.35

# Rezultate obținute și analiza datelor

## 5.1. Hardware:

Procesor: 11th Gen Intel© Core™ i5-11400H @ 2.70GHz × 6

Procesor grafic: Nvidia GeForce RTX 3060 Mobile

RAM: 16GB DDR4 3200MHz

Stocare: SSD M.2 PCIe

## 5.2 Software:

OS: Linux Mint 20.3 Cinnamon

Python: 3.8.10

Golang: 1.20.4 linux/amd64

## 5.3 Incarcare resurse pe microservicii:

Monitorizarea consumului de resurse este deosebit de importantă într-un proiect care implică mai multe microservicii. Pentru a realiza această monitorizare în cadrul proiectului, s-a folosit un program Python împreună cu biblioteca psutil. Această bibliotecă permite să accesarea si colectarea informațiilor relevante despre consumul de resurse al fiecărui microserviciu în parte.  
 Pentru a obține aceste informații, programul Python primește PID-urile microserviciilor din proiect și utilizează biblioteca psutil pentru a citi valorile asociate cu procesorul, RAM-ul, memoria virtuală, gestionarea fișierelor, subprocesele și threadurile. Aceste informații sunt actualizate la intervale regulate de 0.1 secunde.

### 5.3.1 AggregatorMicroservice:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Procesor (%) | RAM (MB) | Memorie virtuala (MB) | Handlere de fisiere | Procese fiu | Threaduri |
| Incarcare medie | 1.51 | 41.785 | 2636.51 | 25.38 | 1.08 | 27.702 |
| Maximum | 19.3 | 43.37 | 2657.69 | 32 | 2 | 28 |

### 5.3.2 AudioMicroservice:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Procesor (%) | RAM (MB) | Memorie virtuala (MB) | Handlere de fisiere | Procese fiu | Threaduri |
| Incarcare medie | 4.34 | 48.43 | 1519.44 | 17.39 | 0 | 19 |
| Maximum | 19.3 | 51.64 | 1525.48 | 29 | 0 | 22 |

### 5.3.3 Client:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Procesor (%) | RAM (MB) | Memorie virtuala (MB) | Handlere de fisiere | Procese fiu | Threaduri |
| Incarcare medie | 0.31 | 236.82 | 1962.91 | 38.72 | 6.80 | 15.14 |
| Maximum | 69.6 | 243.87 | 2578.10 | 41 | 8 | 19 |

### 5.3.4 InputExecutorMicroservice:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Procesor (%) | RAM (MB) | Memorie virtuala (MB) | Handlere de fisiere | Procese fiu | Threaduri |
| Incarcare medie | 0.23 | 31.82 | 708.47 | 28.74 | 0 | 10 |
| Maximum | 19.2 | 31.83 | 708.47 | 30 | 0 | 10 |

### 5.3.5 VideoMicroservice:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Procesor (%) | RAM (MB) | Memorie virtuala (MB) | Handlere de fisiere | Procese fiu | Threaduri |
| Incarcare medie | 62.83 | 83.49 | 2722.28 | 28.73 | 0 | 27.96 |
| Maximum | 105 | 95.19 | 2725.46 | 30 | 0 | 28 |

### 5.3.6 MergerMicroservice:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Procesor (%) | RAM (MB) | Memorie virtuala (MB) | Handlere de fisiere | Procese fiu | Threaduri |
| Incarcare medie | 0.54 | 25.94 | 690.11 | 22.76 | 0.015 | 9.99 |
| Maximum | 44.4 | 1037.75 | 1702.14 | 25 | 1 | 10 |

### 5.3.7 KafkaMicroservice

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Procesor (%) | RAM (MB) | Memorie virtuala (MB) | Handlere de fisiere | Procese fiu | Threaduri |
| Incarcare medie | 2.72 | 690.15 | 12307.2 | 402.96 | 0 | 82 |
| Maximum | 218.8 | 752.6 | 12444.7 | 415 | 0 | 83 |

# Concluzii

Proiectul dezvoltat oferă o soluție care permite utilizatorilor să controleze un calculator de la distanță. Punând la dispoziție o interfață intuitivă și funcționalități avansate, permițând utilizatorilor să manipuleze calculatorul țintă de la distanță. Au fost implementate cu succes funcționalități de control audio-video, permițând utilizatorilor să vizualizeze și să controleze ecranul, să redirecționeze sunetul și să interacționeze cu diferite dispozitive periferice. Aplicația oferă o experiență de control stabilă și fluidă, iar utilizatorii au posibilitatea să inițieze și să încheie sesiuni de control la distanță fără probleme majore. După încheierea unei sesiuni, toate informațiile audio-video înregistrate sunt salvate într-o bază de date pentru accesul ulterior.  
 Pe baza stadiului actual de implementare, există câteva posibile dezvoltări ulterioare ale aplicației. Un aspect important ar fi îmbunătățirea securității și a protecției datelor. Ar trebui să ne asigurăm că comunicația între utilizatori și calculatorul țintă este criptată și securizată, astfel încât să protejăm informațiile confidențiale de acces neautorizat. De asemenea, se poate explora opțiuni de optimizare a performanței pentru a asigura un control la distanță mai fluid și mai responsiv. Îmbunătățirea latentei și reducerea timpului de răspuns pot face experiența utilizatorilor mai plăcută și mai eficientă.   
 Acestea sunt doar câteva direcții posibile de dezvoltare ulterioară a aplicației. Scopul este de a o dezvolta și îmbunătăți pentru a satisface nevoile și cerințele utilizatorilor**.**

# Bibliografie

[1] G. Zhao, J. Liu, Y. Tang, W. Sun, F. Zhang, X. Ye, et al., "Cloud computing: A statistics aspect of users", *Cloud Computing*, pp. 347-358, 2009.

[2] *Latest telecommuting statistics*, 2015.

[3] TeamViewer. (2023), url: <https://www.teamviewer.com/en/>

[4] AnyDesk (2023), url: <https://anydesk.com/en>

[5] A Brief History Of Cloud Computing (2023), url: <https://www.ibm.com/blogs/cloud-computing/2014/03/18/a-brief-history-of-cloud-computing-3/>

[6] A Brief History of SSH and remote access (2023), url: <https://www.jeffgeerling.com/blog/brief-history-ssh-and-remote-access>

[7] PcAnywhere (2023), url: <http://www.anyplace-control.com/pcanywhere.shtml>

[8] Virtual Network Computing (2023), url: <https://www.sciencedirect.com/topics/computer-science/virtual-network-computing>

[9] Understanding the Remote Desktop Protocol (RDP) (2023), url: <https://learn.microsoft.com/en-us/troubleshoot/windows-server/remote/understanding-remote-desktop-protocol>

[10] TeamViewer Security Statement (2023), url: <https://dl.teamviewer.com/docs/en/TeamViewer-Security-Statement-en.pdf>

[11] RemotePC (2023), url:<https://www.remotepc.com/>

[12] X Window System, url: <https://www.techopedia.com/definition/10101/x-window-system>

[13] Most Popular Technologies 2020 (2023), url: [https://insights.stackoverflow.com/survey/2020#most-popular-technologies](https://insights.stackoverflow.com/survey/2020" \l "most-popular-technologies)

[14] The RedMonk Programming Language Rankings (2023), url: <https://redmonk.com/sogrady/2021/08/05/language-rankings-6-21/>

[15] Is AnyDesk Safe? (2023), url: <https://www.thecoldwire.com/is-anydesk-safe/>

[16] Anydesk Performance (2023), url: <https://anydesk.com/en/performance>

[17] LogMeIn Pro Security (2023), url: <https://www.logmein.com/pro/security-information>

[18] Modelarea Cazurilor de Utilizare (2023), url: <http://inf.ucv.ro/~mihaiug/courses/is/lab/lab3-4.pdf>  
[19] What is Go?, url: <https://www.freecodecamp.org/news/what-is-go-programming-language/>

[20] What is FFmpeg?, url: <https://api.video/what-is/ffmpeg/>

[21] Getting Started Writing Qt 6 Applications In Python With PySide6, url: https://medium.com/weekly-python/getting-started-writing-qt-6-applications-in-python-with-pyside6-389ee4c384ee  
[22] Apache Kafka Introduction, url: <https://kafka.apache.org/intro>

[23] MySQl 8.0 Reference, url: <https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html>

[24] What is Docker, url: https://www.ibm.com/topics/docker