Visual Explanation Prototype

Project name: Infooware Edu Prototype — PDF to Slides & Short Video
Assignee: Intern

Project goal (single line)

Convert an input PDF (one chapter / article) into:

1. A short slide deck (PowerPoint / PDF/ series of visuals) that visually explains the key concepts, and

2. A short animated explainer video (30sec)

Key deliverables

1. **Code repository** (GitHub) with clear README and usage instructions.

2. **Script / CLI** that accepts a PDF and outputs:

    - `output/slides.pptx` (or `slides.pdf`) — 6–12 slides, each slide: title, 1–2 short bullets, 1 visual (image/diagram/icon).

    - `output/video.mp4` — 30–90s MP4, animated sequence of the slides with narration (TTS) and simple transitions.

3. **Example run**: run on the provided sample PDF and commit resulting slides + video to repo.

4. **Short demo doc** (1 page) describing how the pipeline works and how to run it locally.

Functional scope (what to implement)

1. **PDF ingestion & text extraction**

- Extract headings & paragraphs from a single PDF file.
- Identify top N (e.g., 6–10) key points / sections to become slides.

2. **Content summarization & slide text**
    - Shorten each key point to a 6–20 word slide headline + 1–2 supporting bullets.
    - Produce speaker notes (one sentence per slide) for narration.
3. **Visual generation / selection**
    - For each slide, either:
        - Generate a simple illustrative image (icon/illustration) programmatically or
        - Select a related royalty-free image from a bundled set (no web downloads required).
    - Create a consistent slide layout (title, visual, bullets).
4. **Slide assembly**
    - Build a PPTX (recommended) or export to PDF.
5. **Video generation**
    - Convert slides into a video with:
        - Per-slide duration (e.g., 5–12s), transitions, and panning/zoom (Ken Burns) effect.
        - Text-to-speech narration from speaker notes (or simple on-screen captions if TTS unavailable).
        - Background music (royalty-free loop) at low volume.
    - Output MP4.
6. **CLI / Script**
    - `python run_pipeline.py --input sample.pdf --outdir output/` or equivalent.

Suggested tech stack & libraries

- Language: **Python**
- PDF parsing: `pdfplumber` or `PyPDF2` / `pdfminer.six`
- NLP summarization / keyphrase extraction: `transformers`, `sentence-transformers`, or simple `nltk`/`sumy` for quick prototype
- Slide creation: `python-pptx`
- Image generation/selection: placeholder images or `Pillow` to compose simple diagrams; (optional) use local icon set (SVG/PNG)
- TTS: `pyttsx3` (offline) or cloud TTS if available (must provide credentials separately)
- Video assembly: `moviepy`
- Packaging: requirements.txt, README with run steps

> Note: If cloud models (LLM/TTS) are used, document auth keys and make steps optional in README.

(Adjust based on intern availability.)

Acceptance criteria (how I will evaluate)

- Running the CLI on the provided sample PDF produces `slides.pptx` with 6–12 coherent slides and `video.mp4` playable in standard players.
- Slides accurately reflect the document's main points (not hallucinated).
- Repo includes: README with steps, requirements, and one-sentence architecture diagram; a sample input PDF and the produced outputs.
- Code is modular, commented, and runnable.

Repo & submission format

- Single GitHub repo with:
    - `README.md` (how to run, dependencies)