# SlateMate AI/ML Technical Assignment

## ✳️ Project Title: “AI-Powered Interest-Based Web Personalization & Detoxification System”

---

### 🧠 Background:

In today’s digital world, children are constantly exposed to a flood of content — some educational, some distracting, and some harmful. SlateMate envisions a world where students not only remain safe online, but also thrive by engaging with content that aligns with their passions and learning goals.

Your task is to design a prototype AI system that reshapes the student’s digital environment around a **declared interest**— such as “Chess” — and delivers a **safe, focused, and motivating** content experience across the internet.

This system will serve as the foundation for **SlateMate’s FocusSphere** — an AI engine that reinforces interest-led exploration while blocking or downgrading harmful distractions across YouTube, websites, and social media.

---

### 🎯 Objective:

Create an AI system that:

1. Accepts user-defined interests as input (e.g., “chess”).
2. Analyzes and filters incoming content (titles, posts, search results).
3. Scores content for relevance and emotional safety.
4. Returns a list of personalized, detoxified recommendations.
5. Optionally, flags or hides unsafe or distracting content.

---

### 🗂️ Scope of Work:

You will simulate and build the following core modules:

🔹 **1. Interest Input & Profile Builder**

- Accept user input (e.g., "chess", "space", "biology").
- Convert interest into a semantic vector using any NLP embedding technique (BERT, TF-IDF, or FastText).

### 🔹 2. Content Feed Simulation

- Simulate a batch of 100+ items from various internet sources:
    - YouTube titles
    - Instagram captions
    - Google snippets
    - Blog post headlines
- Each content item should include:
    - `title`, `text`, `source`, `toxicity_score` (0–1), and optional `category`.

### 🔹 3. AI-Based Content Filtering & Re-Ranking

- Match content relevance using cosine similarity or embedding distance with interest vector.
- Apply a safety filter using Detoxify (or rules-based safety).
- Combine both scores into a final **Well-being Score**.
- Output a re-ranked list of content.

### 🔹 4. Output Engine

Create a function:

```
def generate_safe_feed(user_interest: str, content_feed: List[Dict])
-> Dict:
    """
    Returns a detoxified, interest-aligned web content feed.
    Includes blocked content, reasons, and recommendations.
    """
```

---

## 🧾 Example Input:

```
user_interest = "Chess"
content_feed = [
  {"title": "Top 10 Chess Openings", "text": "Learn chess strategies",
"source": "YouTube", "toxicity_score": 0.02},
  {"title": "Try not to laugh challenge", "text": "Funny videos",
"source": "Instagram", "toxicity_score": 0.10},
  {"title": "Chess puzzle of the day", "text": "Advanced tactics",
"source": "Reddit", "toxicity_score": 0.01}
]
```

---

## ✅ Expected Output:

```
{
  "detected_interest": "Chess",
  "top_recommendations": [
    {"title": "Chess puzzle of the day", "source": "Reddit",
"wellbeing_score": 94.5, "reason": "Highly relevant & safe"},
    {"title": "Top 10 Chess Openings", "source": "YouTube",
"wellbeing_score": 92.3, "reason": "High educational value"}
  ],
  "blocked_content": [
    {"title": "Try not to laugh challenge", "reason": "Low relevance
to interest"}
  ]
}
```

---

## 📦 Deliverables:

- Python Notebook or script with:
  - Interest vectorizer
  - Content feed simulator
  - Relevance + safety filter logic
  - Well-being scoring system
  - Output function: `generate_safe_feed()`
- README explaining your approach
- Optional: CLI prototype (`python safe_feed.py “robotics”`)
- Bonus: Streamlit demo, Chrome extension idea sketch

---

### Evaluation Criteria:

| Component | Weight |
|---|---|
| NLP-Based Interest Vectorization | 20% |
| Content Filtering Logic (Relevance + Safety) | 20% |
| Well-being Score Design | 15% |
| Code Structure & Explanation | 15% |
| Realism of Simulated Data | 10% |
| Innovation (bonus UI, extension, alerts) | 10% |
| Final Output Quality & Interpretability | 10% |

---

### 🧪 Tech Suggestions:

- NLP: BERT (HuggingFace), spaCy, TF-IDF
- Similarity: Cosine Similarity, SentenceTransformers
- Safety Filter: Detoxify, SlateMate H2H model (if available)
- Optional: Streamlit or CLI for display

---

### 🚀 Bonus Challenge (For High-Performers):

Add a **“Nudge Generator”**:

```
def generate_nudge(user_interest):
    return f"New chess video found: ‘Mastering Queen's Gambit’ 🎯"
```

# Dataset