🩺 Health AI: Code Analysis & Generation Platform

1. Project Overview

Health AI is a web-based application that assists developers and analysts in analyzing software requirements and generating code using generative AI. It leverages IBM Granite's large language model and provides a user-friendly interface via Gradio.

---

2. Objectives

- Automate extraction and classification of software requirements from text or PDF documents.
- Generate structured code snippets in multiple programming languages.
- Provide an accessible interface for non-technical users to interact with AI-powered development tools.

---

3. Technologies Used

Component	Technology/Library	Purpose
Language Model	*ibm-granite/granite-3.2-2b-i*	Natural language understanding & code generation
Tokenization	*Transformers (*Hugging Face*)*	Text preprocessing and model interfacing
Inference Engine	*torch*	Model execution and tensor operations
PDF Parsing	*PyPDF2*	Extracting text from uploaded PDF documents
UI Framework	*Gradio*	Building interactive web interface

4. Functional Modules

4.1 Requirement Analysis

Purpose: Extract and categorize software requirements from user input or uploaded PDF.

Features:

- Accepts PDF or free-text input.
- Classifies requirements into:
  - Functional Requirements
  - Non-Functional Requirements
  - Technical Specifications
- Displays structured output.

4.2 Code Generation

Purpose: Generate code snippets based on user-defined requirements.

Features:

- Accepts textual description of desired functionality.
- Supports multiple programming languages.
- Displays generated code in a readable format.

---

5. User Interface

Built using Gradio's Blocks, the UI is divided into two tabs:

Tab Name	Features
Code Analysis	PDF upload, text input, analysis button, structured output display
Code Generation	Requirement input, language dropdown, generate button, code output

---

6. Model Configuration

- Detects GPU availability and adjusts torch_dtype and device_map accordingly.
- Ensures proper padding token setup for consistent generation.

---

7. Installation & Launch

```
pip install transformers torch gradio PyPDF2 -q

app.launch(share=True)
```

---

8. Potential Extensions

- Support for additional file formats (e.g., DOCX, TXT).
- Domain-specific fine-tuning (e.g., healthcare, finance).
- Code validation and testing within the interface.
- Export options for requirement reports and generated code.

---

9. License & Attribution

This project uses open-source libraries and IBM Granite models under their respective licenses. Ensure compliance with usage terms when deploying or distributing.