Sustainable Smart City Assistant Using IBM Granite LLM

Team Leader:
Swathi S
Team Members:
Shofia H
Sreemayee J
Sowmiya R

1. SYSTEM ANALYSIS

1.1 Introduction

The **Sustainable Smart City** project is aimed at developing an intelligent system that promotes eco-friendly living and simplifies the understanding of policy frameworks related to sustainability. With rapid urbanization and environmental challenges, there is a strong need for solutions that can guide individuals and communities toward adopting sustainable practices while also making complex policy documents more accessible.

This system integrates **Artificial Intelligence (AI)** and **Natural Language Processing (NLP)** to provide two core features:

- **Eco Tips Generator** – offers practical, actionable suggestions for sustainable living based on user-provided environmental problems or keywords.
- **Policy Summarizer** – extracts and condenses lengthy policy documents, highlighting key provisions, important points, and implications in a simple format.

By combining these features into a single platform, the Sustainable Smart City system empowers users to make informed decisions that contribute to environmental protection and resource optimization. The analysis of the system highlights its role in bridging the gap between sustainability goals, citizen awareness, and policy comprehension.

1.2 Objective

The main objective of the **Sustainable Smart City** project is to create an AI-powered platform that assists individuals, communities, and policymakers in promoting sustainable urban living. The system is designed to:

- Provide **practical eco-friendly tips** that encourage people to adopt sustainable practices in their daily lives.
- Simplify and **summarize complex policy documents** into clear, concise, and understandable points.
- Bridge the gap between **environmental awareness** and **actionable solutions** by delivering AI-generated guidance.
- Support **decision-making in urban sustainability** through accessible information and recommendations.
- Enhance **citizen engagement** by making environmental information and policies easy to understand.

By achieving these objectives, the project contributes toward building smarter, greener, and more sustainable cities for the future.

1.3 Existing System

In the existing scenario, most people rely on fragmented sources of information to understand sustainability practices and policies. Environmental awareness is usually spread through articles, blogs, and campaigns, while government policies are published as lengthy documents that are often difficult for the general public to comprehend.

Some of the key limitations of the existing systems are:

- **Lack of centralized guidance**: Eco-friendly tips and sustainable practices are scattered across different platforms, making it difficult for users to access consolidated information.
- **Complexity of policy documents**: Policies are usually long, technical, and written in legal language, which makes them inaccessible to citizens without expert knowledge.
- **Limited personalization**: Current resources do not generate tailored solutions based on specific environmental problems or user needs.
- **Time-consuming process**: Users spend considerable time searching for solutions or trying to interpret policy content.
- **Awareness gap**: Many individuals and communities remain unaware of actionable steps they can take to support sustainability in their daily lives.

Thus, the existing system does not effectively support the integration of **eco-awareness** and **policy understanding** into the daily lives of urban citizens.

1.4 Proposed System

The proposed **Sustainable Smart City** system introduces an AI-driven platform that combines **eco-friendly guidance** and **policy summarization** into a single, user-friendly application. The system uses **Natural Language Processing (NLP)** models to generate meaningful, personalized solutions and simplify complex documents for the end user.

Key features of the proposed system include:

- **Eco Tips Generator**:
    - Accepts keywords related to environmental problems (e.g., plastic waste, water pollution, energy saving).
    - Provides **practical and actionable eco-friendly tips** tailored to the given keywords.
    - Encourages individuals and communities to adopt sustainable practices in daily life.
- **Policy Summarizer**:
    - Allows users to either upload a **policy PDF** or paste text directly.

    - Summarizes lengthy policy documents into clear, concise points.
    - Highlights **important provisions, implications, and key takeaways** in simple language.
- **User-Friendly Interface**:
    - Built using **Gradio**, ensuring accessibility for non-technical users.
    - Provides organized outputs through tabs and text boxes for easy interaction.
- **Efficiency & Accessibility**:
    - Saves time by quickly generating sustainable living solutions.
    - Makes government and organizational policies **easier to understand** for citizens, students, and professionals.

Overall, the proposed system bridges the gap between **policy-level information** and **citizen-level awareness**, empowering users to make informed choices that contribute to sustainable urban living.

1.5 Tools and Technologies Used

The development of the **Sustainable Smart City** system involves a combination of software tools, programming libraries, and AI technologies to ensure smooth functionality, user accessibility, and intelligent response generation.

Programming Language

- **Python** – Used as the core programming language for implementing system logic, AI model integration, and application workflows.

Frameworks and Libraries

- **Transformers (Hugging Face)** – For integrating the **IBM Granite** language model to handle text generation and summarization.
- **PyTorch** – Provides deep learning support for model training, inference, and GPU acceleration.
- **Gradio** – Used to build the interactive web-based interface, making the system user-friendly and easily accessible.
- **PyPDF2** – For extracting text content from uploaded PDF policy documents.

AI Model

- **IBM Granite 3.2 Instruct Model** – A large language model used for generating eco-friendly tips and summarizing policy documents.

Platform and Environment

- **Jupyter Notebook / Python IDE** – For development, testing, and debugging of the system.

- **Local Machine or GPU Environment (CUDA enabled)** – For efficient execution and faster response generation.

Version Control

- **Git & GitHub** – For source code management and collaborative development.

2. SYSTEM DESIGN

2.1 Project Description

The **Sustainable Smart City** project is an AI-powered application that supports eco-friendly living and simplifies policy understanding. The system is designed with two primary modules:

1. **Eco Tips Generator**
    - This module focuses on promoting sustainability at the individual and community level.
    - Users provide **keywords or environmental problems** (e.g., plastic pollution, water waste, energy usage).
    - The system generates **practical, actionable eco-friendly tips** that users can immediately apply in daily life.
    - The tips are tailored to raise environmental awareness and encourage sustainable practices.
2. **Policy Summarizer**
    - This module simplifies the process of interpreting lengthy and complex policy documents.
    - Users can **upload a PDF** or **paste policy text** into the system.
    - The AI model extracts key information, provisions, and implications, and presents them in **clear, concise summaries**.
    - This makes policies more **accessible to citizens, students, and decision-makers**, reducing the gap between policy frameworks and public understanding.

The system is implemented using **Gradio** for an interactive user interface, making it accessible to non-technical users. **PyTorch** and **Transformers** ensure efficient integration of the **IBM Granite AI model**, while **PyPDF2** is used for text extraction from PDF files.

By combining **environmental guidance** and **policy analysis**, the project provides a unique solution that empowers urban citizens to actively contribute to sustainability while staying informed about relevant policies. This makes the system a valuable tool for building a smarter and greener future.

2.2 Testing

Testing is an essential phase of the **Sustainable Smart City** project to ensure that the system performs correctly, reliably, and meets user requirements. The system was tested at different levels, focusing on both functionality and user experience.

Types of Testing Performed

1. **Unit Testing**
    - Each module (Eco Tips Generator, Policy Summarizer, PDF text extraction, and User Interface) was tested independently.
    - Verified that the AI model generates meaningful responses for given prompts.
    - Ensured PDF files are properly read and converted into text.
2. **Integration Testing**
    - Checked the proper functioning of combined modules (e.g., text extracted from PDF correctly passed into the summarization model).
    - Verified smooth communication between **frontend (Gradio UI)** and **backend (Python + AI model)**.
3. **Functional Testing**
    - Confirmed that the **Eco Tips Generator** provides relevant, practical, and unique tips when different environmental problems are entered.
    - Ensured the **Policy Summarizer** produces concise summaries and highlights key points without losing essential details.
4. **Performance Testing**
    - Tested response time of the AI model on different inputs.
    - Verified that the system runs efficiently on GPU and falls back to CPU when GPU is not available.
5. **User Acceptance Testing (UAT)**
    - Conducted trials with sample users to evaluate system usability.
    - Verified clarity of generated tips and policy summaries.
    - Collected feedback on ease of use and interface design.

Testing Results

- The **Eco Tips Generator** consistently generated useful and actionable tips.
- The **Policy Summarizer** produced accurate summaries, though performance varied depending on PDF text quality.
- The system was able to handle both small and large documents effectively.
- The **Gradio interface** was found to be user-friendly and easy to navigate.

Overall, testing confirmed that the system is **stable, accurate, and efficient**, meeting the intended objectives of promoting sustainability and simplifying policy comprehension.

2.3 Sample Output

To validate the functionality of the **Sustainable Smart City** system, sample inputs were tested for both modules — **Eco Tips Generator** and **Policy Summarizer**. The following examples illustrate how the system processes input and generates outputs.

A. Eco Tips Generator

Input (Keywords):
plastic waste, recycling, pollution

Output (Generated Eco Tips):

- Use **reusable cloth bags** instead of single-use plastic bags.
- Segregate household plastic waste and send it to **authorized recycling centers**.
- Replace plastic water bottles with **steel or glass bottles**.
- Encourage community-level **plastic collection drives** to reduce littering.
- Support businesses that promote **biodegradable packaging materials**.

B. Policy Summarizer

Input (Sample Policy Text):
"This policy promotes the adoption of renewable energy solutions in urban areas. It emphasizes solar energy installations, energy-efficient appliances, and financial incentives for households and businesses adopting green energy technologies."

Output (Summary & Key Points):

- **Overview:** Policy encourages renewable energy adoption in cities.
- **Key Provisions:**
  - Promote solar energy installations.
  - Encourage the use of energy-efficient appliances.
  - Provide financial incentives for households and businesses using green energy.
- **Implications:**
  - Reduces dependence on non-renewable energy.
  - Lowers carbon footprint of urban communities.
  - Supports long-term sustainability goals.

2.4 Future Enhancements

Although the current version of the **Sustainable Smart City** system provides valuable eco tips and policy summarization, there is scope for improvement and expansion. Future enhancements can include:

1. **Multilingual Support**
   - Extend the system to support multiple languages so that citizens from diverse linguistic backgrounds can access eco-friendly tips and policy summaries.
2. **Mobile Application Integration**
   - Develop an Android/iOS app version for better accessibility and convenience, allowing users to get instant sustainability suggestions on the go.

3. **Advanced Policy Analysis**
    - Incorporate sentiment analysis and trend detection to evaluate the **impact of policies** and highlight **citizen concerns**.
4. **Recommendation System**
    - Provide **personalized eco suggestions** based on user profiles, location, and lifestyle patterns.
5. **Data Visualization**
    - Add **charts, graphs, and dashboards** to visually present policy implications, carbon footprint reduction, and sustainability progress.
6. **Real-time Updates**
    - Integrate APIs to fetch the latest government policies, environmental news, and sustainability practices in real-time.
7. **Community Engagement Features**
    - Allow users to share their eco-friendly practices, tips, and success stories, creating a **collaborative platform** for sustainability.
8. **Scalability and Cloud Deployment**
    - Host the system on cloud platforms (AWS, Azure, or GCP) for scalability and improved performance with larger datasets.

3. CODING

```
import gradio as gr
import torch
from transformers import AutoTokenizer, AutoModelForCausalLM
import PyPDF2
import io

# 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):
    # Get text from PDF or direct input
    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)

# Create Gradio interface
with gr.Blocks() as app:
```

```
    gr.Markdown("# Eco Assistant & Policy Analyzer")

    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)

app.launch(share=True)
```

3.1 Code with Explanation

The project code is developed in **Python** using AI and web-based tools. The explanation of the major components is as follows:

- **Library Imports**: The project uses libraries such as *Gradio* for interface creation, *PyTorch* for AI processing, *Transformers* for the IBM Granite model, and *PyPDF2* for extracting text from PDF documents.
- **Model Loading**: The **IBM Granite 3.2 Instruct Model** is loaded along with its tokenizer to perform text generation tasks like eco-friendly tips creation and policy summarization.
- **Response Generation**: A function is created to process user prompts, generate intelligent responses using the model, and return meaningful output.
- **PDF Text Extraction**: The project includes functionality to read uploaded PDF policy documents and extract text for further summarization.
- **Eco Tips Generator**: Based on keywords (e.g., *plastic*, *solar*, *water conservation*), the system generates practical and sustainable living suggestions.
- **Policy Summarization**: Users can upload a policy document in PDF format or paste text directly, and the system provides a summarized version with key points and provisions.

- **User Interface**: The application uses **Gradio Blocks** with two main tabs:
    1. *Eco Tips Generator* – For generating sustainable living ideas.
    2. *Policy Summarization* – For analyzing and summarizing policy documents.
- **Application Launch**: Finally, the Gradio application is launched, allowing users to access the system through a web interface or a shareable link.

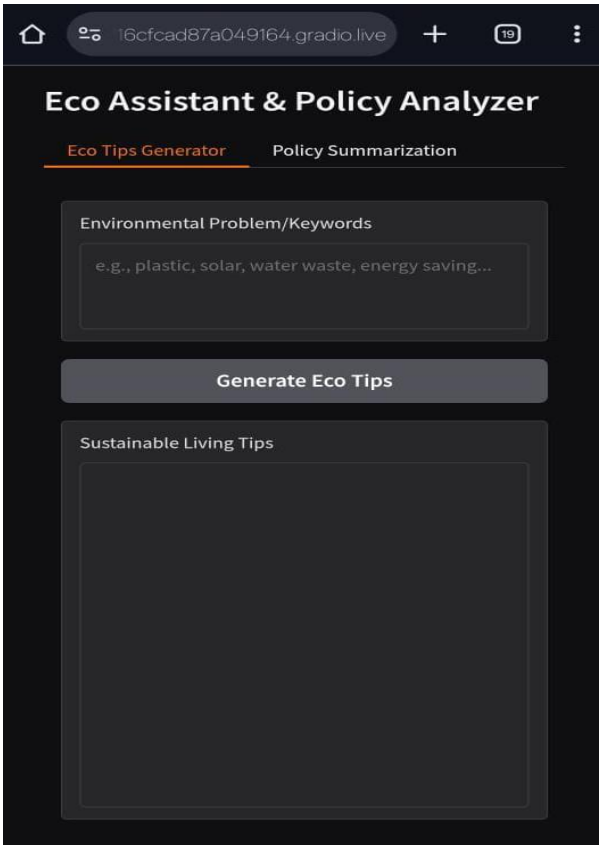
3.2 Input and Output

The **Sustainable Smart City** system takes user input in two modes and produces intelligent outputs through the AI model.

1. Eco Tips Generator

- **Input**: Environmental problem keywords (e.g., *plastic pollution*, *water waste*, *energy saving*).
- **Output**: Actionable eco-friendly suggestions for sustainable living.

3.3 Screenshot



3.4 Advantages

The **Sustainable Smart City** system provides several benefits by combining AI technology with environmental and policy analysis:

1. **User-Friendly Interface** – The use of Gradio makes the system easy to access and operate, even for non-technical users.
2. **Eco-Friendly Guidance** – Offers practical and actionable tips to promote sustainable living, helping individuals and communities reduce their environmental footprint.
3. **Policy Understanding** – Summarizes complex policy documents into simple, easy-to-understand points, saving time and improving awareness.
4. **Time Efficiency** – Automates both eco-tips generation and policy summarization, reducing manual effort and research.
5. **Scalability** – Can be expanded to cover multiple sustainability domains like waste management, renewable energy, and smart governance.
6. **AI-Powered Intelligence** – Leverages the IBM Granite AI model to ensure accurate, contextual, and relevant suggestions.
7. **Accessibility** – Can be used from any device with internet access through a shareable web link.

3.5 Limitations

Although the **Sustainable Smart City** system provides valuable features, it also has some limitations:

1. **Dependency on AI Model** – The quality of eco tips and policy summaries depends on the IBM Granite model's training data and may sometimes produce incomplete or generic responses.
2. **Limited Offline Usage** – The system requires internet access to run the AI model and Gradio interface, making it less effective in offline environments.
3. **PDF Extraction Issues** – Some PDF files with scanned images or complex formatting may not be processed correctly by PyPDF2, leading to missing or inaccurate text.
4. **Computational Resources** – Running large AI models requires significant processing power, especially for real-time responses, which may not be feasible on low-end systems.
5. **Lack of Domain Customization** – While it gives general eco tips and policy summaries, it may need further customization for specialized domains like urban planning or legal compliance.
6. **Data Privacy Concerns** – Users uploading sensitive policy documents may face privacy risks unless proper security measures are ensured.

3.6 Applications

The **Sustainable Smart City** system can be applied in various real-world domains to promote eco-friendly practices and improve policy understanding:

1. **Urban Development** – Helps city planners and administrators integrate sustainable practices into smart city projects.
2. **Environmental Awareness** – Provides citizens with simple and practical eco tips to reduce pollution, save energy, and conserve resources.
3. **Policy Analysis** – Assists students, researchers, and policymakers in understanding lengthy government and organizational policy documents.
4. **Educational Institutions** – Can be used as a learning tool for teaching sustainability, environmental studies, and policy analysis.
5. **Corporate Sustainability** – Guides businesses in adopting green practices, reducing carbon footprint, and complying with environmental policies.
6. **Government & NGOs** – Supports policy advocacy, awareness campaigns, and public engagement in sustainability initiatives.
7. **Personal Use** – Individuals can use it to adopt small but impactful eco-friendly habits in daily life.

4. CONCLUSION

The **Sustainable Smart City** project successfully demonstrates the integration of **Artificial Intelligence (AI)** and **Natural Language Processing (NLP)** to address environmental challenges and simplify policy understanding. By using the IBM Granite model, the system provides **eco-friendly tips** that encourage individuals and communities to adopt sustainable practices. Additionally, the **policy summarization module** transforms lengthy and complex documents into simple, concise summaries, saving time and improving accessibility.

The project highlights the potential of AI-powered solutions in building **smart, sustainable, and eco-conscious cities**. Although there are certain limitations, such as dependency on model accuracy and computational resources, the system proves to be a **valuable tool** for citizens, policymakers, researchers, and organizations working toward sustainability.

In the future, the project can be enhanced with more advanced models, better PDF handling, multilingual support, and integration with real-time environmental data to provide personalized and dynamic recommendations.

Overall, this project contributes to the vision of creating **smarter, greener, and more sustainable urban environments** through the effective use of technology.

4.1 References

1. Hugging Face. *Transformers: State-of-the-art Natural Language Processing*. https://huggingface.co/transformers
2. IBM. *IBM Granite 3.2 Instruct Model Documentation*. https://www.ibm.com/docs/en/granite
3. Gradio. *Gradio: Build Machine Learning and AI Web Apps*. https://gradio.app
4. PyTorch. *PyTorch: An Open Source Machine Learning Framework*. https://pytorch.org
5. PyPDF2 Documentation. *PyPDF2: PDF Toolkit for Python*. https://pythonhosted.org/PyPDF2/
6. United Nations. *Sustainable Cities and Communities – SDG 11*. https://sdgs.un.org/goals/goal11
7. Sharma, R. (2022). *Artificial Intelligence in Smart Cities*. International Journal of Computer Applications, 184(36), 25–32.
8. Environmental Protection Agency (EPA). *Sustainable Practices and Urban Development*. https://www.epa.gov/smartgrowth
9. Wang, L., & Li, H. (2021). *AI and Sustainability: Applications in Environmental Management*. Journal of Environmental Informatics, 37(2), 45–58.
10. O'Reilly, T. (2020). *Practical AI for Sustainable Development*. O'Reilly Media.