# Eco Assistant and Policy Analyzer-Source code

```
!pip install transformers torch gradio PyPDF2 -q

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


# Load model and tokenizer
model_name = "ibm-granite/granite-3.2-2b-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
```

```
def generate_response(prompt, max_length=1024):
    inputs = tokenizer(prompt, return_tensors="pt", truncation=True,
max_length=512)
    if torch.cuda.is_available():
        inputs = {k: v.to(model.device) for k, v in inputs.items()}
    with torch.no_grad():
        outputs = model.generate(
            **inputs,
            max_length=max_length,
            temperature=0.7,
            do_sample=True,
            pad_token_id=tokenizer.eos_token_id
        )
    response = tokenizer.decode(outputs[0], skip_special_tokens=True)
    response = response.replace(prompt, "").strip()
    return response


def extract_text_from_pdf(pdf_file):
    if pdf_file is None:
        return ""
    try:
        pdf_reader = PyPDF2.PdfReader(pdf_file)
        text = ""
        for page in pdf_reader.pages:
            text += page.extract_text() + "\n"
        return text
    except Exception as e:
        return f"Error reading PDF: {str(e)}"


def eco_tips_generator(problem_keywords):
    prompt = f"Generate practical and actionable eco-friendly tips for sustainable living related to: {problem_keywords}. Provide specific solutions and suggestions:"
    return generate_response(prompt, max_length=1000)


def policy_summarization(pdf_file, policy_text):
    if pdf_file is not None:
        content = extract_text_from_pdf(pdf_file)
        summary_prompt = f"Summarize the following policy document and extract the most important points, key provisions, and implications:\n\n{content}"
    else:
        summary_prompt = f"Summarize the following policy document and extract the most important points, key provisions, and implications:\n\n{policy_text}"
    return generate_response(summary_prompt, max_length=1200)
```

```
# ------------------- APP UI -------------------
with gr.Blocks(css="""
    .topbar {display:flex; justify-content:space-between; align-
items:center; padding:10px; background:#f0f0f0;}
    .auth-buttons {display:flex; gap:10px;}
    .modal {
        position: fixed !important;
        top:0; left:0;
        width:100%; height:100%;
        background: rgba(0,0,0,0.6);
        display:flex; justify-content:center; align-items:center;
        z-index:9999;
    }
    .modal-box {
        background:white; padding:20px;
        border-radius:10px; width:350px;
        box-shadow:0px 4px 20px rgba(0,0,0,0.3);
        position: relative;
        z-index:10000;
    }
    .overlay-btn {
        position: absolute !important;
        top:0; left:0;
        width:100%; height:100%;
        background: transparent;
        border: none !important;
        cursor: default;
    }
""") as app:

    # -------- Header with Nav + Auth --------
    with gr.Row(elem_classes="topbar"):
        with gr.Column(scale=3):
            gr.Markdown("## 🌍 Eco Assistant & Policy Analyzer")
        with gr.Column(scale=1, elem_classes="auth-buttons"):
            login_btn = gr.Button("Login")
            signup_btn = gr.Button("Sign Up")

    # -------- Login Modal --------
    with gr.Column(visible=False, elem_classes="modal") as login_modal:
        close_bg_login = gr.Button("", elem_classes="overlay-btn")
        with gr.Column(elem_classes="modal-box"):
            gr.Markdown("### 🔐 Login")
            login_user = gr.Textbox(placeholder="Username")
            login_pass = gr.Textbox(placeholder="Password",
type="password")
            login_submit = gr.Button("Login")
            login_status = gr.Label()
            close_login = gr.Button("Close")

    close_bg_login.click(lambda: gr.update(visible=False), None,
login_modal)
    close_login.click(lambda: gr.update(visible=False), None,
login_modal)

    # -------- Signup Modal --------
    with gr.Column(visible=False, elem_classes="modal") as
signup_modal:
        close_bg_signup = gr.Button("", elem_classes="overlay-btn")
        with gr.Column(elem_classes="modal-box"):
            gr.Markdown("### 📝 Sign Up")
            signup_user = gr.Textbox(placeholder="New Username")
            signup_pass = gr.Textbox(placeholder="New Password",
type="password")
            signup_submit = gr.Button("Sign Up")
            signup_status = gr.Label()
            close_signup = gr.Button("Close")

    close_bg_signup.click(lambda: gr.update(visible=False), None,
signup_modal)
    close_signup.click(lambda: gr.update(visible=False), None,
signup_modal)
```

```
    # Show modals
    login_btn.click(lambda: gr.update(visible=True), None, login_modal)
    signup_btn.click(lambda: gr.update(visible=True), None,
signup_modal)

    # -------- Tabs --------
    with gr.Tabs():
        with gr.TabItem("Eco Tips Generator"):
            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=15)
            generate_tips_btn.click(eco_tips_generator,
inputs=keywords_input, outputs=tips_output)

        with gr.TabItem("Policy Summarization"):
            with gr.Row():
                with gr.Column():
                    pdf_upload = gr.File(label="Upload Policy PDF",
file_types=[".pdf"])
                    policy_text_input = gr.Textbox(
                        label="Or paste policy text here",
                        placeholder="Paste policy document text...",
                        lines=5
                    )
                    summarize_btn = gr.Button("Summarize Policy")
                with gr.Column():
                    summary_output = gr.Textbox(label="Policy Summary & Key
Points", lines=20)
            summarize_btn.click(policy_summarization, inputs=[pdf_upload,
policy_text_input], outputs=summary_output)

        with gr.TabItem("About Us"):
            gr.Markdown("This app promotes eco-friendly living and helps
summarize policy documents.")

        with gr.TabItem("Contact Us"):
            gr.Markdown("📧 Email: ecoassistant@example.com\n🌐
Website: www.ecoassistant.org")

    # -------- Footer --------
    gr.Markdown(
        """
        ---
        <div style="text-align: center; color: gray;">
        © 2025 Eco Assistant | Built with ❤ using Gradio & Transformers
        </div>
        """
    )

app.launch(share=True)
```