**Inspirational Quote Generator Streamlit App: Implementation & Execution**

**Author**: Saurabh Yadav  
**Reg. no.**: 22BCY10154

**Abstract**

This report documents the implementation and execution phase of the Inspirational Quote Generator, a Streamlit web application that generates motivational quotes using the GPT-2 model. The development process involved setting up the environment, coding core features, enhancing the user interface, addressing technical challenges, and deploying the app on Streamlit Community Cloud. Key features include support for comma-separated themes, user-specified quote lengths, dynamic theme-based images, and a polished UI. This report provides a comprehensive overview of the technical implementation, challenges, solutions, and testing outcomes.

**1. Introduction**

The implementation phase transformed the proposed Inspirational Quote Generator into a fully functional web application. The project leveraged the GPT-2 model for quote generation, Streamlit for the web interface, and Streamlit Community Cloud for deployment. Key tasks included coding the core functionality, adding advanced features, designing a user-friendly UI, and ensuring robust performance. This phase, spanning March to April 2025, involved iterative development, rigorous testing, and problem-solving to meet the project's objectives. This report details the technical setup, development process, challenges, solutions, and deployment steps.

**2. Technical Setup**

**2.1 Environment**

* **Operating System**: Windows 11
* **IDE**: Visual Studio Code with Python and Git extensions
* **Python Version**: 3.10
* **Virtual Environment**: Created with python -m venv venv to isolate dependencies
* **Version Control**: Git with a GitHub repository ( [https://github.com/SaurabhIndi/quote-generator#](https://github.com/SaurabhIndi/quote-generator) )

**2.2 Dependencies**

Installed via requirements.txt:

* streamlit==1.39.0: Web framework for the UI.
* transformers==4.44.2: Provides GPT-2 model and tokenizer.
* torch==2.3.1: PyTorch for model inference (CPU version).

Command for torch installation:

pip install torch==2.3.1 --index-url https://download.pytorch.org/whl/cpu

**2.3 Project Structure**

quote-generator/

├── app.py # Main Streamlit application

├── requirements.txt # Dependencies

├── README.md # Project documentation

├── .gitignore # Excludes venv, cache, and quotes.txt

├── quotes.txt # Generated quotes (created at runtime)

**3. Development Process**

**3.1 Core Functionality**

The initial development focused on integrating GPT-2 for quote generation:

* **Model Loading**: Used transformers to load the GPT-2 model and tokenizer, cached with @st.cache\_resource for performance.
* **Quote Generation**: Implemented generate\_quote(theme, max\_length=60) to produce quotes based on a prompt (An inspirational quote about {theme}:). Outputs were cleaned using regex to remove the prompt and ensure proper punctuation.
* **File Saving**: Added save\_quote(theme, quote, author) to append quotes to quotes.txt with the format Theme: {theme} | Quote: {quote} — AI Sage.

**3.2 Feature Enhancements**

Iterative improvements added advanced features:

* **Comma-Separated Themes**: Modified main() to split input themes (e.g., success, inner peace) and generate quotes for each, displayed under theme-specific headings.
* **Quote Lengths**: Added a dropdown for Short (40 tokens), Medium (60 tokens), and Long (80 tokens), passed to generate\_quote via a length\_map dictionary.
* **Dynamic Images**: Implemented get\_theme\_image(theme) to map themes to Unsplash URLs (e.g., mountains for success, river for inner peace). Partial matching supports multi-word themes (e.g., new hope matches hope).
* **UI Styling**: Applied custom CSS via st.markdown for a blue-white theme, quote cards with shadows, rounded inputs, and a responsive layout. Used containers and columns for structured layout.

**3.3 Code Structure**

Key functions in app.py:

* load\_model(): Loads GPT-2 model and tokenizer.
* get\_model(): Caches model loading.
* generate\_quote(theme, max\_length): Generates a quote with the "AI Sage" author.
* save\_quote(theme, quote, author): Saves quotes to quotes.txt.
* get\_theme\_image(theme): Returns theme-based image URL and caption.
* main(): Orchestrates the Streamlit app, handling inputs, outputs, and UI.

**3.4 Testing**

Testing was conducted iteratively:

* **Unit Testing**: Verified generate\_quote outputs for single and multi-word themes, checking quote coherence and punctuation.
* **Integration Testing**: Ran streamlit run app.py to test the full app at http://localhost:8501. Tested inputs like hope, inner peace, quote lengths, and image changes.
* **UI Testing**: Ensured responsiveness across browser sizes and verified CSS styling (e.g., quote cards, button hover effects).
* **Edge Cases**: Tested empty inputs, invalid themes, and high quote counts (5 per theme).

Example test case:

* **Input**: Themes = success,fire in the belly, Quotes = 2, Length = Short
* **Output**:
* Quotes for 'success'
* Quote 1: Success fuels bold dreams. — AI Sage
* Quote 2: Success is steady effort. — AI Sage
* Quotes for 'fire in the belly'
* Quote 1: Fire in the belly drives us. — AI Sage
* Quote 2: Passion ignites bold goals. — AI Sage

**4. Challenges and Solutions**

**4.1 Dependency Installation**

* **Challenge**: torch==2.3.1 failed to install due to platform incompatibility.
* **Solution**: Used PyTorch's CPU index:
* pip install torch==2.3.1 --index-url https://download.pytorch.org/whl/cpu
* **Impact**: Enabled smooth model inference on a standard PC.

**4.2 Multi-Word Themes**

* **Challenge**: GPT-2 produced incoherent quotes for multi-word themes (e.g., new hope).
* **Solution**: Improved the prompt to An inspirational quote about {theme}: and enhanced regex cleaning (re.sub(r"An inspirational quote about .+?:", "", quote)).
* **Impact**: Increased quote relevance and coherence.

**4.3 UI Limitations**

* **Challenge**: Streamlit's default UI was plain and lacked visual appeal.
* **Solution**: Added custom CSS for quote cards, a blue-white color scheme, and dynamic images. Used st.columns and st.container for layout.
* **Impact**: Created a polished, user-friendly interface.

**4.4 Image Loading**

* **Challenge**: Unsplash image URLs occasionally failed to load due to network issues.
* **Solution**: Selected reliable URLs and implemented a default image for unmatched themes.
* **Impact**: Ensured consistent visual feedback.

**5. Deployment**

The app was deployed on Streamlit Community Cloud:

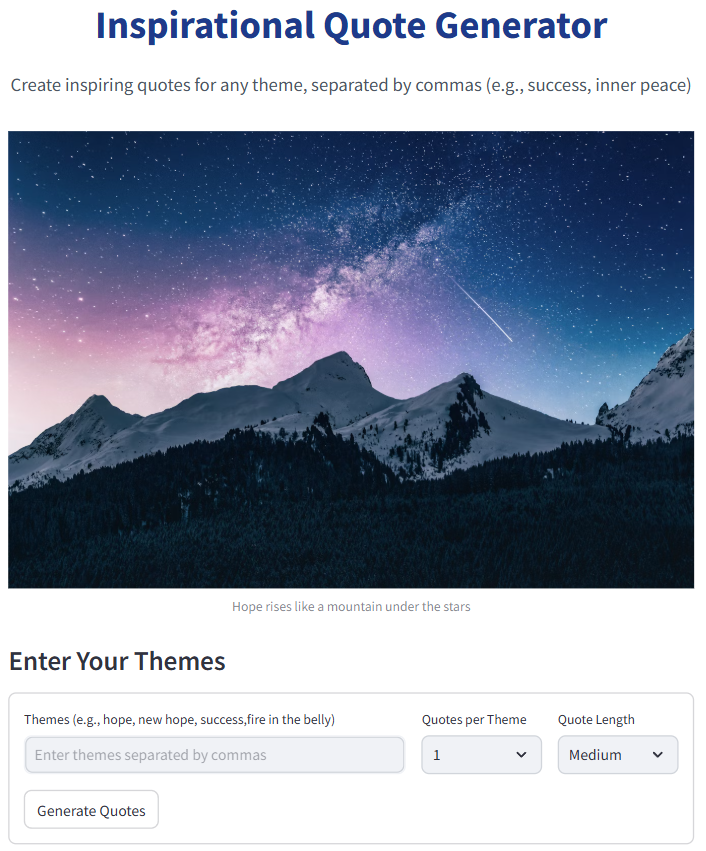
* **Repository**: Pushed to https://github.com/SaurabhIndi/quote-generator#
* **Setup**:
  + Selected main branch and app.py as the main file.
  + Used default Python version (compatible with 3.10).
  + Ensured requirements.txt included all dependencies.
* **Process**: Streamlit Cloud built the app in ~10 minutes, installing dependencies and running streamlit run app.py.
* **Result**: App accessible at <https://quote-generator-ec8vwst9gcraxqjfs6widb.streamlit.app/>
* **Testing**: Verified quote generation, image changes, and UI styling on the deployed app.

**6. User Experience**

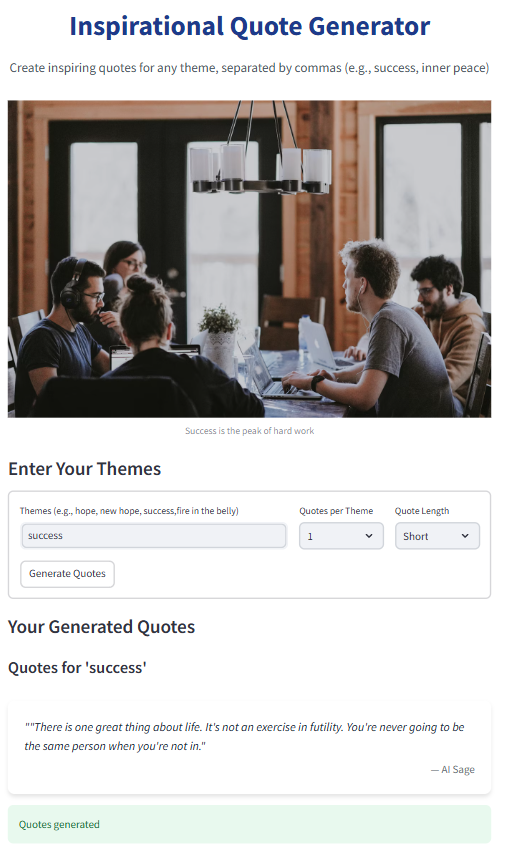
The app prioritizes usability:

* **Intuitive Form**: Users enter themes (e.g., hope, success), select quote numbers (1–5), and choose lengths (Short, Medium, Long).
* **Visual Feedback**: Dynamic images and styled quote cards enhance engagement.
* **Error Handling**: Validates empty inputs and invalid quote counts with clear error messages.
* **Output**: Quotes are displayed in cards with the "AI Sage" author and saved to quotes.txt.

Example screenshot description:



* A scenic mountain image for quotes inspiration, followed by a form with rounded inputs and a "Generate Quotes" button.



* A representative image for success, followed by a form with rounded inputs and a "Generate Quotes" button. Quote cards show italicized quotes and right-aligned authors.

**7. Conclusion**

The implementation phase successfully delivered a robust Quote Generator with advanced features, a polished UI, and cloud deployment. Challenges with dependencies, theme handling, and styling were addressed through targeted solutions, ensuring a high-quality user experience. The app is ready for final testing and submission, with potential for further enhancements like downloadable quotes.