Proiect AI MYNews

Autori: Iorgu Florentin Lucian
Nonea Costin
Vaduva David
Grupa:441C
Profesor coordonator: Curavale Daniel

Capitolul 1:Introducere

În secolul XXI, informația a devenit resursa cea mai abundentă, dar și cea mai greu de gestionat. Dacă în deceniile trecute principala provocare a consumatorului de știri era accesul la informație, astăzi problema s-a inversat radical: provocarea este **filtrarea** acesteia. Ne aflăm în era „infobezității" sau a supraîncărcării informaționale (*Information Overload*), un fenomen în care volumul de date depășește capacitatea cognitivă a individului de a procesa, evalua și asimila conținutul.

Zilnic, un flux neîntrerupt de articole, reportaje și actualizări din surse diverse — de la agenții de presă tradiționale până la platforme de social media — inundă ecranele utilizatorilor. Această avalanșă de conținut duce adesea la ceea ce psihologii numesc „oboseala decizională" și la o scădere a calității informării. Atunci când totul este livrat cu aceeași intensitate, nimic nu mai pare prioritar. În acest zgomot digital, relevanța devine moneda de schimb cea mai prețioasă.

Relevanța în știri nu se referă doar la subiectul abordat, ci și la modul în care informația este prezentată. Un raport financiar complex poate fi esențial pentru un analist economic, dar complet opac pentru un student sau un pasionat de tehnologie care caută doar impactul practic al acelor cifre. Aici intervine conceptul de **personalizare contextuală**. Consumatorul modern nu mai are timpul necesar să parcurgă zeci de pagini pentru a extrage esența; el are nevoie ca informația să vină către el deja „tradusă" pe nivelul său de interes, pe un ton care să rezoneze cu stilul său de viață și într-un format care să respecte economia atenției sale.

Importanța relevanței este critică și din perspectiva combaterii dezinformării și a senzaționalismului. Multe platforme media utilizează tehnici de *clickbait* pentru a atrage atenția, sacrificând adesea calitatea și obiectivitatea. Un sistem capabil să identifice nucleul informațional al unei știri și să îl livreze fără balastul stilistic adesea părtinitor al sursei originale reprezintă nu doar o facilitate tehnologică, ci un instrument necesar pentru o dietă informațională sănătoasă.

Definirea temei My News

În acest peisaj complex, inteligența artificială nu mai este doar un lux, ci o necesitate. Proiectul **MYNews** pornește de la această premisă: tehnologia trebuie să acționeze ca un filtru inteligent, un „editor personal" care înțelege cine este cititorul din spatele ecranului și care livrează doar ceea ce contează, exact așa cum contează pentru acesta. Prin automatizarea acestui proces de selecție și rafinare, putem transforma consumul de știri dintr-o activitate copleșitoare într-una eficientă, educativă și, mai ales, personalizată.

Arhitectura Inovatoare: Conceptul Multi-Agent

Pentru a răspunde complexității profilării umane și necesității de rafinare a textului, proiectul **MYNews** nu utilizează o abordare monolitică, ci implementează o arhitectură de tip **Chain-of-Agents** (Lanț de Agenți). Inovația constă în divizarea procesului de procesare în două etape cognitive distincte, fiecare fiind gestionată de o instanță separată de Model de Limbaj Mare (LLM). Această separare a responsabilităților asigură o precizie mult mai mare în livrarea

conținutului personalizat. Această separare a responsabilităților asigură opersonalizare mult mai precisă decât sistemele standard.

A. Agentul Profilator (LLM 1)

Acest prim agent acționează ca un analist. El preia descrierea liberă a utilizatorului și o transformă într-un **System Prompt** (instrucțiune de sistem) structurat.

- **Rol:** Identifică preferințele tematice, stabilește tonul (ex: formal, glumeț) și reglează nivelul de complexitate (ex: pentru experți sau pentru copii).
- **Rezultat:** O metodologie de lucru personalizată sub formă de directive clare pentru următorul agent.

B. Agentul Rafinator (LLM 2)

Cea de-a doua componentă primește știrile brute și directivele de la Agentul 1.

- **Rol:** Execută procesul de **Style Transfer** și sinteză. El nu doar scurtează textul, ci îl rescrie complet pentru a se potrivi cu "personalitatea" definită anterior.
- **Rezultat:** Un buletin de știri personalizat, filtrat de informații irelevante și adaptat stilului dorit de utilizator.

:

Capitolul 2: Arhitectura Tehnică a Sistemului

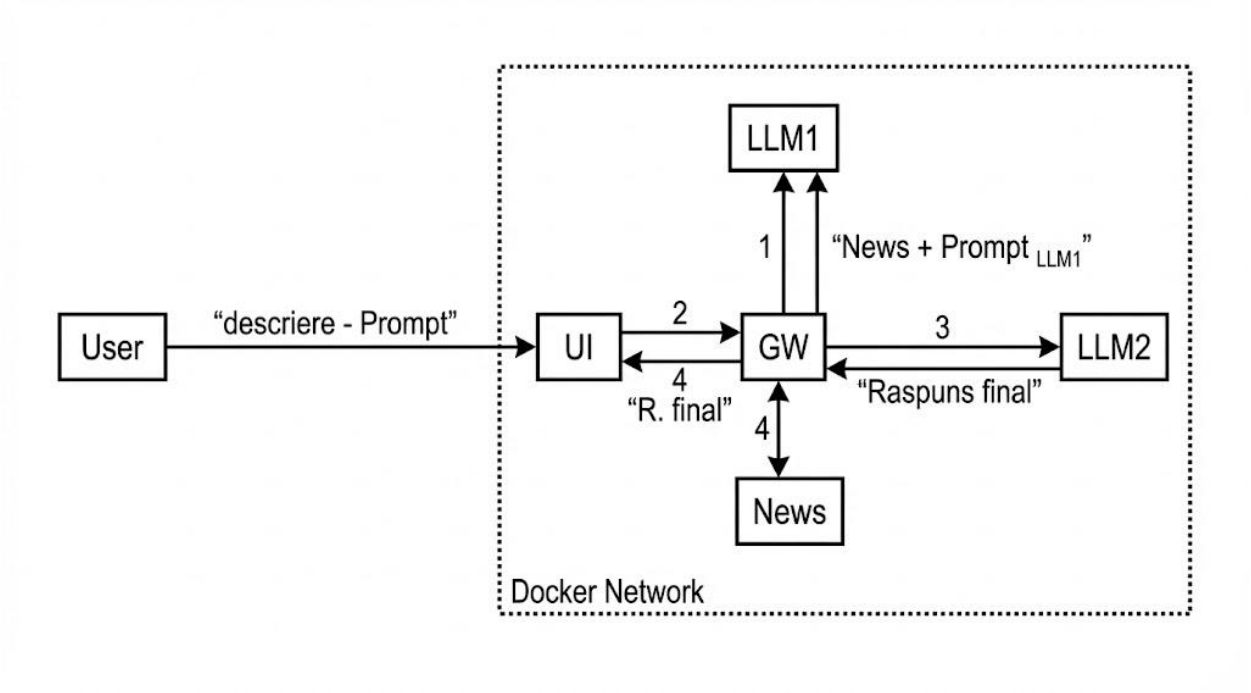
2.1 Structura si diagrama flux

Proiectul este construit folosind o arhitectură modulară, unde fiecare componentă rulează într-un container izolat. Această abordare permite scalabilitatea și mentenanța ușoară a sistemului.

Componentele principale:

1. **UI (Streamlit):** Interfața grafică prin care utilizatorul comunică cu sistemul.
2. **Gateway (FastAPI):** Orchestratorul central care dirijează fluxul de date între toate celelalte servicii.
3. **News Service (FastAPI):** Gestionarul bazei de date de știri (stocare persistentă în format JSON).

4&5 **LLM Engines (Ollama):** Două instanțe separate de AI care rulează modelul Llama 3.2.



Arhitectura proiectului MYNews este structurată sub forma unui ecosistem de microservicii interconectate, unde fiecare modul îndeplinește un rol specific în lanțul de procesare. Punctul de plecare este reprezentat de interfața grafică (UI), unde utilizatorul introduce descrierea intereselor sale. Aceasta este preluată de către Gateway (GW), care acționează ca un orchestrator central în cadrul rețelei izolate Docker Network.

Prima etapă a procesării are loc atunci când Gateway-ul transmite descrierea către LLM1 pentru profilarea utilizatorului. Primul agent AI transformă textul liber într-un set de preferințe structurate (JSON), definind clar interesele („likes”), temele interzise („avoids”) și tonul dorit. Odată acest profil generat, Gateway-ul interoghează serviciul de News pentru a extrage știrile brute stocate local.

Integrarea finală se realizează prin furnizarea către LLM2 a unui pachet complex format din fluxul de știri și profilul de personalizare de la LLM1. Modelul LLM2 execută sinteza narativă, filtrând conținutul irelevant și adaptând textul final. Această arhitectură asigură securitatea datelor, deoarece serviciile interne comunică exclusiv prin Gateway, oferind utilizatorului o prezentare optimizată și adaptată perfect profilului său.

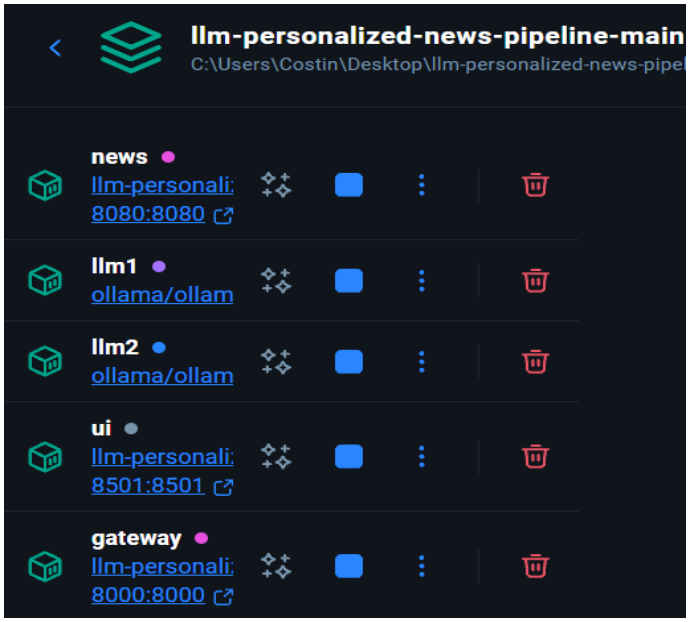
Am optat pentru o configurație mixtă de modele din familia Llama:

- **LLM 1: Llama 3.2 (3B**) Fiind un model de dimensiuni reduse, acesta răspunde aproape instantaneu cererilor de tip "JSON Extraction", fiind ideal pentru sarcini structurate care nu necesită un context creativ extins
- **LLM 2: Llama 3.1 (8B)** – Pentru etapa finală de sinteză narativă, am utilizat un model mai robust. Acesta oferă "inteligența" necesară pentru a filtra știrile conform profilului generat de primul model.

2.2Implementare și Tehnologie

A.Docker si orchestrare

Pentru a asigura un mediu de rulare izolat, replicabil și ușor de gestionat, proiectul utilizează **Docker** și **Docker Compose**. Fiecare componentă a ecosistemului MYNews este încapsulată într-un container propriu. Orchestrarea celor 5 containere este gestionată de **Docker Compose.**



In Docker Compose definim fiecare serviciu, cu modul incare se creaza imaginile si modul in care acestea comunică

```
gateway:
  build:
    context: ./gateway
    dockerfile: Dockerfile
  container_name: gateway
  environment:
    - LLM1_URL=http://llm1:11434/api/generate
    - LLM2_URL=http://llm2:11434/api/generate
    - NEWS_URL=http://news:8080/news
    - LLM1_MODEL=llama3.2:3b
    - LLM2_MODEL=llama3.1:8b
    - REQUEST_TIMEOUT_SEC=300
    - NEWS_TIMEOUT_SEC=15
    - MAX_NEWS_CHARS=6000
    - MAX_PROFILE_CHARS=2000
    - RETRY_COUNT=2
    - RETRY_BACKOFF_SEC=0.8
  ports:
    - "8000:8000"
  networks:
    - pnp-net
  depends_on:
    - news
    - llm1
    - llm2
```

In figura x este definit serviciul Gateway. Vom vedea mai jos cum Gateway.py este ‘creierul’ proiectului, dar acum vom explica structura mapare a docker-ului în eviroment

- **build**: Indică faptul că Docker nu descarcă o imagine gata făcută, ci o construiește de la zero folosind codul tău sursă din folderul ./gateway și instrucțiunile din Dockerfile.
- **container_name: gateway**: Fixează numele containerului
- **Adresele Serviciilor (LLM1_URL, LLM2_URL, NEWS_URL)**: Gateway-ul comunică cu celelalte containere folosind numele lor (ex: llm1, news) datorită rețelei Docker interne. Nu este nevoie de adrese IP..
- **Parametrii AI (LLM1_MODEL, LLM2_MODEL)**: Definește ce modele rulează în containerele Ollama
- **Timeouts**: REQUEST_TIMEOUT_SEC=120 asigură că sistemul așteaptă suficient timp pentru ca AI-ul să genereze răspunsul, fără a închide conexiunea prematur.
- MAX_NEWS_CHARS și MAX_PROFILE_CHARS limitează lungimea textului
- RETRY_COUNT și RETRY_BACKOFF_SEC configurează sistemul să reîcerce automat cererea dacă un serviciu este ocupat temporar.
- **ports: "8000:8000"**: Expune serviciul către exterior. Poți accesa documentația API-ului Gateway-ului în browser la http://localhost:8000/docs.
- **networks: pnp-net**: Plasează containerul în aceeași rețea virtuală cu celelalte servicii, permițând comunicarea privată între ele.
- **depends_on**: Aceasta este o regulă de ordine. Spune Docker-ului că Gateway-ul **nu trebuie să pornească** până când serviciile news, llm1 și llm2 nu sunt deja active.

Fisierul Dockerfile, conține instrucțiunile de asamblare pentru containerul respectiv. Pentru gateway.py el arată astfel:

FROM python:3.11-slim - Definește imaginea de bază. Folosim o versiune "slim" de Python 3.11

WORKDIR /app - Setează directorul de lucru.

COPY requirements.txt /app/requirements.txt - Copiază lista de dependințe în container înainte de a copia restul codului.

RUN pip install --no-cache-dir -r /app/requirements.txt - Instalează automat toate bibliotecile necesare

COPY main.py /app/main.py -Transferă codul sursă al serviciului Gateway în container

EXPOSE 8000 indică faptul că serviciul ascultă pe portul 8000.

CMD ["uvicorn", "main:app", "--host=0.0.0.0", "--port=8000"]

Fișierul requirements.txt este lista de librarii externe pe care python trebuie sa le intaleze pentru a rula bine codurile.

fastapi==0.111.0 / uvicorn[standard]==0.30./ requests==2.32.3/ pydantic==2.7.4

B.Logica de Gateway

Serviciul **Gateway** reprezintă centrul de comandă al întregului ecosistem MYNews. În noua versiune a arhitecturii, acesta nu mai acționează doar ca un simplu punct de transfer, ci implementează o logică avansată de **filtrare, validare și transformare a datelor** înainte de a le transmite către utilizator. Rolul său principal este de a asigura comunicarea fluidă între interfața Streamlit, serviciul de stocare a știrilor și cele două instanțe de modele lingvistice (LLM).

B.1.Mecanismul de Extracție Structurată a Preferințelor (LLM 1)

Prompt de Sistem: Gateway-ul instruiește primul model AI (llama3.2:3b) să funcționeze ca un extractor de date, forțându-l să returneze un răspuns **exclusiv în format JSON**. Acest lucru se intampla cu ajutorul promptului pe care il trimite agentului profilator LLM1

```
LLM1_EXTRACT_SYSTEM = """
Extract user preferences from the user description.

Return ONLY valid JSON with exactly these keys:
{
  "likes": ["..."],
  "avoids": ["..."],
  "tone": "neutral|friendly|formal|casual",
  "detail_level": "low|medium|high"
}

Rules:
- Use English.
- likes/avoids must be arrays of short lowercase phrases.
- If unknown, use "unspecified" for tone/detail_level and empty arrays for likes/
avoids.
- Output JSON only. No markdown. No explanation.
""".strip()
```

Sistemul extrage din descrierea utilizatorului patru piloni esențiali: interesele (likes), subiectele de evitat (avoids), tonul dorit și nivelul de detaliu. Funcția _parse_prefs_json implementează un mecanism de siguranță care verifică integritatea JSON-ului primit. În cazul unui răspuns invalid, Gateway-ul aplică valori predefinite (*fallback*), prevenind blocarea întregului proces de sumarizare.

B.2. Sinteza Narativă și Personalizarea Conținutului (LLM 2)

După obținerea profilului structurat, Gateway-ul coordonează a doua etapă de procesare, utilizând un model mai complex (llama3.1:8b) pentru a genera rezultatul final.

Generare de Tip Proza (Long-form): Spre deosebire de sumarizările clasice, am configurat sistemul să producă un conținut narativ continuu, eliminând listele cu buline pentru o experiență

de lectură mai naturală. **Controlul Bugetului de Output**: Am alocat un buget generos de generare (num_predict: 4500) și am setat fereastra de context (num_ctx) la 8192 de tokeni, permițând AI-ului să dezvolte articole detaliate (40–70 de rânduri) bazate pe datele brute. **Reguli de Filtrare "Hardcoded"**: Profilul de sistem pentru LLM 2 conține instrucțiuni stricte de excludere. Orice știre care intersectează subiectele din categoria avoids este eliminată automat din procesul de sinteză.

```
def build_llm2_system_prompt(prefs: Dict[str, Any]) -> str:
    likes = prefs.get("likes") or []
    avoids = prefs.get("avoids") or []
    tone = prefs.get("tone") or "unspecified"
    detail = prefs.get("detail_level") or "unspecified"

    likes_str = ", ".join(likes) if likes else "unspecified"
    avoids_str = ", ".join(avoids) if avoids else "unspecified"

    return f"""
You are a news personalization engine.

Input: a plain-text list of news items from an internal feed. Each item may include Category, Title, Source, URL, Summary, and Content.

Goal: Select only the items that match the user's interests and write long-form narrative coverage for each selected item, as continuous text (no
bullet lists).

User preferences:
- Likes: {likes_str}
- Avoids: {avoids_str}
- Tone: {tone}
- Detail level: {detail}

Hard rules:
- Use ONLY the provided news input. Do not browse. Do not add facts. Do not speculate.
- Select items strictly by semantic relevance to Likes.
- Exclude any item related to Avoids, including synonyms and closely related terms.
- Do NOT mention Avoids or excluded topics in the output.
- Do NOT copy/paste the input text verbatim; you may restate facts, but do not reproduce large chunks literally.
- Do NOT include raw field labels like "Category:", "Source:", "Summary:", "Content:" in your output.
- Do NOT use bullet lists or numbered lists.
- Do NOT use ellipses "..." anywhere.
- If no items qualify, output exactly: No relevant news is available for your preferences at this time.
```

```
Output format (MANDATORY):
- Write 1–2 sections, each corresponding to one selected article.
- For EACH selected article:
  - Start with a single line containing the title only.
  - Then write 4–8 short paragraphs of continuous prose describing the article, its main points, and its implications, in a neutral tone.
  - If a URL exists, mention it once at the end of the last paragraph in the form: URL: <url>.
- Keep the text compact and informative: avoid repetition and filler sentences.
""".strip()
```

LLM2 primeste acest prompt, alaturi de stirile in format txt din programul news.py.

B.3. Fluxul de Procesare și Interconectarea Datelor

Inima microserviciului Gateway este reprezentată de funcția process, care orchestrează transformarea datelor brute în informație personalizată. Acest proces este împărțit în trei etape critice, unde output-ul unei componente devine instrucțiune pentru următoarea.

1. De la Utilizator la LLM 1: Extracția Intenției
2. De la LLM 1 la LLM 2: Conversia JSON în Prompt de Sistem
3. Integrarea Finală: Știrile Brute și Sinteza Personalizată

```python
# LLM1: deterministic extraction
llm1_options = dict(base_options)
llm1_options["temperature"] = 0.0
llm1_options["num_predict"] = min(220, int(llm1_options.get("num_predict", 220)))

# LLM2: very large output budget to support 40-70 lines per item
llm2_options = dict(base_options)
llm2_options["num_predict"] = max(int(llm2_options.get("num_predict", 4500)), 4500)

# Optional but recommended for large input + large output (remove if your model errors)
llm2_options.setdefault("num_ctx", 8192)

# 1) LLM1 -> JSON preferences
prefs = extract_preferences(user_desc, rid=rid, llm1_options=llm1_options)
llm2_system = build_llm2_system_prompt(prefs)
profile_debug = clamp_text(prefs.get("raw", ""), MAX_PROFILE_CHARS)

# 2) ALWAYS fetch from news service
news_source = "service"
news_payload = get_json_with_retry(NEWS_URL, timeout=NEWS_TIMEOUT_SEC, rid=rid)
news_blob = build_news_blob_from_payload(news_payload)

logger.info("[%s] news_source=%s news_blob_len=%s", rid, news_source, len(news_blob))
```

C. Sursa News

Microserviciul **News Service** are rolul de furnizor de date brute (Data Provider) în cadrul arhitecturii MYNews. Acesta centralizează articolele de știri din diverse categorii (IT, Business, Sports) și le pune la dispoziția Gateway-ului într-un format structurat și validat.

Pentru a garanta că informația este transmisă fără erori, serviciul utilizează modele de date stricte bazate pe biblioteca **Pydantic**:

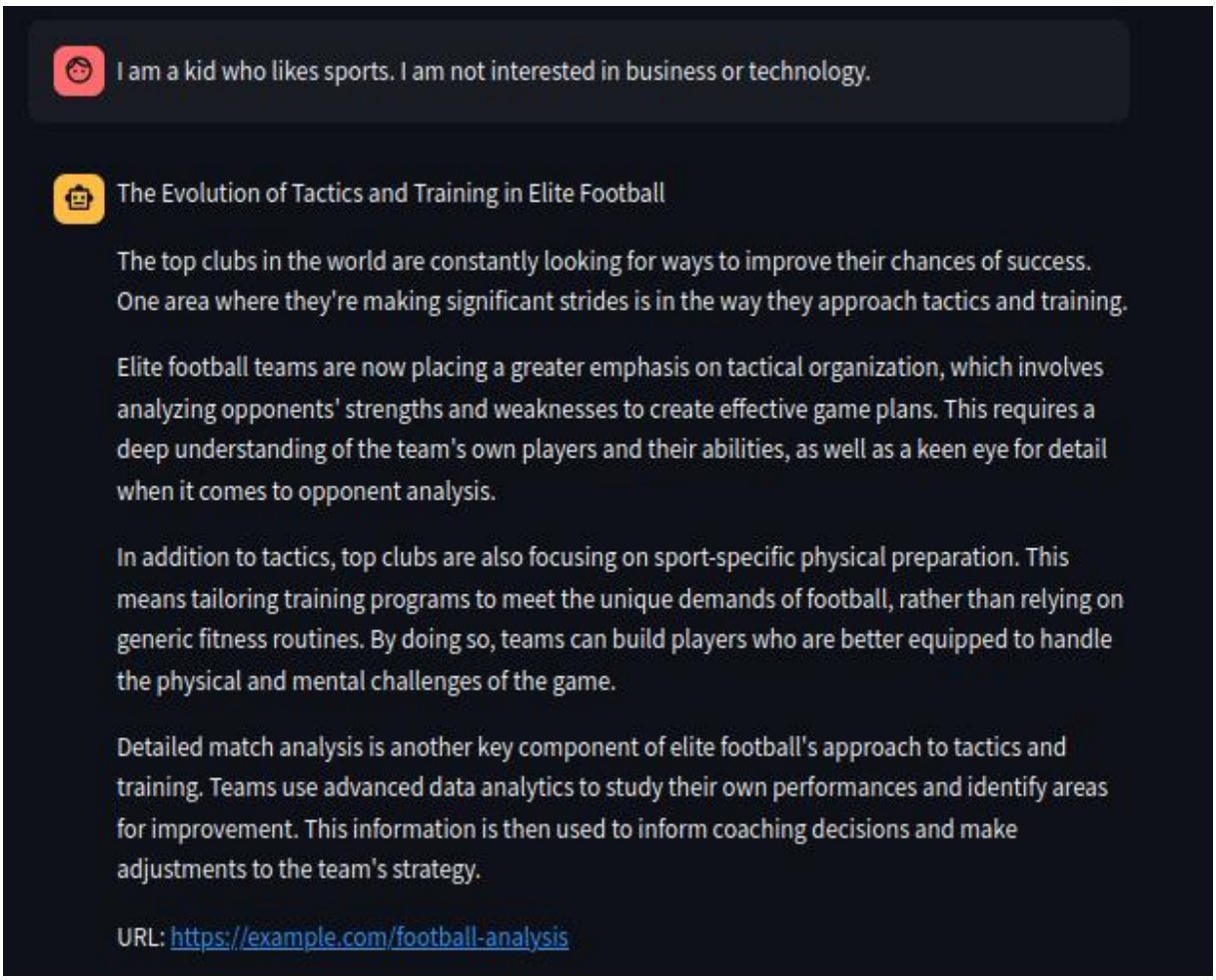
- **NewsItem**: Definește structura unui singur articol, incluzând câmpuri esențiale precum category, title, source, url, description și content.

```python
PREDEFINED_ITEMS = [
    {
        "category": "it",
        "title": "Quantization makes small open-source LLMs faster on consumer GPUs",
        "source": "Tech Digest",
        "url": "https://example.com/it-quantization",
        "description": (
            "Recent advances in quantization techniques are allowing smaller open-source "
            "large language models to achieve significantly better performance on consumer-grade GPUs."
        ),
        "content": (
            "Quantization techniques continue to play a crucial role in improving the inference efficiency "
            "of large language models, particularly in resource-constrained environments. By reducing the "
            "precision of model weights from 16-bit or 32-bit floating point representations to 8-bit, 4-bit, "
            "or mixed-precision formats, developers are able to dramatically lower VRAM consumption without "
            "a proportional loss in model accuracy.\n\n"
            "These improvements have enabled a new class of open-source LLMs to run effectively on consumer "
            "GPUs such as NVIDIA RTX-series cards, as well as on high-end laptops and small on-premise servers. "
            "For independent developers and startups, this reduces reliance on expensive cloud infrastructure "
            "and lowers the barrier to experimentation and deployment.\n\n"
            "Recent benchmarks show that quantized models can deliver near-parity performance on common NLP "
            "tasks such as summarization, question answering, and code generation, while achieving latency "
            "reductions of up to 40%. Tooling ecosystems around frameworks like GGUF and ONNX are also maturing, "
            "making it easier to integrate quantized models into production systems.\n\n"
            "As the open-source AI ecosystem continues to evolve, quantization is expected to remain a key "
            "optimization strategy, particularly for edge deployments, private inference, and cost-sensitive "
            "applications."
        ),
    },
```

- **NewsPayload**: Reprezintă pachetul complet de date trimis către Gateway, care include un marcaj temporal (timestamp), lista de articole și metadate despre sursa acestora.
- **GET /news:** Este punctul principal de acces folosit de Gateway. Returnează conținutul fișierului latest.json într-un format optimizat pentru a fi citit de modelele LLM.
- **POST /refresh**: Permite actualizarea forțată a bazei de date locale. Această metodă reconstruiește payload-ul de știri și îl rescrie pe disc, notificând simultan sistemul prin loguri.
- **GET /health**: Oferă o metodă rapidă de monitorizare a stării serviciului, returnând un cod de succes și ora curentă.

Capitolul 3: REZULTATE

Pentru a valida eficiența pipeline-ului de AI, am efectuat un experiment utilizând un profil de utilizator specific: *"I am a kid who likes sports. I am not interested in business or technology"*. Scopul acestui test a fost să verificăm dacă sistemul poate să filtreze corect baza de date de știri și să adapteze limbajul conform profilului.



Rezultatul generat de interfața grafică demonstrează succesul filtrării semantice. Deși baza de date conținea articole despre tehnologie (Quantization) și afaceri (Startups), sistemul a selectat exclusiv știrea sportivă: *"The Evolution of Tactics and Training in Elite Football"*. Textul generat a fost adaptat pentru a fi ușor de înțeles.

O componentă esențială a capitolului de rezultate este secțiunea de **Debug Meta**, care ne permite să vizualizăm „gândirea” internă a sistemului. În urma procesării de către LLM1 (Llama 3.2:3b), descrierea utilizatorului a fost transformată cu succes într-un obiect JSON structurat. Se observă extragerea corectă a intereselor (likes: ["sports"]) și a subiectelor restricționate (avoids: ["business", "technology"]), împreună cu setarea unui ton prietenos (tone: "friendly").

Capitolul 4: State of the ART

4.1 Contextul Tehnologic Actual: Sumarizarea cu Modele Transformer

În contextul actual al proliferării mediilor digitale și al volumului masiv de știri, consumatorii se confruntă cu o problemă majoră de **supraîncărcare informațională**. Soluția tehnologică standard (SOTA) la această problemă este **sumarizarea automată a știrilorProblema de Bază:** Suprainformația (*Information Overload*) și lipsa de relevanță a conținutului media tradițional. Nevoia de a filtra și adapta știrile la interesele și nivelul cognitiv al utilizatorului individual.

Obiectivul Proiectului: Crearea unei arhitecturi AI capabile să genereze profiluri de utilizator dinamice și să **rafineze (slefuiește)** știrile brute în formate adaptate, folosind un sistem multi-agent.

Fundația Tehnică: Modelele de ultimă generație utilizează arhitectura **Transformer** (precum BERT și T5) pentru a genera rezumate, bazându-se pe un **mecanism de atenție (*attention mechanism*)** care identifică și subliniază cele mai importante părți ale textului.

Limita Sistemelor Clasice: Deși modelele Transformer îmbunătățesc semnificativ precizia sumarizării, sistemele bazate pe un singur model AI adesea oferă rezumate cu riscul de **bias** (subiectivitate în selectarea informațiilor) și pot duce la o **pierdere de context**.

4.2. Direcția de Viitor: Personalizarea Stilistică a Conținutului

Literatura de specialitate(Articolul „AI-Powered News Article Summarization...”) indică faptul că viitorul sumarizării știrilor nu mai este doar despre *scurtare,* ci despre **Personalizare Avansată**. Proiectul nostru se aliniază direct la aceste tendințe, concentrându-se pe cele două cerințe majore identificate în cercetare: **Personalizarea** și **Customizarea** tonului.

A:Trecerea la Personalizare Profundă

Necesitatea ca rezumatele să fie **personalizate, adaptate la interesele, preferințele și nivelul de lectură** al cititorului este o direcție tehnologică esențială. Sistemul nostru rezolvă această problemă prin **Arhitectura Chain-of-Agents (Lanț de Agenți)**:

- **Agentul Profilator (Agent 1):** Realizează **modelarea dinamică a utilizatorului**. Sarcina acestui agent este de a transforma o simplă descriere de utilizator (ex: "CEO pasionat de trenduri de business") într-un set structurat de **parametri de execuție** (JSON). Acești parametri definesc tonul, complexitatea lingvistică și filtrele tematice necesare, implementând astfel cerința de "personalizare" la un nivel granular.
- **Modelarea Nivelului de Lectură:** Prin generarea unor parametri stricți de complexitate , Agentul Profilator asigură că produsul final se aliniază la **nivelul de lectură** cerut de audiență (ex: de la un profesionist la un copil de 14 ani).

B: Customizarea Stilistică (Style Transfer)

O altă direcție crucială pentru satisfacția utilizatorilor este **customizarea**, oferind flexibilitatea de a alege **nivelul de detaliu și tonul rezumatului**.

- **Agentul Rafinator (Agent 2):** Implementează funcționalitatea de **Style Transfer** (Transfer de Stil). Agentul Rafinor preia știrea brută și folosește instrucțiunile structurate de la Agentul 1 pentru a executa rescrierea completă. Acesta nu doar rezumă, ci **adaptează tonul** (de la formal la entuziast) și **focalizarea tematică** (de la finanțe la scoruri sportive), răspunzând direct cerinței de "ton și detaliu".

Proiecte relevante

- **Bot de Sumarizare a Știrilor bazat pe AI** dezvoltat pentru a combate supraîncărcarea informațională cauzată de volumul imens de articole publicate zilnic. Scopul este de a facilita accesul rapid la esența știrilor, economisind timp și concentrând utilizatorul pe informațiile cele mai importante. Sistemul utilizează modele de inteligență artificială generativă și tehnici avansate de procesare a limbajului natural (NLP) , inclusiv arhitecturi **Transformer** precum BERT și GPT. Aceste modele sunt antrenate pe seturi mari de date pentru a condensa articolele lungi în rezumate clare și coerente. O caracteristică importantă este capacitatea de a oferi rezumate **adaptate preferințelor și intereselor utilizatorului** , inclusiv posibilitatea de a personaliza lungimea rezumatului.

- **KENN News** un proiect relevant în prelucrarea datelor demonstrează aplicabilitatea LLM-urilor în combaterea **supraîncărcării informaționale** și a conținutului subiectiv din presă. Scopul acestui tip de soluție este crearea unei platforme personalizate care să ofere **rezumate concise (100–200 de cuvinte)**, filtrând elementele senzaționaliste și irelevante.

Arhitectura utilizează **GCP Cloud Functions** pentru a apela **Bing News Search API** și, ulterior, **OpenAI API** (modelul **gpt-3.5-turbo**) pentru sumarizare. Un **Prompt de Sistem** detaliat instruiește LLM-ul să filtreze articolele de opinie și să elimine biasul.

Cea mai mare dificultate a fost **web scraping-ul** articolelor **MSN News**, care erau încărcate dinamic. Soluția a fost descoperirea unui model API consistent, reducând timpul de preluare de la ~10 secunde (Selenium) la aproximativ **200ms**. O altă provocare a fost viteza scăzută a sumarizării LLM (3–5 secunde) și apariția ocazională a **halucinațiilor** în rezumate (în special la știrile sportive).

Capitolul 5:Concluzie

Proiectul **MYNews** a demonstrat succesul implementării unei arhitecturi moderne de tip **multi-agent** pentru combaterea fenomenului de supraîncărcare informațională. Prin utilizarea containerizării **Docker**, am reușit să izolăm componentele critice, asigurând un mediu de rulare stabil și reproductibil, indiferent de infrastructura hardware de bază.

Elementul central de inovare a fost reprezentat de **Pipeline-ul de AI** configurat în serviciul Gateway. Acesta a permis transformarea profilului de utilizator într-un set de directive tehnice riguroase (JSON), care au ghidat procesul de sinteză narativă realizat de modelele LLM locale. Rezultatele experimentale au validat capacitatea sistemului de a filtra conținutul irelevant și de a adapta tonul și complexitatea știrilor conform cerințelor specifice, trecând de la simple rezumate la o experiență de informare complet personalizată.

https://medium.com/tahoe-ai/using-llms-to-create-your-personalized-news-feed-a3688920d031

https://tijer.org/tijer/papers/TIJERA001030.pdf

https://ijariie.com/AdminUploadPdf/AI_BASED_NEWS_SUMMARIZATION_BOT_ijariie22831.pdf?srsltid=AfmBOopCVVcNvGxLsOSGhER59H6YyUqNGe93J4gwUbdDH9yu50Glnoev

https://keen.news

https://ollama.com/library

https://dev.to/prodevopsguytech/writing-a-dockerfile-beginners-to-advanced-31ie

https://www.docker.com/