C U R S 3

Colectare de date, curatare si corpus

Memoria aplicatiei EchoChamber

CURS 3.

C1 LLM-uri, API-uri – ce construim și cum pornim
C2 Ecosistemul de modele – alegem modelul de bază
C3 Date și corpus – colectare, curățare, metadate
C4 Prompting, adnotare, context engineering
C5 Embeddings, retrieval, RAG
C6 RAG, agenți, LangChain, LangGraph
C7 Agenți orchestrați – LangGraph,metrici,etică
C8 Integrare aplicație – Gradio
C9 Demo final – prezentări și feedback

Tematică
- Date, corpus și curățare
- Surse publice: YouTube
- Metadate, snapshot și reproductibilitate
- Introducerea bulelor discursive
- Sentiment vs poziționare față de țintă
- Primul prompt exploratoriu pe comentarii politice

Activități practice
- Colectăm datele din surse YouTube
- Curățăm și standardizăm datele
- Discutăm exemple clare pentru fiecare bulă discursive
- Observăm limitele promptului simplu: ambiguitate, ținte multiple, confuzie între sentiment și stance

Livrabile
- `data/raw/`
- `data/cleaned/`
- `docs/README_corpus.md`
- `annotation_prompt_v0.md`

→ **La finalul C3**
- Avem un corpus curățat și documentat
- Avem primele exemple pentru fiecare bulă discursivă

Astăzi construim memoria aplicatiei

FARA CORPUS

- Modelul răspunde din datele de antrenament
- Halucinații despre context local și actual
- Nu poți ancora raspunsul în realitate
- Bulele discursive nu au baza empirica

CU CORPUS

- Modelul răspunde ancorat în surse reale
- Poți cita sursa, poti audita rezultatul
- Bulele emergente din discurs real observat
- Sistemul este reproductibil și evaluabil

Sursă	Ce colectăm	Acces	Rol în curs
YouTube	comentarii la videouri politice	API / export pregătit	**activ** - corpus principal pentru adnotare
Presă RSS	titluri, rezumate, articole scurte	feed public, stabil	**activ** - context factual / instituțional
Documente oficiale	comunicate, decizii, rapoarte, texte publice	manual / site-uri publice	**opțional** - surse de verificare
Facebook public	postări și comentarii publice	manual / demo	**demo** - dificil tehnic și etic
Telegram public	mesaje din canale publice	API Telegram	**demo** - discurs mai polarizat, necesită prudență

Colectare YouTube: corpus de comentarii politice

Sursele sunt alese ca puncte de pornire pentru zone discursive diferite, nu ca etichete finale ale comentariilor. Un comentariu colectat de la un canal suveranist nu este automat „suveranist”; această clasificare se face ulterior, prin adnotare.

Zonă discursivă	Exemple de surse	Rol în corpus
Mainstream instituțional	StirileProTV, digi24hd56, InfoTVR, hotnews_ro, euronewsro, Antena3CNN, libertatearo, europafmromania, b1tvchannel, rfiromania, euractivromania9532	Discurs media central, relativ instituțional; oferă bază de comparație pentru celelalte zone
Civic / investigativ / corectiv	RecorderRomania, europa-libera-romania, g4media479, VeridicaRO, StareaNatiei	Discurs critic față de putere, dar orientat spre transparență, verificare și responsabilitate publică
Suveranist oficial / anti-sistem electoral	georgesimionoficial, partidulAUR2024, CălinGeorgescu-CanalulOficial, DianaSosoacaOfficial	Discurs politic mobilizator, anti-elite, centrat pe suveranitate, identitate națională și neîncredere în instituțiile dominante
Neîncredere sistemică / media alternativă	RomaniaTVOFICIAL, realitatea2025, TuDecizi-s3g, turcescu111	Surse cu audiențe active în zone de contestare politică; utile pentru observarea discursului anti-sistem și a suspiciunii față de instituții
Conspiraționist / marginal / mitic-național	AdevaruriSecrete, otvdirect, AltcevacuAdrianArtene, roxindaniel, Canal33Romania, Curiozitate-Canal33, Spiritualitate-Canal33	Discurs cu narațiuni alternative, explicații ascunse, teme spiritualizate sau mitologizate; util pentru compararea cu discursul politic instituțional

Sursă selecție media: Sultănescu et al. (2026), *Media leaning in Romania: an audience-centric study*, *Journal of Contemporary European Studies*. DOI: 10.1080/14782804.2026.2627986.

Cum creăm cheia pentru YouTube API

1. Intrăm în Google Cloud Console
Folosim un cont Google obișnuit.
2. Creăm un proiect
Exemplu: `echochamber-course`
3. Activăm YouTube Data API v3
APIs & Services → Library → caută `YouTube Data API v3` → Enable
4. Creăm cheia API
APIs & Services → Credentials → Create credentials → API key
5. Punem cheia în fișierul `.env`
`YOUTUBE_API_KEY=cheia_ta_aici`
6. Verificăm că `.env` nu ajunge pe GitHub
În `.gitignore` trebuie să existe:
`.env`

Corpus utilizabil: calitate, unitate, probleme

Construim un corpus curat, documentat și ușor de folosit pentru adnotare, căutare semantică și agenți.

Decizie	Recomandat	De evitat	De ce contează
Calitatea corpusului	comentarii relevante, din videoclipuri politice, curate și documentate	volum mare fără selecție	datele devin mai ușor de analizat și comparat
Unitatea de lucru	comentariu + context minim: canal, videoclip, dată, link	comentariu fără context sau text prea scurt	putem interpreta mai corect sensul comentariului
Curățare	eliminare duplicate, filtrare după limbă și lungime, eliminare linkuri și zgomot	spam, comentarii tăiate, texte aproape goale	reducem erorile și etichetele greșite
Metadate	canal, titlu videoclip, dată, link, platformă	texte fără sursă clară	putem verifica, reproduce și cita rezultatele
Probleme speciale	marcăm sarcasm, ironie, ținte multiple, ambiguitate	forțăm interpretări clare acolo unde textul e neclar	evităm clasificări false

Lanțul de prelucrare a datelor

FLUXUL DE DATE

SURSE
canale YouTube selectate
|
COLECTARE
script Python + YouTube Data API
|
FILTRARE INIȚIALĂ
lungime minimă, proporție de litere, eliminare linkuri
|
STANDARDIZARE
același format pentru toate comentariile
|
CORPUS DOCUMENTAT
data/corpus_youtube.jsonl + metadate + README_corpus.md

UNDE INTRA CORPUSUL IN APLICATIE

Corpus documentat (C3)
|
Adnotare asistată (C4)
target, stance, sentiment, ton, registre discursive
|
Export pe bule (C4)
data/bubbles/*.jsonl
|
Căutare semantică (C5)
index vectorial FAISS / Chroma
|
Răspunsuri cu surse (C5-C6)
context relevant din corpus
|
RAG + agenți (C6)
răspunsuri ancorate în date reale

Datele nu intră direct în model. Ele trec prin colectare, filtrare, standardizare și documentare.

Ce colectam efectiv — schema minima

CAMPURI OBLIGATORII

```
{
  "id": "yt_<video_id>_<comment_id>",
  "source_platform": "youtube",
  "source_channel": "<handle>",
  "text": "...",
  "text_raw": "...",
  "bubble_label": null,
  "bubble_self_identified": false,
  "topic": null,
  "rhetoric_type": null,
  "video_id": "...",
  "video_title": "...",
  "video_date": "YYYY-MM-DD",
  "comment_date": "YYYY-MM-DD",
  "likes": 0,
  "lang": "ro",
  "collected_at": "YYYY-MM-DD"
}
```

- text — continutul analizat si indexat
- platform — variabila de analiza cross-platform
- sursa — trasabilitate si audit
- data — analiza temporala, detectie tendinte
- link — verificare si reproducere
- tip — filtru in retrieval si anotare

Curatarea datelor

PASI OBLIGATORII

- Eliminăm duplicatele sau textele aproape identice
- Filtrăm textele prea scurte sau greu de interpretat
- Verificăm limba și păstrăm textele relevante pentru corpus
- Eliminăm zgomotul: linkuri, simboluri excesive, spații inutile
- Păstrăm textul original pentru verificare

DE CE E IMPORTANT

- Corpusul necurățat produce rezultate instabile
- Textele foarte scurte duc la etichete fragile
- Duplicatele pot distorsiona distribuția bulelor discursive
- Metadatele permit verificarea și reproducerea pașilor
- Curățarea reduce erorile în adnotare și căutare semantică

Standardizarea datelor

INAINTE DE STANDARDIZARE

Datele vin în formatul propriu al platformei:
`textDisplay`, `likeCount`, `publishedAt`, `videoId`, `title`

DUPA STANDARDIZARE

Le transformăm într-o schemă comună pentru corpus:
`id`, `text`, `text_raw`, `source_platform`, `source_channel`, `video_id`, `video_title`, `comment_date`, `likes`, `collected_at`

Schema datelor colectate?

Grup	Câmpuri	Rol
Identificare	`id`, `video_id`	identifică unic comentariul și videoclipul
Sursă	`source_platform`, `source_channel`	arată de unde vine textul
Text	`text`, `text_raw`	păstrează varianta curățată și varianta originală
Context	`video_title`, `video_date`, `comment_date`, `likes`	oferă context pentru interpretarea comentariului
Adnotare ulterioară	`bubble_label`, `bubble_self_identified`, `topic`, `rhetoric_type`	câmpuri lăsate goale la colectare, completate mai târziu
Metadate și reproducere	`lang`, `collected_at`	ajută la filtrare, verificare și reproducere

```
{
  "id": "yt_<video_id>_<comment_id>",
  "source_platform": "youtube",
  "source_channel": "<handle>",
  "text": "...",
  "text_raw": "...",
  "bubble_label": null,
  "bubble_self_identified": false,
  "topic": null,
  "rhetoric_type": null,
  "video_id": "...",
  "video_title": "...",
  "video_date": "YYYY-MM-DD",
  "comment_date": "YYYY-MM-DD",
  "likes": 0,
  "lang": "ro",
  "collected_at": "YYYY-MM-DD"
}
```

Consideratii etice

CE NU COLECTĂM

- Date personale identificabile: nume complet, telefon, email, adresă, CNP
- Conținut din surse private, grupuri închise sau conturi restricționate
- Date colectate prin metode care încalcă termenii platformei
- Articole complete protejate de copyright
- Username-uri păstrate ca identitate analizabilă
- Texte care pot expune persoane vulnerabile sau situații sensibile

CE FACEM OBLIGATORIU

- Folosim doar surse publice și accesibile
- Lucrăm cu fragmente relevante, nu cu texte complete
- Păstrăm metadatele: sursă, dată, platformă, link
- Documentăm regulile de selecție în README_corpus.md
- Anonimizăm nume, ID-uri și orice indiciu personal
- Separăm analiza discursului de evaluarea persoanelor
- Notăm limitele corpusului: ce include, ce exclude, ce bias poate avea

https://www.oxethica.com/blog/ai-ethics-and-values

Bule discursive

- Nu este suficient să spunem că un comentariu este „pozitiv” sau „negativ”. Un text negativ poate ataca instituțiile, poate apăra un lider sau poate invoca o conspirație. Poziția trebuie legată de o țintă.
- Bulele nu sunt personaje inventate și nu sunt grupuri sociale reale. Sunt tipare de limbaj care apar repetat într-un corpus politic.
- Pornim de la comentarii reale și ne uităm la trei lucruri: despre cine vorbește comentariul, ce poziție are față de acea țintă și ce logică discursivă activează.

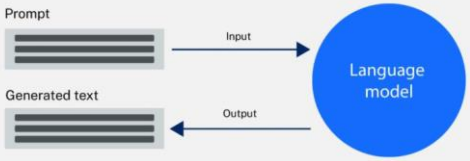
Bulele apar din combinații între:

- **`target`** - cine este evaluat
- **`stance`** - poziția față de țintă
- **`sentiment`** - valența general
- **`tone`** - modul de formulare
- **`registru discursiv`** - logica politică activată

Agent	Personalitate	Cum vorbește	Ce îl definește
Personalist-salvator	devotat, admirativ, sigur	laudativ, emoțional, încrezător	vede liderul ca soluție excepțională
Anti-sistem	furios, suspicios, dezamăgit	acuzator, moralizator, direct	vede instituțiile și „sistemul” ca profund compromise
Anti-suveranist	critic, vigilent, defensiv	contestatar, mai argumentativ	respinge liderii și discursul suveranist
Conspiraționist	alarmist, hiper-suspicios	speculativ, revelator, totalizant	explică evenimentele prin forțe ascunse și actori externi
Pro-european	normativ, moderat, legalist	sobru, justificativ, procedural	apără regulile, instituțiile și ancorarea europeană

Prompt Engineering

→ cum transformi o cerere vagă într-o instrucțiune clară și controlabilă pentru model.



https://www.digital.nsw.gov.au/policy/artificial-intelligence/chatbot-prompt-essentials

- Procesul de formulare și rafinare a prompturilor pentru a obține răspunsuri mai bune de la model.
- Ghidezi modelul prin instrucțiuni clare, context relevant și format explicit de răspuns.
- Presupune testare și ajustare: compari variante de formulare, structură și exemple, apoi modifici promptul.
- Scopul este să obții răspunsuri mai clare, mai utile, mai consistente și mai ușor de evaluat.
- Reduce răspunsurile vagi, incomplete sau greșite prin clarificarea sarcinii și a regulilor.
- Include tehnici precum zero-shot, few-shot, role prompting și răspuns structurat.

→ Promptul este interfața de lucru dintre utilizator și model.

Rolurile conversației

system, user, assistant

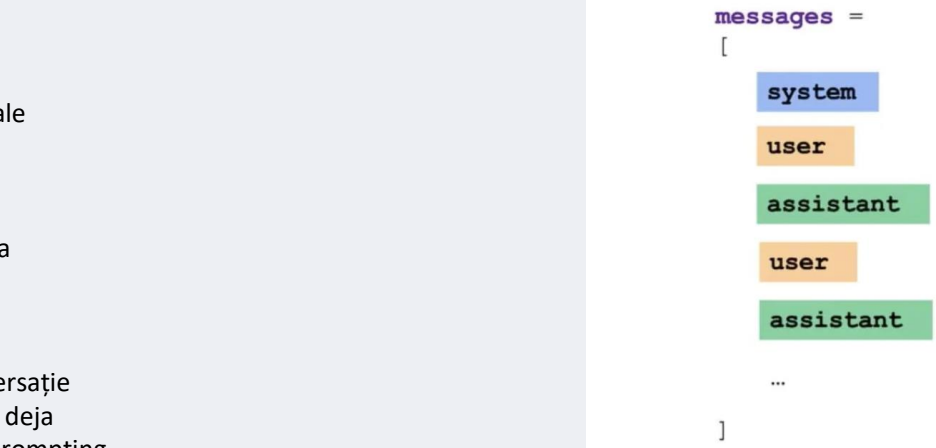
- system = regulile jocului
- user = cererea curentă
- assistant = ce a spus deja modelul / exemplul de răspuns

system / developer
- stabilește rolul modelului, regulile și constrângerile generale
- spune *cum* trebuie să răspundă modelul

user
- aduce cererea concretă, textul, documentul sau întrebarea
- spune *ce task* trebuie făcut acum.

assistant
- reprezintă răspunsurile anterioare ale modelului din conversație
- ajută la păstrarea continuității: modelul vede ce a răspuns deja
- poate fi folosit și ca exemplu de output dorit în few-shot prompting

Exemplu:
- system: „Ești un analist de discurs politic. Etichetezi strict pe baza textului și returnezi doar JSON valid.”
- user: „Analizează comentariul: «Guvernul minte din nou despre inflație.»”
- assistant: {"target":"guvern","stance":"anti","ton":"acuzator"}
- user: „Acum analizează și: «Bruxelles-ul ne impune politici împotriva intereselor noastre.»”

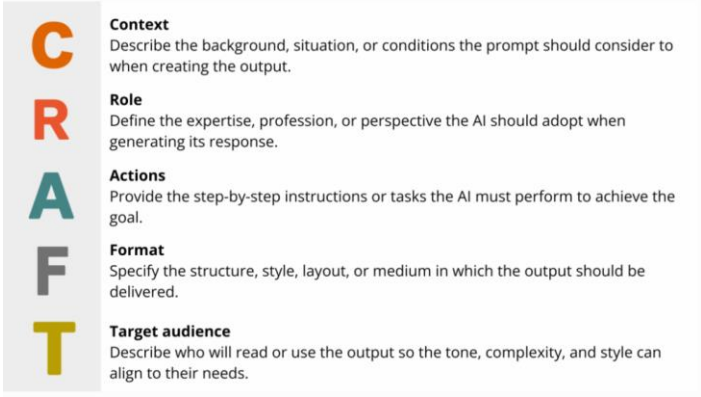


https://medium.com/techhappily/prompt-engineering-part-1-fbbf42904266

Ce face un prompt bun

- **Definește criteriul de succes înainte să scrii.** Cum arată un răspuns corect? Cum arată unul greșit? Fără asta nu poți evalua și nu poți îmbunătăți.
- **Specifică formatul răspunsului explicit.** Nu „răspunde scurt" - ci „maxim 3 propoziții". Nu „returnează date structurate" - ci câmpurile exacte, tipurile exacte. Cu cât formatul e mai precis, cu atât răspunsul e mai stabil și mai ușor de procesat.
- **Separă instrucțiunile de date.** Instrucțiunea spune ce să facă modelul; datele sunt textul pe care îl procesează. Dacă le amesteci, modelul poate trata fragmente din date ca instrucțiuni. Folosește delimitatori expliciți - etichete, separatori, titluri.
- **1–3 exemple când vrei consistență de stil sau format.** Exemplele funcționează mai bine decât orice descriere verbală pentru taskuri ambigue sau stilistice.
- **Descompune taskurile complexe.** Un prompt care cere simultan extracție, clasificare și rezumare va performa mai slab decât trei pași separați conectați într-un flux.
- **Gândește-te și la ce nu vrei.** Constrângerile negative contează: „nu adăuga introduceri", „nu reformula întrebarea", „nu folosi liste dacă nu există cel puțin 3 elemente".
- **Construiește iterativ.** Testezi pe exemple reale, compari variante, modifici o singură variabilă o dată.

→ **Un prompt bun nu este doar clar; este testabil, repetabil și ușor de evaluat.**



https://bacoach.nl/2025/11/prompt-engineering-craft-framework/

https://www.altexsoft.com/blog/prompt-engineering/

altexsoft

Partea tehnică 1: colectăm datele

Descărcăm sau încărcăm texte brute din surse publice și le salvăm într-un format simplu.

CE FACEM ÎN NOTEBOOK

- Alegem un canal YouTube prin handle
- Colectăm 1videoclipuri, maximum 50 comentarii per videoclip
- Salvăm datele brute în `data/raw/`
- Curățăm rapid textul și salvăm în data/cleaned/

CE TREBUIE SĂ CONȚINĂ DATELE BRUTE

```
id
source_platform
source_channel
text_raw
video_id
video_title
video_date
comment_date
likes
collected_at
```

FIȘIERE

```
notebooks/
  student_XX/
    c3_collect_clean.ipynb      # colectare mică + curățare rapidă
scripts/
  collect_youtube.py            # script final de colectare
  clean_youtube.py              # script comun de curățare
data/
  raw/
    student_XX_youtube_raw.jsonl
  cleaned/
    student_XX_youtube_clean.jsonl
  provided/
    corpus_youtube_large_raw.jsonl
```

Păstrăm separat datele brute, datele curățate și corpusul mare oferit pentru analiză.

Partea tehnică 2: curățăm și salvăm corpusul

Transformăm datele brute într-un corpus utilizabil pentru adnotare și retrieval.

CE FACEM ÎN NOTEBOOK

- Citim fișierul brut colectat individual
- Curățăm textul: linkuri, spații, caractere inutile
- Eliminăm comentariile foarte scurte
- Eliminăm duplicatele
- Salvăm rezultatul în `data/cleaned/`
- Comparăm câteva exemple: `text_raw` vs. `text`

REGULI MINIME DE CURĂȚARE

- elimină linkurile
- normalizează spațiile
- elimină textele foarte scurte
- elimină duplicatele
- păstrează text_raw
- creează text curățat
- nu rescrie sensul comentariului

FIȘIERE

```
notebooks/
  student_XX/
    c3_collect_clean.ipynb          # colectare mică +
curățare rapidă
scripts/
  clean_youtube.py                  # curățare comună,
aceeași pentru toți
data/
  raw/
    student_XX_youtube_raw.jsonl
  cleaned/
    student_XX_youtube_clean.jsonl
    corpus_youtube_large_clean.jsonl
  provided/
    corpus_youtube_large_raw.jsonl
```

Curatarea nu inseamna sa facem textul 'frumos'. Inseamna sa il facem stabil, comparabil si verificabil.

Partea tehnică 3: primul prompt exploratoriu

Testăm pe corpusul curățat dacă modelul poate observa target, stance, ton

CE FACEM ÎN NOTEBOOK

- Încărcăm `corpus_youtube_large_clean.jsonl`
- Alegem 10 comentarii curate
- Trimitem fiecare comentariu la model
- Cerem răspuns scurt în JSON
- Comparăm rezultatul cu lectura noastră
- Notăm unde modelul greșește

CE OBSERVĂM

- Modelul poate confunda tonul cu poziționarea
- Un comentariu poate avea mai multe ținte
- Sarcasmul și ironia sunt greu de interpretat
- JSON-ul poate fi instabil fără schemă clară
- Avem nevoie de definiții și exemple în C4

PROMPT EXPLORATORIU

```
Ești analist de discurs politic.
Citește comentariul și identifică:
1. ținta principală
2. poziționarea față de țintă
3. tonul
4. tema
5. probleme de interpretare
Răspunde scurt, în JSON.
Comentariu:
<<< {comment_text} >>>
```

OUTPUT AȘTEPTAT

```
{
  "target": "...",
  "stance": "support|attack|neutral|ambiguous",
  "tone": "...",
  "topic": "...",
  "interpretation_problem": "...",
  "reason": "..."
}
```

FIȘIERE

```
notebooks/
  student_XX/
    c3_exploration_prompt.ipynb # Tema 1
data/
  cleaned/
    corpus_youtube_large_clean.jsonl
outputs/
  student_XX_prompt_outputs.jsonl   #
opțional
```

Promptul exploratoriu nu produce etichete finale. Ne arată ce trebuie definit mai clar în C4.

Livrabil C3: corpus curățat și documentat

Individual
- notebooks/student_XX/c3_collect_clean.ipynb — colectare mică + curățare rapidă
- data/raw/student_XX_youtube_raw.jsonl — eșantion brut individual
- data/cleaned/student_XX_youtube_clean.jsonl — eșantion curățat individual
- notebooks/student_XX/c3_exploration_prompt.ipynb — Tema 1: explorare + primul prompt

Echipă
- scripts/collect_youtube.py — script final de colectare YouTube
- scripts/clean_youtube.py — script comun de curățare
- data/provided/corpus_youtube_large_raw.jsonl — corpus mare oferit de profesor
- data/cleaned/corpus_youtube_large_clean.jsonl — corpus mare curățat
- README.md — secțiunea Data Gathering completată

Opțional
- outputs/student_XX_prompt_outputs.jsonl — rezultate salvate din primul prompt

Data viitoare

Curs 4 - De la Prompt la Context Engineering

În C4 folosim corpusul curățat din C3 și începem să-l transformăm în date analizabile.

Ce construim în C4

- promptul de adnotare pentru comentarii politice
- schema JSON: target, stance, sentiment, tone, bubble_candidate
- primul batch de comentarii adnotate cu LLM
- verificare umană pentru cazurile ambigue

-> În C3 construim corpusul. În C4 îl transformăm în variabile analizabile.