Silvia.AI - Project Report

1. Project Overview

Silvia.AI is a Flask-based web application that serves as an intelligent AI assistant. It leverages Google's Gemini API to provide natural language responses and text summarization. The application features a distinctive "Glassmorphism" dark UI inspired by the Nissan Silvia S15, complete with a custom lightning bolt logo and neon accents.

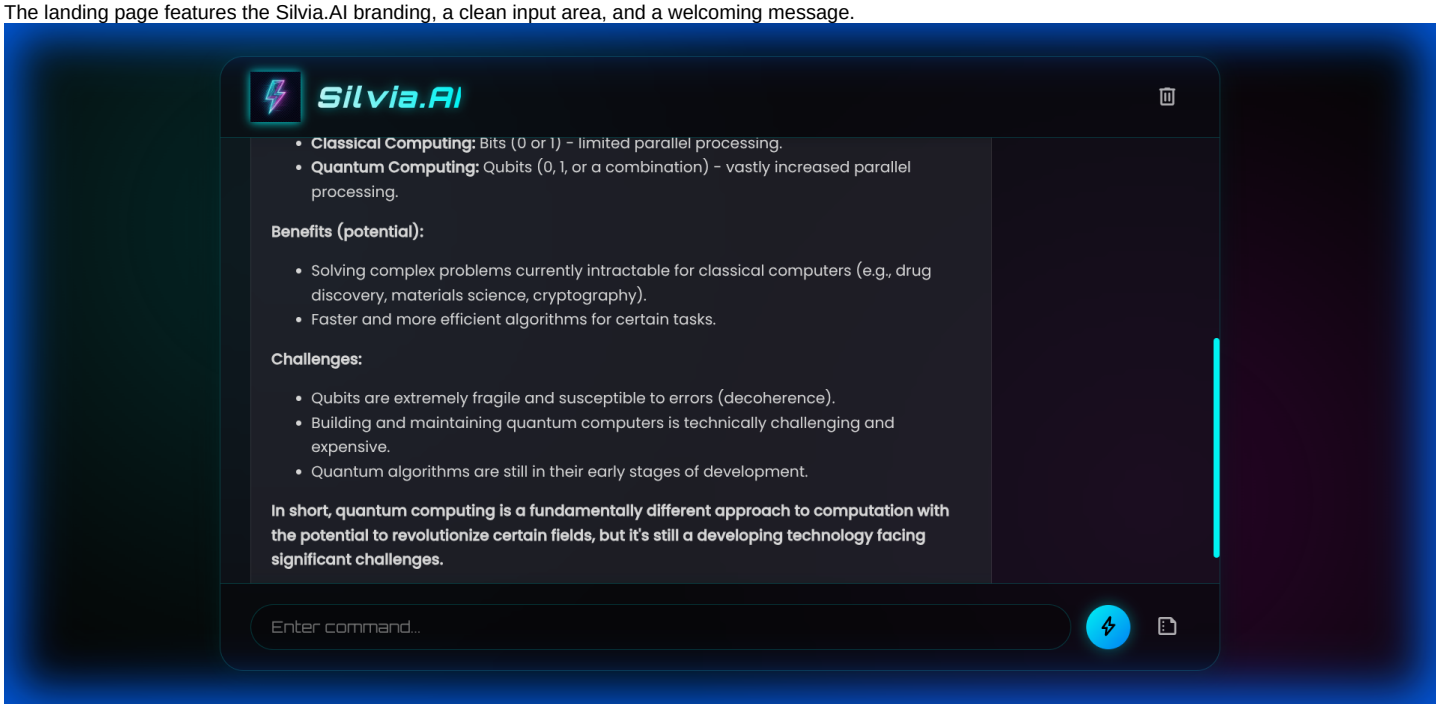
2. Key Features

- **AI Chat Interface**: Real-time conversation with Google Gemini.
- **Text Summarization**: Dedicated tool for summarizing long texts.
- **Persistent History**: Chat history is saved within the session.
- **Custom UI**: A responsive, dark-themed interface with glassmorphism effects.
- **Secure Configuration**: API keys are managed via environment variables.

3. Application Screenshots

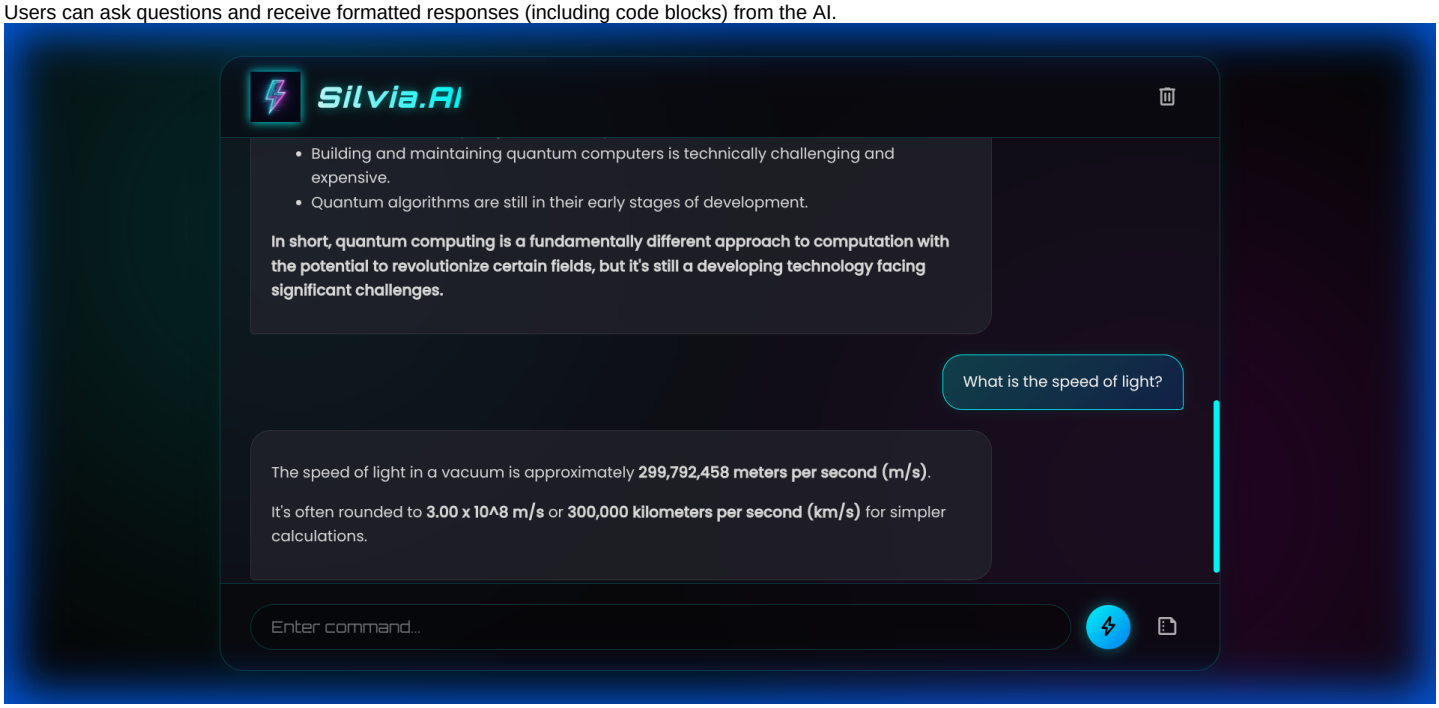
3.1 Home Page

The landing page features the Silvia.AI branding, a clean input area, and a welcoming message.



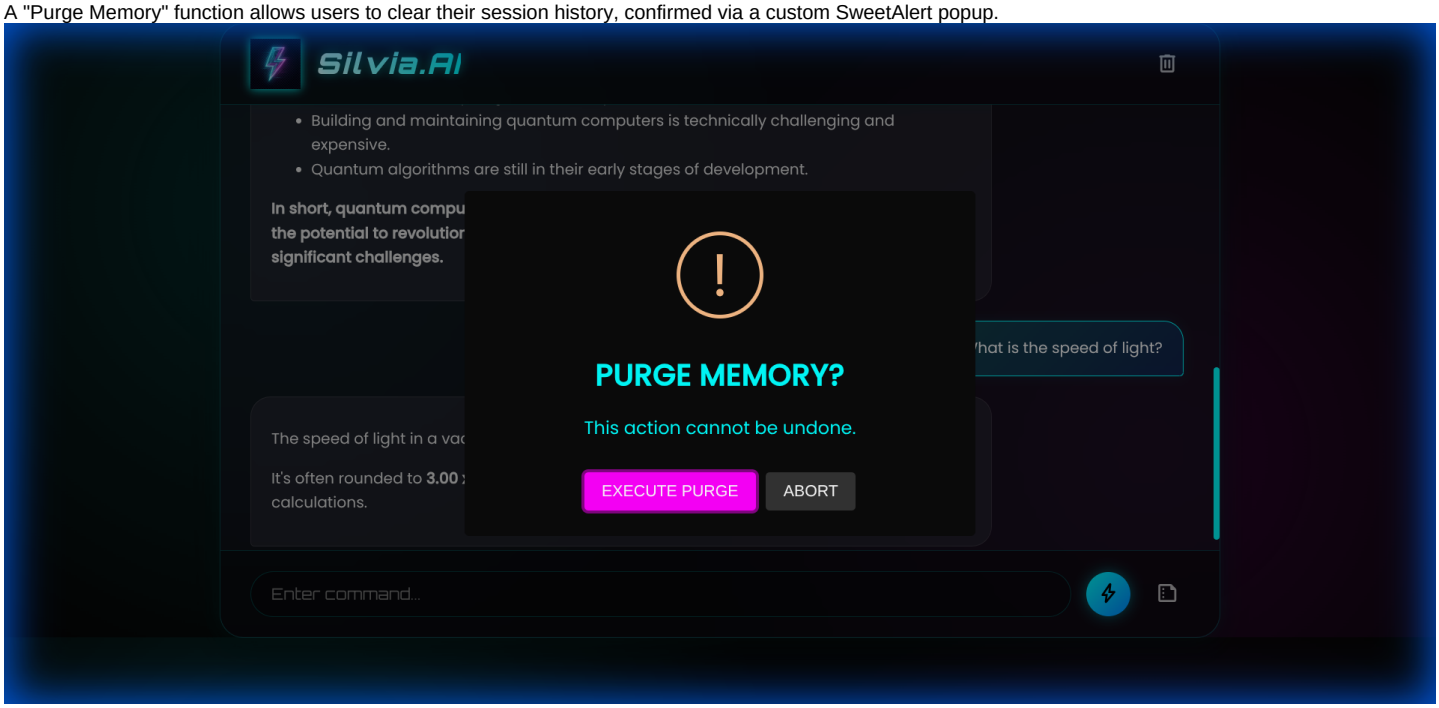
3.2 Chat Interaction

Users can ask questions and receive formatted responses (including code blocks) from the AI.



3.3 Clear History

A "Purge Memory" function allows users to clear their session history, confirmed via a custom SweetAlert popup.



4. Code Structure & Explanation

4.1 `ai_web_helper_pkg/core.py`

This module contains the core logic for interacting with the Google Gemini API.

```
import google.generativeai as genai
import markdown

def get_response(prompt, api_key):
   """
   Sends a prompt to the Google Gemini API and returns the text response.
   """
   genai.configure(api_key=api_key)
   model = genai.GenerativeModel('gemini-2.0-flash')
   response = model.generate_content(prompt)
   return response.text
```

```
def format_response(text):
   """
   Converts Markdown text to HTML for display in the web interface.
   """
   return markdown.markdown(text, extensions=['fenced_code', 'codehilite'])
```

4.2 `flask_app/app.py`

The Flask application handles routing and session management.

```
from flask import Flask, render_template, request, session, redirect, url_for
from ai_web_helper import get_response, summarize_text, format_response

app = Flask(__name__)
app.secret_key = os.getenv("SECRET_KEY", "dev_key")

@app.route("/", methods=["GET", "POST"])
def index():
   if "history" not in session:
       session["history"] = []

   if request.method == "POST":
       user_input = request.form.get("user_input")
       action = request.form.get("action")

       # ... logic to call get_response or summarize_text ...

       # Store in session history
       session["history"].append({"role": "user", "content": user_input})
       session["history"].append({"role": "ai", "content": formatted_response})

   return render_template("index.html", history=session["history"])
```

4.3 `flask_app/templates/index.html`

The HTML template implements the Glassmorphism UI using custom CSS variables and backdrop filters.

```
:root {
   --glass-bg: rgba(20, 20, 30, 0.6);
   --primary-color: #00ffff; /* Cyan */
}

.glass-container {
   background: var(--glass-bg);
   backdrop-filter: blur(20px);
   border: 1px solid var(--glass-border);
   box-shadow: 0 8px 32px 0 rgba(0, 0, 0, 0.5);
}
```

5. Conclusion

Silvia.AI demonstrates the integration of modern AI capabilities with a highly stylized, custom web interface. The separation of concerns between the library package and the Flask app ensures maintainability and scalability.