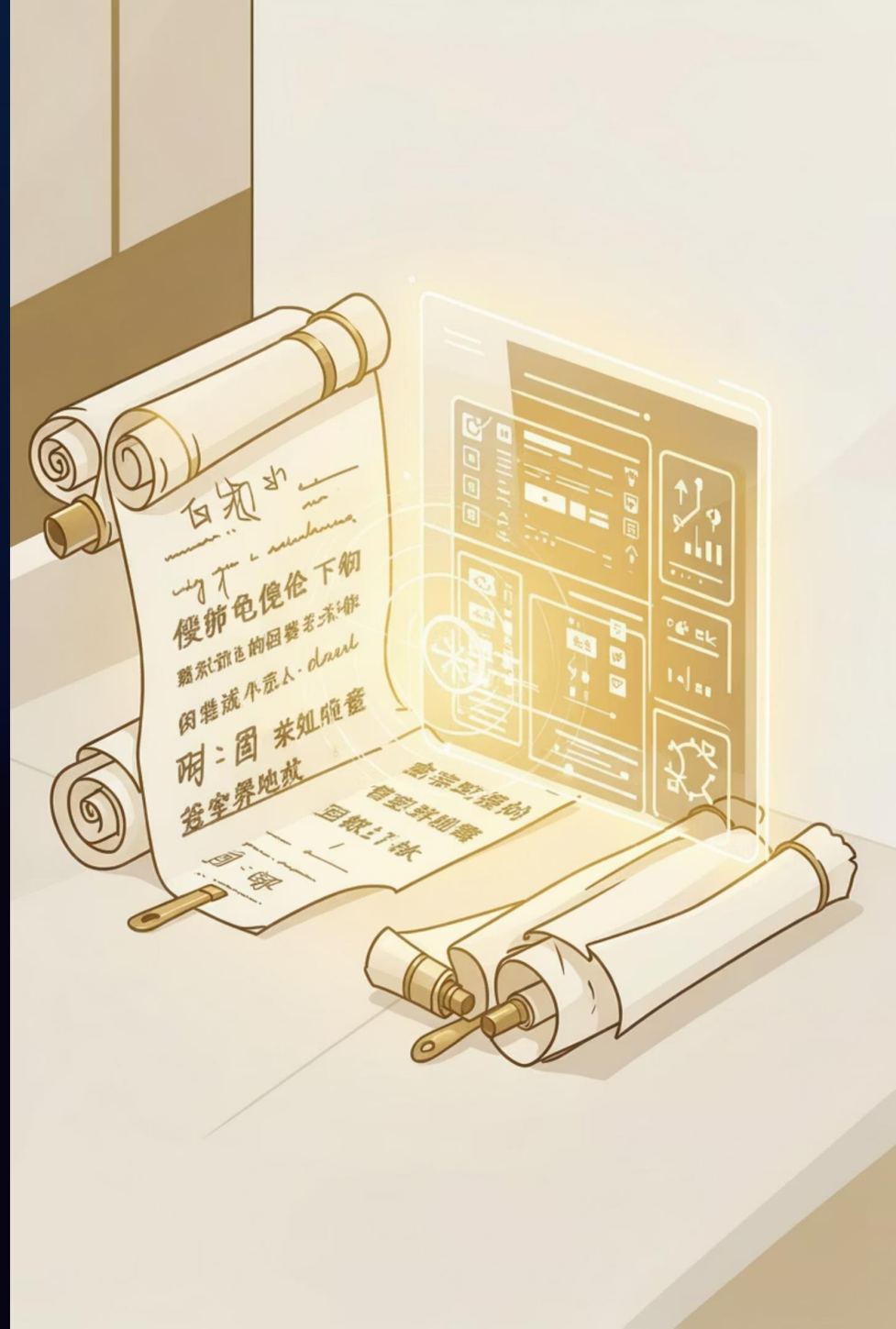
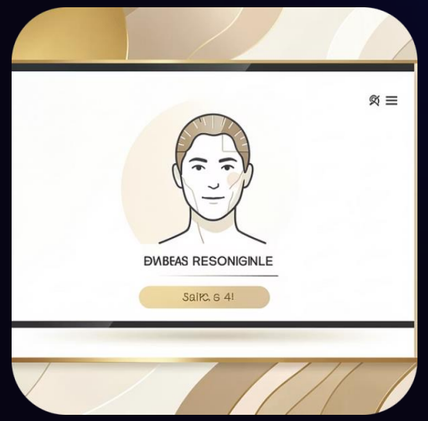
Gemini Historical Artifact Description

Leveraging Google Gemini Generative AI to create detailed historical descriptions of artifacts. This project showcases the practical application of AI in making history accessible.



Introduction



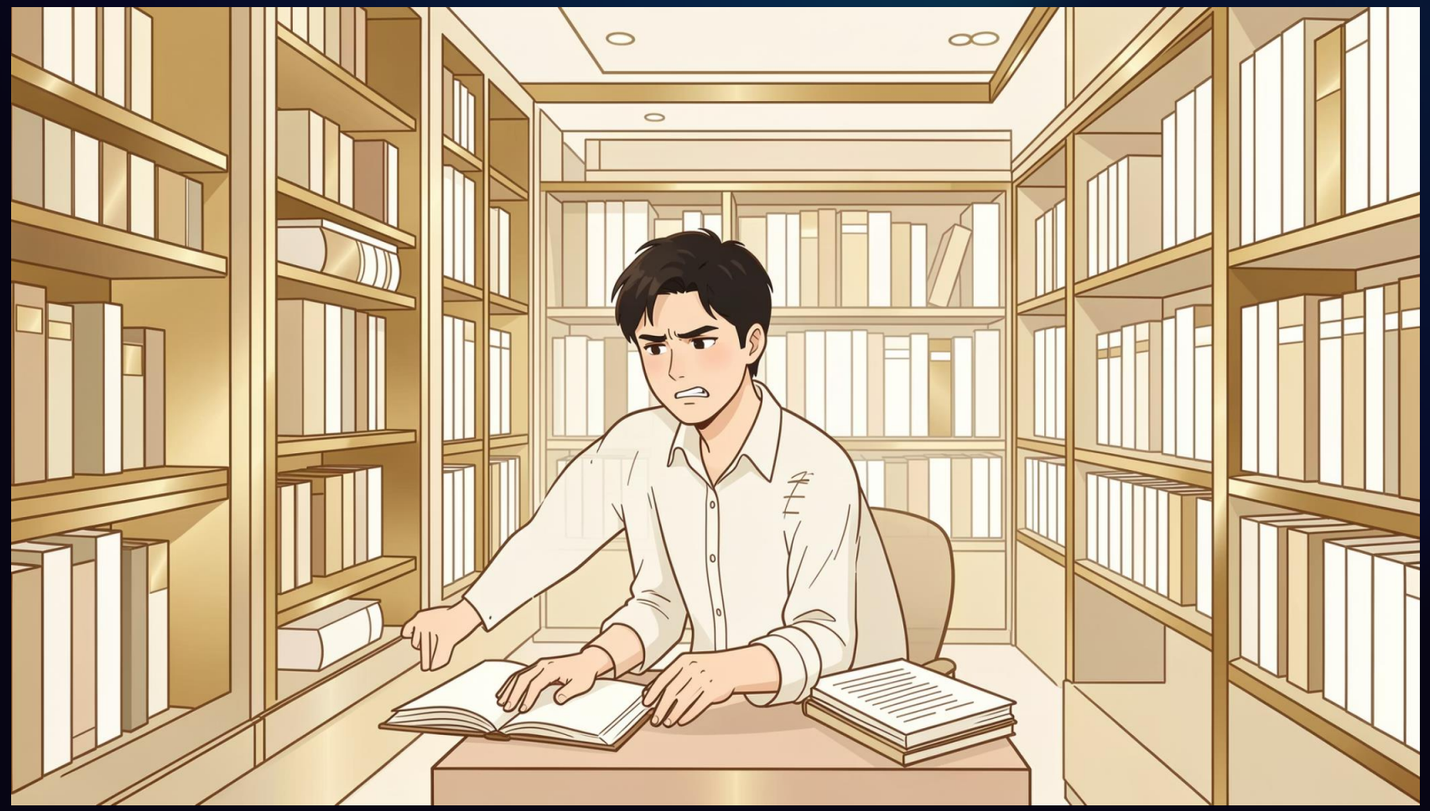
Generative AI Power

Generative AI models, like Google's Gemini, produce human-like responses for educational and informational purposes.

Project Focus

This project uses Gemini AI to generate detailed historical descriptions of artifacts, including origin, significance, and time period.

Problem Statement: Bridging the Knowledge Gap



Understanding historical artifacts often requires extensive research through books, museums, or expert knowledge. Many learners lack quick and interactive tools for easy exploration.

Instant Information

Provide immediate historical data.

User-Friendly

Simple and intuitive interface.

Web Accessibility

Accessible from anywhere via the web.

Project Objectives

01

Implement Gemini AI

Integrate Google Gemini Generative AI into a real-world application.

02

Build Web Application

Develop a user-friendly web application using Streamlit.

03

Generate Accurate Descriptions

Ensure AI produces precise historical information.

04

Deploy to Cloud

Host the application on a cloud platform for accessibility.

05

Secure API Management

Understand and implement secure API key handling in cloud environments.

Scope & Technologies

Project Scope

- Text-based artifact descriptions.
- Utilises Google Gemini Flash model.
- Cloud deployment via Streamlit Cloud.
- Secure API key handling with Streamlit Secrets.
- No image processing due to quota limits.

Technologies Used

Python	Programming language
Streamlit	Web application framework
Google Gemini API	Generative AI model
Google AI Studio	API key generation
Streamlit Cloud	Application deployment

System Architecture

The system efficiently processes user input to generate historical descriptions, ensuring a smooth flow from query to result.

Methodology & Implementation

Methodology

- Streamlit web interface for user input.
- Google Gemini API integrated with Python.
- Prompts designed for accurate AI responses.
- Secure API key handling via environment variables.
- Deployed on Streamlit Cloud.
- Tested for accuracy and reliability.

Implementation Details

- Accepts text input from users.
- Gemini Flash model generates responses.
- Prompts guide structured historical information.
- Graceful error and empty input handling.
- Application is deployed and accessible via a live URL.

Successful Output & Challenges Overcome

Successful Output Examples

- Taj Mahal
- Ashoka Pillar
- Mohenjo-daro
- Qutub Minar

Output includes artifact name, origin, historical importance, and approximate time period.

Challenges Faced & Resolved

- API quota limitations (switched to text-based, quota-friendly models).
- Model compatibility issues (selected appropriate Gemini models).
- Secure API key handling (used Streamlit Secrets).

Advantages & Applications

User-Friendly

Simple and easy to use with fast response times.

Accessible Anywhere

Cloud-based, eliminating manual research.

Secure & Scalable

Designed for secure and scalable operation.

Educational Tool

Ideal for educational platforms, history learning, and digital museums.

Future Enhancements & Conclusion

Future Enhancements

- Multilingual support.
- Reintroduce image-based analysis (with paid quota).
- Improved UI design.
- User query analytics.
- Voice input integration.

Conclusion

The project successfully integrates Google Gemini Generative AI with a web application, providing an effective solution for interactive historical knowledge generation. It highlights the practical application of Generative AI and cloud deployment, meeting all project requirements.