■ Project Documentation

SMART SDLC – AI Enhanced Software Development Lifecycle

Team Members

* Team Leader: Amarish Vicky
* Team Members: Kuppusamy, Mohana Krishnan, Ayyapan*

* College: Govt Arts College, Nandhanam

Abstract

This project introduces SMART SDLC – AI Enhanced Software Development Lifecycle, which integrates Artificial Intelligence into the traditional Software Development Life Cycle. The implementation focuses on building an Eco Assistant & Policy Analyzer using Natural Language Processing (NLP) models and Gradio for user interaction. The system provides eco-friendly lifestyle tips and also summarizes lengthy environmental policy documents for easy understanding.

Introduction

Traditional SDLC models provide structured approaches for software development but lack adaptability in emerging areas like sustainability. The AI Enhanced SMART SDLC integrates AI-driven automation and eco-awareness into the development process.

This project demonstrates how AI can:
* Generate eco-friendly lifestyle tips.
* Summarize and extract insights from policy documents.
* Provide an easy-to-use Gradio interface for interaction.

System Implementation

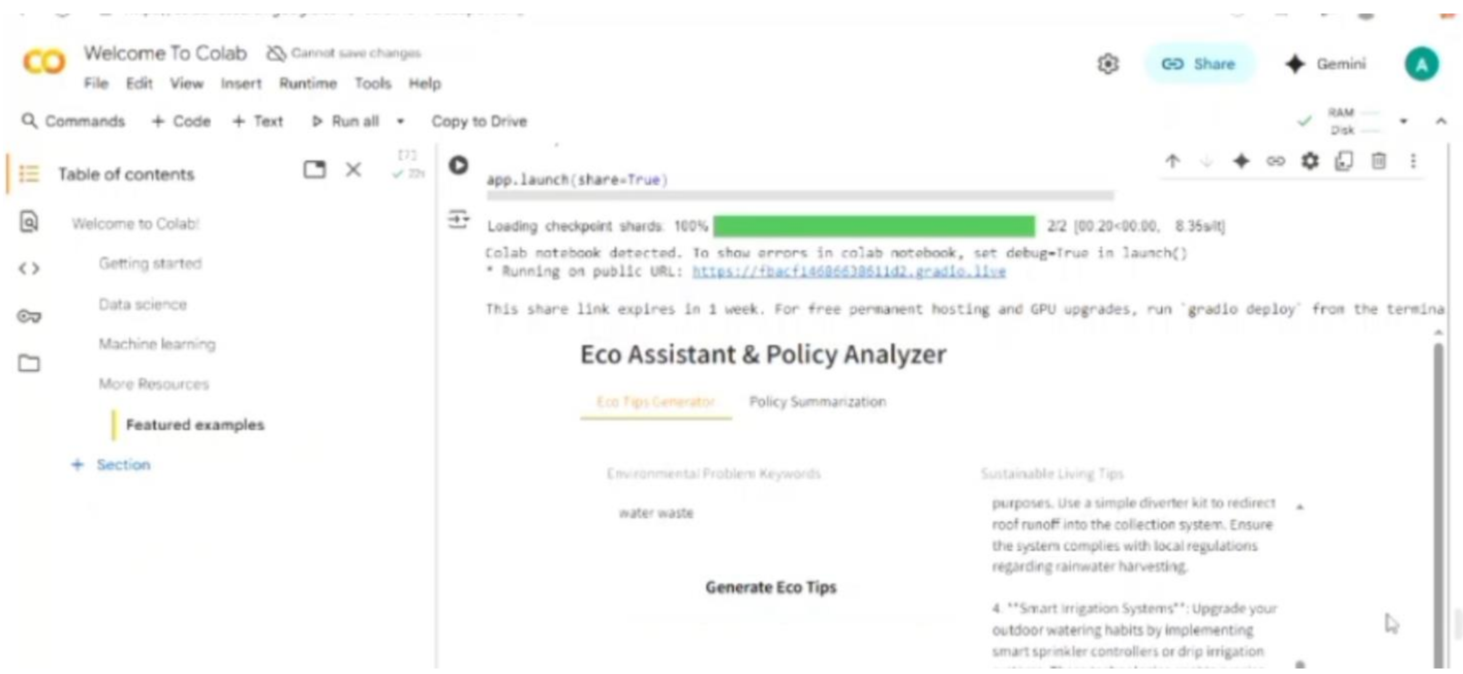
The system is implemented using:

* Python for backend logic.
* Transformers (Hugging Face) for NLP model.
* Gradio for creating a simple web interface.
* PyPDF2 for reading PDF policy documents.

Source Code

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

# Load model and tokenizer model_name = "JIN-granite-1.2b-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 ) tokenizer.pad_token
= 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() + " "
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_keyw
return generate_response(prompt, max_length=1000)

def policy_summarization(pdf_file, policy_text):
    content = ""     if
pdf_file is not None:
        content = extract_text_from_pdf(pdf_file)
elif policy_text:
        content = policy_text
if not content:
        return "No content to summarize."         summary_prompt = f"Summarize the following policy document and
extract the most important points, key provision     return generate_response(summary_prompt, max_length=1200)

# Create Gradio Interface
with gr.Blocks() as app:
    gr.Markdown("# Eco Assistant & Policy Analyzer")
with gr.Tab("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..."                 )
                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.Tab("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)
```

Output



```
app.launch(share=True)
```

```
Loading checkpoint shards: 100%                                  2/2 [00:20<00:00,  8.35s/it]
Colab notebook detected. To show errors in colab notebook, set debug=True in launch()
* Running on public URL: https://fbacf1468663861ld2.gradio.live

This share link expires in 1 week. For free permanent hosting and GPU upgrades, run `gradio deploy` from the termina
```

Eco Assistant & Policy Analyzer

water waste

purposes. Use a simple diverter kit to redirect roof runoff into the collection system. Ensure the system complies with local regulations regarding rainwater harvesting.

4. **Smart Irrigation Systems**: Upgrade your outdoor watering habits by implementing smart sprinkler controllers or drip irrigation

The system successfully generates eco-friendly suggestions and summarizes uploaded policy PDFs. The user interface, built with Gradio, is straightforward and functional.

Conclusion

This project demonstrates how AI can be integrated into the SDLC for building eco-aware applications. Using NLP and interactive tools like Gradio, software can be enhanced to not only meet functional requirements but also promote sustainability and awareness. The SMART SDLC framework serves as a model for future software development that incorporates emerging technologies and social responsibility into its core design.