```
\documentclass[12pt,a4paper]{article}
\usepackage[utf8]{inputenc}
\usepackage[catalan]{babel}
\usepackage[margin=2.5cm]{geometry}
\usepackage{inconsolata} % Font monoespaiada agradable
\usepackage{hyperref}
\hypersetup{colorlinks=true, linkcolor=MidnightBlue, urlcolor=MidnightBlue}
\usepackage{fancyvrb} % per mostrar codi

\title{PROJECTE ASSIMOV: Un Manifest per a Educadors en l'Era dels Transformers}
\author{Dídac Valenciano Gener}
\date{maig del 2025}

\begin{document}
\section*{ L'any 2025 és —i serà— l'any ChatGPT}.
```

A l'Europa ibèrica i mediterrània dins la UE, amb un subconscient meca latent que avui encara es nota, *Mazinger Z* va deixar una profunda empremta cultural, emocional i en diversos altres àmbits que explica alguns comportaments compartits de la *Generació X*. Si tens entre 45 i 60 anys, ja has pensat en "*¡Puños fuera!*", has recordat com vas sobreviure al *Blandiblú*, als *Clicks* de *Famobil*, i a veure dibuixos violents sense convertir-te en un psicòpata.

Als Estats Units, *Hasbro* inicia el relleu generacional amb la saga *Transformers*, que culmina culturalment l'any 2007 quan, just arrencar el metratge, els *Decepticons* ataquen una base militar nord-americana al desert de *Qatar*. Mentrestant, *Megan Fox* i *Shia LaBeouf* ens deixen sense alè. És en aquest moment quan una nova imatge s'enganxa al subconscient d'una altra generació: un petit *Decepticon* —un *transformer*, però dels dolents—, amagat entre les pertinences dels protagonistes, absorbeix metall de l'entorn per modificar-se, estirant les seves formes, perllongant-se i adaptant-se fins a assolir el seu objectiu.

Deu anys després, fruit d'un llarg recorregut d'investigació per part de múltiples entitats acadèmiques i privades, en paral·lel al món de l'entreteniment, *Google* presenta públicament l'any 2017 una tecnologia algorítmica que es replica, s'estira i amplia la seva pròpia estructura, modificant-se i perllongant-se fins que, de manera purament probabilística, aconsegueix assolir el seu objectiu. Quina sorpresa —i quina coincidència— que aquesta tecnologia, desconeguda fins aleshores, rebés el nom de: *transformers*.

*"Attention is All You Need"*, publicat per *Google* el 2017, va crear rebombori —especialment a *Twitter* (o *Grok*, o com es digui aquesta setmana). Però no va ser fins que *OpenAI* va agafar el relleu que ens van col·locar al davant, literalment, un petit *transformer*. Un que, cada vegada que li parlam, *tokenitza* el nostre llenguatge, i com aquell *Decepticon* al desert de *Qatar*, s'estira, s'adapta i reorganitza la seva forma per generar, de manera purament probabilística, una resposta que optimitzi la situació que té al davant.

I la gràcia, amics meus, és que ho simula tan bé, que si entenem per *intel·ligència artificial* una simulació funcional d'intel·ligència... bé, potser hi ha altres implementacions, però aquesta —definitivament— n'és una.

El que ve tot seguit és una altra implementació. Una que, com els algoritmes d'*OpenAI*, s'estira, s'adapta i reorganitza la seva pròpia estructura per generar, de manera purament probabilística, una resposta òptima a la situació que té al davant: crear de manera eficient contingut didàctic, un article, un llibre. Tot plegat perquè nosaltres, *professors*, puguem disposar de les nostres pròpies armes per a la batalla que ens espera.

## AI-enhanced LaTeX generation pipeline

github.com/nassaba/project-assimov - AI-enhanced content generation pipeline

```
# Phase 0: Interpret prompt and assign roles with gramaneute + llm_router

# Phase 1: Generate structural skeleton with yaml_generator using Claude or o3 or ...

# Phase 2: Expand each section with role-assigned LLMs (GPT-4o, MythoMax, Lit-6B...) using cached YAMLs

# Phase 3: Review narrative coherence, argument and tone with o3 or Claude using section context

# Phase 4: Compile formatted output into .tex with writer or translator

# Phase 5 (optional): Final stylistic polish by purist LLM or human reviewer

# CONFIGURATION
# Load API key (replace with your secure method)

import openai
from pathlib import Path
import time


# AI-enhanced content generation pipeline (Assimov)

def phase_0_prompt_intake():
    """Receive user prompt and determine functional roles + LLMs."""
    prompt = user_input()
    roles = assign_roles(prompt)  # via gramaneute.py -> llm_router.py
    llms = select_models(roles, config="config.yaml")
    return plan(roles, llms)

def phase_1_generate_structure(plan):
    """Create YAML skeleton per section with titles, themes, targets."""
    yamls = []
    for section in plan.sections:
    yamls.append(generate_yaml(section))  # via yaml_generator.py or real LLM
    return yamls

```

```python
def phase_2_expand_sections(yamls):
    """Expand each section via assigned LLM, using cached context."""
    content = []
    for yaml in yamls:
        llm = yaml.assigned_model
        section_text = expand_from_yaml(yaml, llm=llm, use_cache=True)
        content.append(section_text)
    return content

def phase_3_refine_coherence(content):
    """Polish narrative flow, argument structure, and tonal coherence."""
    polished = []
    for section in content:
        coherent = enforce_coherence(section)  # coherence supervisor
        refined = polish_argument(coherent)
        toned = adjust_tone(refined)
        polished.append(toned)
    return polished

def phase_4_compile_tex(polished):
    """Format polished content into a .tex document with structure."""
    doc = initialize_tex()
    for section in polished:
        doc.append(format_section(section))  # via writer.py or translator.py
    return doc

def phase_5_final_review(tex_document):
    """Optional: stylistic and poetic pass by a purist LLM or human."""
    reviewed = manual_review(tex_document)
    return reviewed

# Main pipeline execution

if __name__ == "__main__":
    plan = phase_0_prompt_intake()
    yamls = phase_1_generate_structure(plan)
    raw_content = phase_2_expand_sections(yamls)
    refined = phase_3_refine_coherence(raw_content)
    tex = phase_4_compile_tex(refined)
    final_output = phase_5_final_review(tex)
    save(final_output, "output/document.tex")
```