ZEROCODE AI/ML INTERNSHIP ASSIGNMENT

Submitted by: *Samyukta Gade*
Role Applied: AI/ML Intern

Task 1: Multimodal Identity Extractor

➢ **Inputs:**

- logo.png — Company logo image
- persona.txt — Short description of target user persona
- docpdf — Official brand style guide in PDF format

➢ **Tools & Libraries Used:**

- [easyocr] – for extracting text from the logo image
- [pdfplumber] – for reading textual content from the PDF
- [CLIP (openai/clip-vit-base-patch32)] – to embed visual and textual inputs into a shared semantic space
- [sentence-transformers (all-MiniLM-L6-v2)] – for encoding the persona into a semantic text vector
- Python (with NumPy, PIL, Transformers)

➢ **Logic:**

1. Extracted raw text from the logo image using OCR.
2. Parsed the full text from the brand's PDF guide using pdfplumber.
3. Encoded both the logo + PDF text using CLIP, and the persona using SentenceTransformer.
4. Matched the dimensions of the output vectors using zero-padding.
5. Averaged the vectors to form one **unified brand identity vector**.

➢ **Output:**

- output/identity_vector.json
  Contains a numeric list — the **final brand identity embedding** combining visual, textual, and strategic inputs.
  This vector can be used for tasks like brand matching, clustering, UI recommendation systems, etc.

Task 2: Visual Theme Interpreter & Config Generator

➢ **Inputs:**

- theme.png — UI mood board or visual reference image
- Prompt: *“Futuristic neon vibe for interactive UI”*

➢ **Tools & Libraries Used:**

- OpenCV – for reading and reshaping image data
- KMeans (from scikit-learn) – to extract the 5 dominant RGB colors
- flan-t5-base (via Hugging Face pipeline) – to generate a text-based config based on colors + prompt

➢ **Logic:**

1. Read and flatten the image to pixel RGB values.
2. Applied KMeans clustering to detect 5 most dominant colors.
3. Converted them to HEX codes like #ff1493, #00ffee, etc.
4. Constructed a natural language prompt combining the extracted colors and intended UI theme.
5. Passed the prompt into the Flan-T5 model to generate a JSON-style, creative theme configuration.

➢ **Output:**

- output/theme_config.json
  Contains a **theme description string** that explains how the extracted color palette can be applied to design elements (background, buttons, typography, etc.) in a UI.

Design Highlights

- **Multimodal Processing**: Combined PDF parsing, OCR, and computer vision in one pipeline (Task 1).
- **Use of Embedding Models**: CLIP + MiniLM helped generate meaningful, comparable semantic vectors.
- **Text-to-Theme Generation**: Used generative AI to simulate how a designer might interpret a color moodboard into a usable UI configuration.

- **Modular Code**: All scripts are reusable, cleanly structured, and produce reproducible JSON outputs.

Project Structure Overview

```
Zerocode/
 ├── task1.py
 ├── task2.py
 ├── doc.pdf
 ├── requirements.txt
 ├── README.md
 ├── .gitignore
 ├── sample_inputs/
|    ├── logo.png
|    ├── theme.png
|   └── persona.txt
└── output/
    ├── identity_vector.json
   └── theme_config.json
```