

RED DYN0 AI: AI-Enhanced Sustainable Development

Team Leader: VISHAL RAJ .A.V.K

Team Members: YOGESH.M.R

Team Members: SUBRAMANI.S

1. Introduction

RED DYN0 AI is an eco-focused AI-powered project that leverages IBM Granite models to enhance the Software Development Lifecycle (SDLC). It automates requirement gathering, code generation, testing, deployment, and documentation, while also providing sustainability-focused solutions such as carbon footprint estimation, policy summarization, eco-friendly lifestyle tips, and green technology ideas.

2. Project Overview

Purpose: The purpose of GreenSpark AI is to integrate AI-driven automation with eco-sustainability. It supports developers, researchers, and citizens in reducing environmental impact while demonstrating the power of SmartSDLC. Features: - EcoLife Guide: Generate eco-friendly lifestyle tips. - Policy Snapshot: Summarize environmental policy documents. - Carbon Meter: Estimate and suggest reduction for carbon footprint. - EcoTech Sparks: Recommend sustainable technology ideas.

3. Architecture

Frontend: Gradio-based interactive UI Backend: Python with Hugging Face Transformers and Torch Model: IBM Granite 3.3-2B Instruct (Hugging Face) Deployment: Google Colab with GPU (T4) Version Control: GitHub for hosting and collaboration

4. Setup Instructions

Prerequisites: - Python 3.9 or later - Install libraries: transformers, torch, gradio, pypdf - IBM Granite model access from Hugging Face - GPU-enabled runtime (Google Colab recommended) Steps: 1. Clone the repository 2. Install dependencies 3. Run the Python script 4. Launch the Gradio app 5. Interact with EcoLife, Policy Snapshot, Carbon Meter, and EcoTech modules

5. Folder Structure

app/ – Core functions and logic ui/ – Gradio UI definitions models/ – Model integration with Hugging Face docs/ – Documentation and reports main.py – Entry point to run the project

6. Running the Application

➤ Run the Python script ➤ Launch the Gradio interface in browser ➤ Navigate through tabs: EcoLife, Policy Snapshot, Carbon Meter, EcoTech ➤ Upload PDFs, enter activities, and generate responses

7. Module Documentation

- Eco Tips Generator: Provides actionable eco-friendly suggestions. - Policy Summarization: Extracts and summarizes environmental policies from PDFs or text. - Carbon Footprint Estimator: Estimates monthly CO₂ emissions and reduction strategies. - Green Tech Ideas: Suggests innovative sustainable technologies.

8. Authentication

The current version does not include authentication. Future versions may implement role-based access and API key security for extended deployments.

9. User Interface

TheUllis builtusing Gradio with multiple tabs: - EcoLife Guide: Textbox input and AI-generated tips - Policy Snapshot: PDF upload/text input for summaries - Carbon Meter: Lifestyle activity input with footprint estimates - EcoTech Sparks: Sector/industry input for innovation ideas Additional features: Copy-to-clipboard buttons for outputs

10. Testing

Testing includes: - Unit testing (functions and modules) - Manual testing of UI workflows - PDF text extraction validation - Edge case handling (invalid inputs, empty files)

11. Known Issues

- Requires internet access to fetch model - Limited offline support - Performance depends on GPU availability
- UI basic styling (can be improved)


12. Future Enhancements

- Mobileapp version - Multi-languagesupport - Voice-based eco-tips - IoT integration for real-time monitoring
- Enhanced dashboards for visualization

13. Screenshots & Outputs

[Insert screenshots of Gradio interface, sample outputs, and code execution here]

Program



The screenshot shows the SDLC.ipynb interface. The top bar includes the SDLC logo, the filename 'SDLC.ipynb', a status message 'Saving failed since 11:55 AM', and a menu bar with 'File', 'Edit', 'View', 'Insert', 'Runtime', 'Tools', and 'Help'. Below the menu bar is a toolbar with icons for 'Commands', 'Code', 'Text', 'Run all', and a dropdown arrow. The main area displays a code cell with the following Python code:

```
!pip install transformers torch gradio pyPDF

import gradio as gr
import torch
from transformers import AutoTokenizer, AutoModelForCausalLM
from pyPDF import PdfReader

# ----- Load Model -----
model_name = "ibm-granite/granite-3.3-2b-instruct"

tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModelForCausalLM.from_pretrained(
    model_name,
    torch_dtype=torch.float16 if torch.cuda.is_available() else torch.float32,
    device_map="auto" if torch.cuda.is_available() else None
)

if tokenizer.pad_token is None:
    tokenizer.pad_token = tokenizer.eos_token

# ----- Core Functions -----
def generate_response(prompt, max_length=1024):
```

```

# ----- Gradio UI -----
with gr.Blocks() as app:
    gr.Markdown("# RED DYNO AI")

    with gr.Tabs():
        # Tab 1: Eco Tips
        with gr.TabItem("Green Travel Choices"):
            with gr.Row():
                with gr.Column():
                    keywords_input = gr.Textbox(
                        label="Environmental Problem/Keywords",
                        placeholder="e.g., plastic, solar, water waste, energy saving...",
                        lines=3
                    )
                    generate_tips_btn = gr.Button("Generate Eco Tips")
                with gr.Column():
                    tips_output = gr.Textbox(label="Sustainable Living Tips", lines=10, elem_id="tips_box")
                    tips_copy = gr.HTML("""
                    <button onclick="navigator.clipboard.writeText(
                        document.querySelector('#tips_box textarea').value)"> Copy</button>
                    """)
                    generate_tips_btn.click(
                        eco_tips_generator,
                        inputs=keywords_input,
                        outputs=tips_output
                    )

```

Output

Inputs=sector_input,
outputs=ideas_output
)
Launch the app
app.launch(share=True)

Loading checkpoint shards: 100% 2/2 [00:18<00:00, 7.74s/it]
Colab notebook detected. To show errors in colab notebook, set debug=True in launch()
* Running on public URL: <https://7579f3cc999a26dc11.gradio.live>
This share link expires in 1 week. For free permanent hosting and GPU upgrades, run 'gradio deploy' from the terminal in the working directory to d

RED DYNO AI

Green Travel Choices Biodiversity Protection Acts Eco-Friendly Habits Eco-Friendly Materials

Environmental Problem/Keywords
e.g., plastic, solar, water waste, energy saving...

Sustainable Living Tips

ic saving has been pending for 40 minutes. Reloading may fix

GitHub

yogesh1553D / smart-sdlc

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

smart-sdlc (Public)

main 1 Branch 0 Tags

Go to file Add file Code

yogesh1553D Add files via upload f9d4fe · 22 minutes ago 2 Commits

README.md	Initial commit	23 minutes ago
smartsdlc.py	Add files via upload	22 minutes ago

README

smart-sdlc

ai

About

ai

Readme

Activity

0 stars

0 watching

0 forks

Releases

No releases published

Create a new release

Packages

No packages published