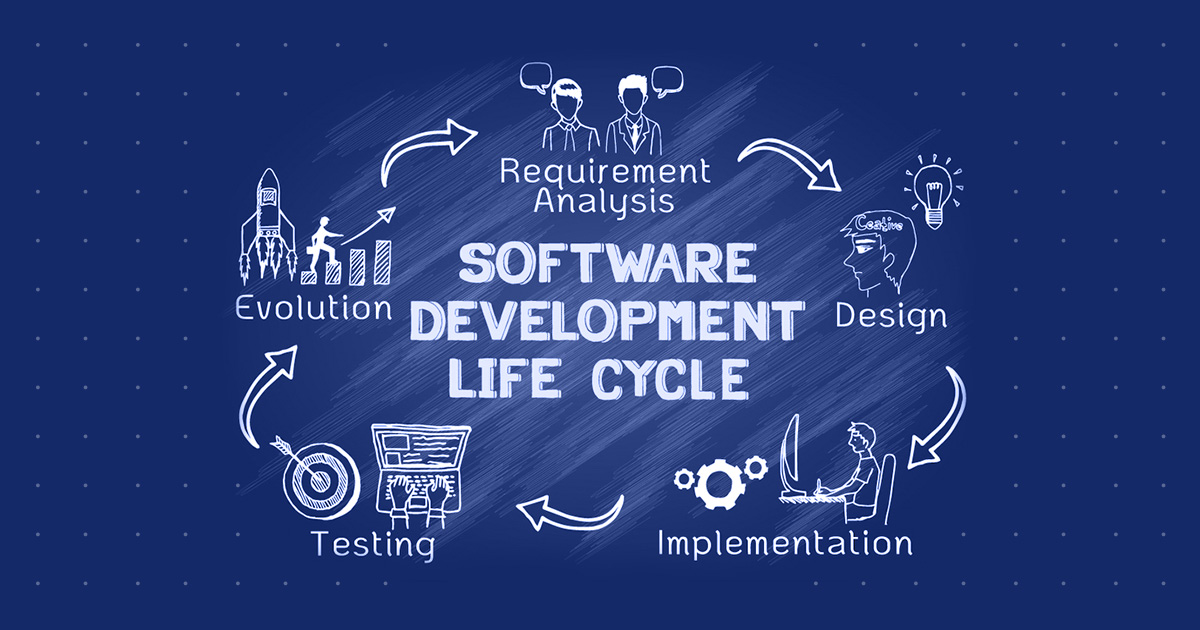
**SmartSDLC – AI-Enhanced Software Development Lifecycle**

***Generative AI with IBM***



**SmartSDLC**

***1. Introduction***

***• Project Title: SmartSDLC***

* ***Team Leader : Sri Dharshini S***
* ***Team member : Theepa Lakshmi V***
* ***Team member : Praveena R***
* ***Team member : Ragavi S***

***2. Project Overview***

***The Eco Assistant & Policy Analyzer is an AI-driven application that helps users:***

1. ***Generate Eco-Friendly Tips: Provides practical and actionable sustainable living suggestions based on environmental problem keywords.***
2. ***Summarize Policy Documents: Extracts key points, provisions, and implications from environmental and sustainability-related policy documents (uploaded as PDF or pasted text).***

***This project integrates IBM Granite (LLM model) with Gradio UI to provide a simple and interactive interface for both individuals and policymakers.***

***3. Architecture***

***Components:***

1. ***Model & Tokenizer***
   * ***Uses Hugging Face’s AutoTokenizer and AutoModelForCausalLM with IBM Granite model (ibm-granite/granite-3.2-2b-instruct).***
   * ***Runs on CPU/GPU with PyTorch backend.***
2. ***Core Functions***
   * ***generate\_response(prompt): Generates AI-powered responses from user prompts.***
   * ***extract\_text\_from\_pdf(pdf\_file): Reads and extracts text from uploaded PDF files using PyPDF2.***
   * ***eco\_tips\_generator(problem\_keywords): Produces eco-friendly living suggestions.***
   * ***policy\_summarization(pdf\_file, policy\_text): Summarizes policy documents.***
3. ***Gradio Interface***
   * ***Tab 1: Eco Tips Generator → User enters keywords (e.g., solar, plastic, energy saving).***
   * ***Tab 2: Policy Summarization → User uploads PDF or pastes policy text to get a concise summary.***

***4. Setup Instructions***

***Prerequisites***

* ***Python ≥ 3.9***
* ***pip package manager***
* ***GPU with CUDA (optional for faster inference)***

***5. Folder Structure***

***EduTutor-AI/***

***│── app.py # Main application script***

***│── requirements.txt # Dependencies***

***│── docs/ # Documentation files***

***│── models/ # Pretrained model references***

***│── utils/ # Helper functions (if extended later)***

***6. Running the Application***

***- Launch the Gradio interface by running app.py***

***- Navigate between Concept Explanation and Quiz Generator tabs***

***- Input the desired topic and view the AI-generated output in real time***

***7. Authentication***

***Currently, the project runs locally with no authentication.  
If deployed (e.g., on cloud or Hugging Face Spaces), authentication can be added using:***

* ***API Key-based Authentication (restrict model access).***
* ***Gradio Auth (username/password login).***
* ***OAuth (Google/GitHub login) for enterprise use.***

***8. User Interface***

***Built with Gradio Tabs:***

***Tab 1 – Eco Tips Generator***

* ***Input: Keywords (e.g., "solar", "plastic", "energy saving").***
* ***Output: Sustainable living tips (bulleted, structured).***
* ***Button: Generate Eco Tips.***

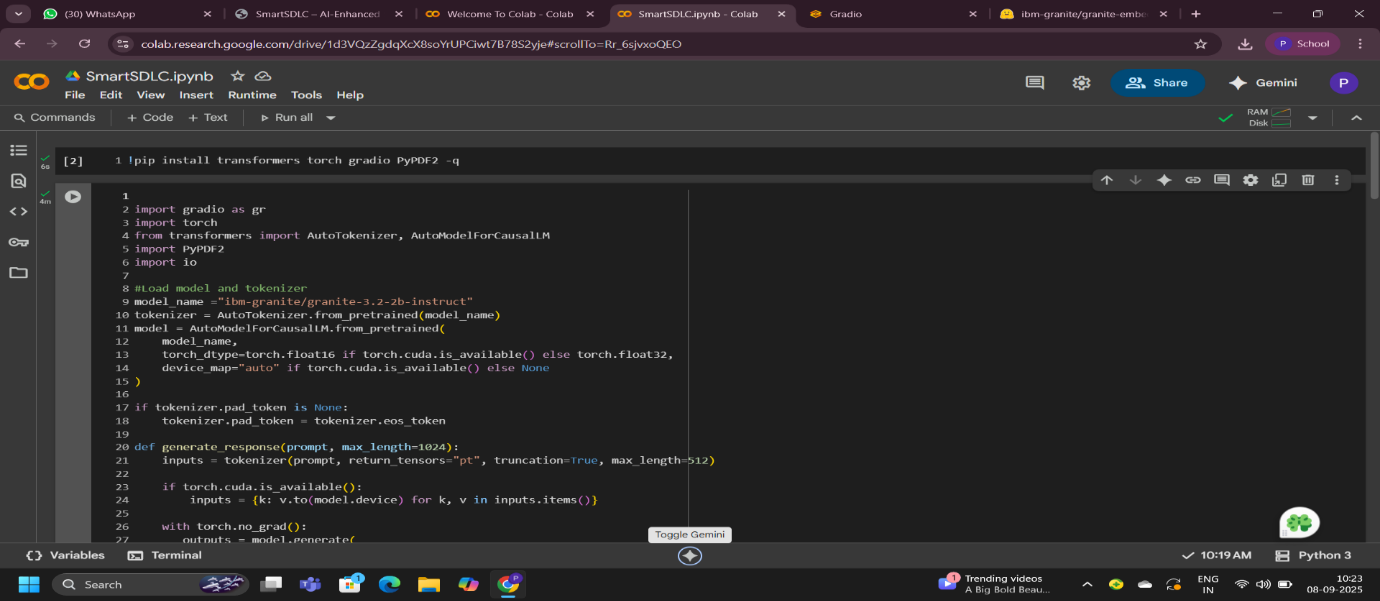
***Tab 2 – Policy Summarization***

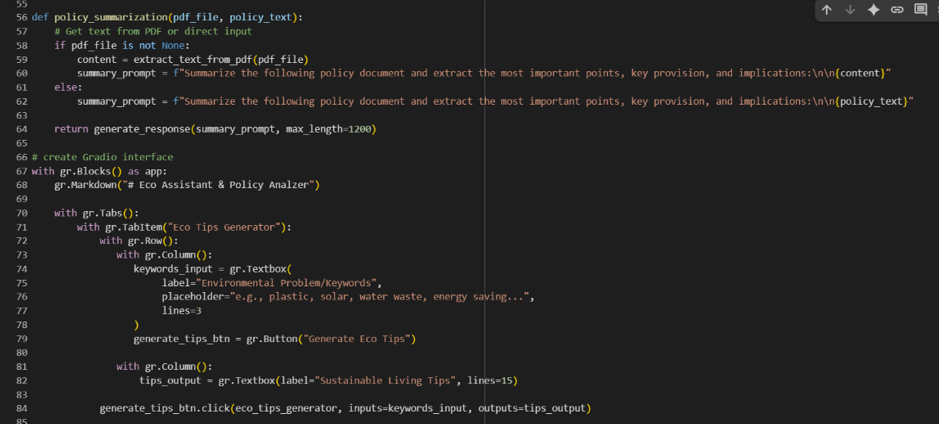
* ***Input Option 1: Upload Policy PDF.***
* ***Input Option 2: Paste policy text.***
* ***Output: Summarized policy with key provisions and implications.***

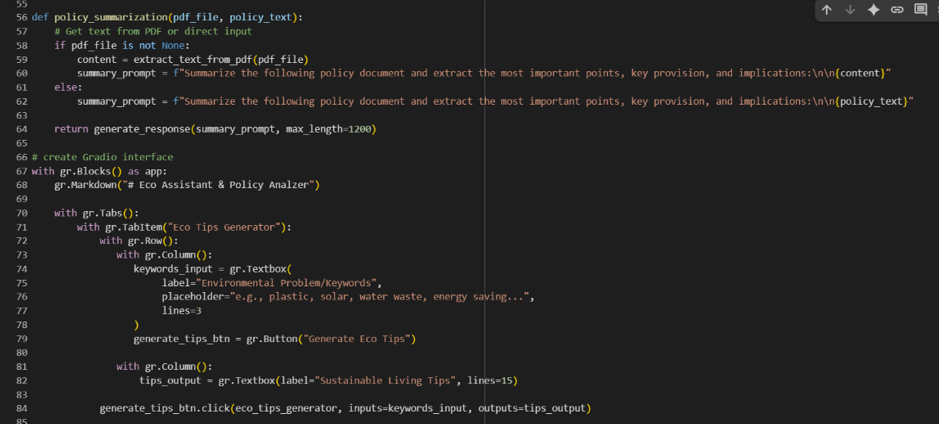
**9. Testing**

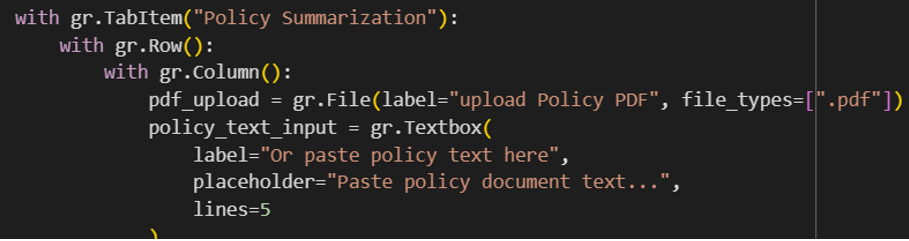
* ***Run the app locally or on Colab.***
* ***Test with valid and invalid inputs:***
  + ***Empty city name or query should return validation messages***.
  + ***Vary temperature and max tokens to observe output diversity.***
* ***Verify outputs are relevant and coherent.***

***10. Screenshots***

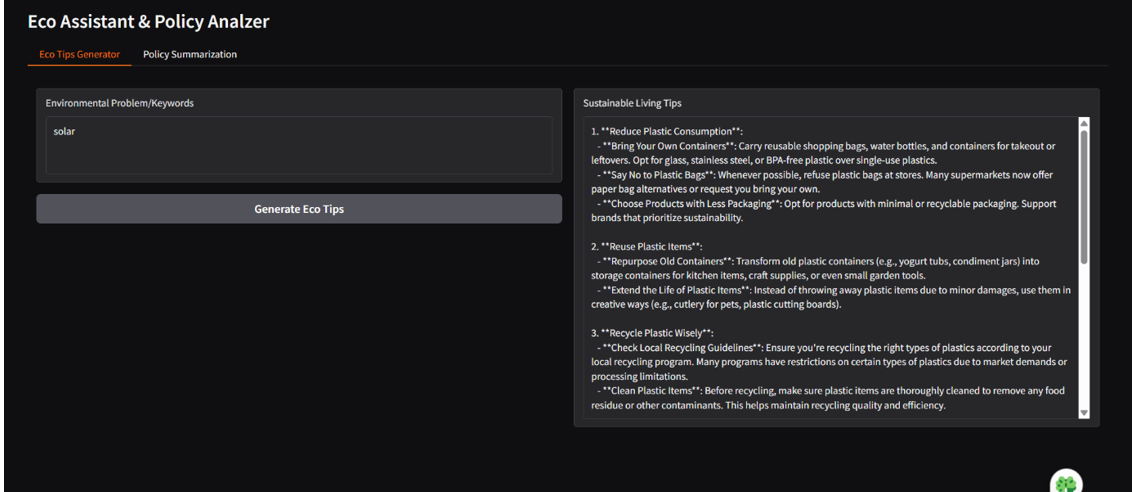


******

******

******

***Output:***

******

***11.Known Issues:***

* ***Lower accuracy for rare/niche environmental topics due to dataset limits.***
* ***No offline mode; dependent on stable internet and Hugging Face API.***
* ***No direct database or EHR/EMR integration for storing or fetching policies.***
* ***PDF text extraction may fail with scanned or image-based PDFs.***
* ***Limited UI (basic Gradio interface, no advanced visualization).***

***12. Future Enhancements***

* ***Add offline model support to reduce API dependency.***
* ***Improve PDF parsing with OCR for image-based files.***
* ***Integrate voice input for accessibility.***
* ***Build a dashboard for saving, organizing, and comparing summaries.***
* ***Expand datasets for rare environmental and policy topics.***
* ***Add cloud storage & security (encrypted summaries & tips).***
* ***Enable integration with government APIs for live policy updates.***