

GEMINI HISTORICAL ARTIFACT DESCRIPTION

TEAM DETAILS:

Team ID : LTVIP2026TMIDS86614

Team Lead: Darisi Venkata Lahari

Team Member: Kalla Rama Sri Sai Sandeep

Team Member: Anitha Maganti

Team Member: B Vinay Mohan Krishna

1. INTRODUCTION

1.1 Project Overview:

The **Gemini Historical Artifact Description App** is an GENAI-powered multimodal application designed to bridge the gap between technology and history. It leverages the **Gemini 2.5 Flash** model to analyze uploaded images of historical artifacts and generate insightful descriptions, including origin, significance, and time period.

1.2 Purpose:

The purpose of this project is to provide accurate and precise descriptions of artifacts directly from images, eliminating the need for manually maintaining extensive physical records. This assists educators, students, and curators in better understanding and documenting cultural heritage worldwide.

2. IDEATION PHASE

2.1 PROBLEM STATEMENT:

Historical artifacts hold great cultural and educational value, but they often lack clear descriptions especially in digital platforms or museums. This project addresses that problem by providing accurate and precise description of the artifact from image and avoiding the need of manually maintaining details about the artifact. This helps the educators, students, and curators better understand and document the artifacts from all around the world.

2.2 Empathy Map Canvas:

- **Educators/Students:** Need quick, reliable access to artifact details without searching through multiple archives.
- **Curators:** Require a tool to assist in the initial identification and descriptive documentation of new artifacts.

2.3 Brainstorming:

To solve the identification gap, we conceptualized a system that uses **Computer Vision** and **Large Language Models (LLMs)**. By combining image processing with historical context prompts, the system acts as a "Digital Historian".

3. REQUIREMENTS:

- **3.1 Functional Requirements:**

The interface should allow users do the following tasks:

- Uploading the image of an artifact (.png, .jpeg, .jpg).
- Input the image along with optional prompts by the user.
- Generate a description of the artifact given by the user.
- Download the description generated as a text file.

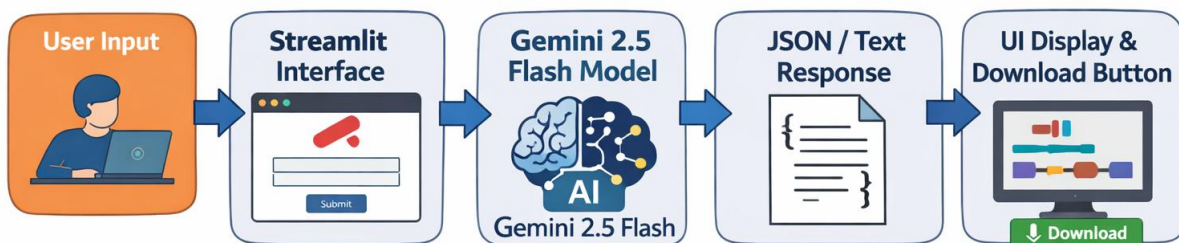
- **3.2 Non-Functional Requirements:**

The non-functional requirements include:

- Providing an accurate description to the user prompt.
- User should easily be able to access the features of the artifact description through a user-friendly interface along with safety integrated through API keys and environment variables

3.3 Data Flow Diagram:

The application follows a linear, sequential flow from user input to final data export:



Detailed Flow Breakdown:

1. **User Input:** The user provides an artifact image (JPG/PNG) and an optional text prompt through the Streamlit frontend.
2. **Streamlit Interface:** The application captures the file and converts the raw image data into a base64 or byte-stream format suitable for the API.
3. **Gemini 2.5 Flash Model:** Sends a multimodal request (Image + Prompt + System Instructions) to the Gemini 2.5 Flash model.
4. **JSON/Text Response:** The Gemini model processes the visual features and returns a structured text response containing historical insights.

5. **UI Display & Download Button:** Renders the description on the main page and enables a download button for local saving.

3.4 Technology Stack:

- **Frontend:** Streamlit Used to build the web interface, providing a responsive and interactive platform for image uploads and real-time analysis.
- **AI Engine:** Google Gemini 2.5 Flash A multimodal model that processes text and images simultaneously to identify artifacts and provide historical context.
- **Image Handling:** Pillow (PIL) Handles image processing, ensuring uploaded files like .jpg and .png are correctly formatted for display and analysis .
- **Environment:** Python-dotenv Secures the application by managing the Gemini API key as an environment variable, preventing sensitive data from being exposed in the code.

4. PROJECT DESIGN:

The project folder includes the following files:

➤ **requirements.txt**

The file contains the Python libraries and packages needed to be installed for the project.

➤ **README**

Provides an overview of the project.

➤ **app.py**

This Python folder contains all the main code which helps make this generative model work

➤ **.env**

Helps in storing sensitive and vulnerable information such as API keys.

➤ **secrets.toml**

Contains local configuration settings and secrets specifically for Streamlit's internal management.

➤ **.streamlit**

The configuration folder used by Streamlit to manage local settings and theme preferences.

➤ **.gitignore**

Ensures that sensitive files like .env and secrets.toml are not uploaded to public repositories.

4.1 CODE:

- Requirements.txt

```
Streamlit
google-generativeai
Pillow
python-dotenv
```

- app.py

```
app.py > ...
1  import os
2  from dotenv import load_dotenv
3  import streamlit as st
4  from google import genai
5  from PIL import Image
6
7  # Milestone 2: Configure API key
8  # 1. Load local .env file (useful for your local VS Code testing)
9  load_dotenv()
10
11 # 2. Step 2 Update: Try to get the key from Streamlit Secrets (Cloud)
12 # If not found in Secrets, it falls back to your local environment variable (.env)
13 api_key = st.secrets.get("GOOGLE_API_KEY") or os.getenv("GOOGLE_API_KEY")
14
15 if not api_key:
16     st.error("Missing GOOGLE_API_KEY. Please set it in Streamlit Secrets or a .env file.")
17     st.stop()
```

```

# Initialize the modern unified SDK with the retrieved key
client = genai.Client(api_key=api_key)

def get_gemini_response(input_text, image, prompt):
    # Milestone 4: Using Gemini 2.5 Flash
    response = client.models.generate_content(
        model="gemini-2.5-flash",
        contents=[prompt, image, input_text]
    )
    return response.text

# Page Configuration & Header
st.set_page_config(page_title="Gemini Historical Artifact Description")
st.header("🖼️ Gemini Historical Artifact Description App")

# Step 5.2: User Inputs
input_text = st.text_input("Input Prompt (e.g., 'Focus on its religious use'):", key="input")

uploaded_file = st.file_uploader(
    "Choose an image of a historical artifact...",
    type=["jpg", "jpeg", "png"]
)

```

```

# Step 5.3: Show Uploaded Image
if uploaded_file:
    image = Image.open(uploaded_file)
    # Replaced deprecated use_column_width with width="stretch"
    st.image(image, caption="Uploaded Image", width="stretch")

# Step 5.4: Generate Button
submit = st.button("Generate Artifact Description")

input_prompt = """
You are a historian. Analyze the historical artifact in the image and provide:
- Name of the artifact
- Origin
- Historical significance
- Approximate time period
"""

```



```
# Milestone 5: Output Generation
if submit:
    if uploaded_file:
        try:
            image = Image.open(uploaded_file)

            with st.spinner("📄 Historian is analyzing..."):
                response = get_gemini_response(input_text, image, input_prompt)

            st.subheader("📄 Description of the Artifact:")
            st.write(response)

            # Add the Download Button
            st.download_button(
                label="📄 Download Description as TXT",
                data=response,
                file_name="artifact_description.txt",
                mime="text/plain"
            )
        except Exception as e:
            st.error(f"Error: {str(e)}")
    else:
        st.warning("⚠️ Please upload an image first!")
```

4.2 CODE OVERVIEW:

- The necessary libraries and packages are present in the requirements.txt
- The .env file store the API key and it securely loaded and setup for the project.
- The input_image_setup() function prepares the uploaded image so it can be sent to the API.
- The get_gemini_response() function accepts the user input , sends them to the API and returns the generated description.
- Streamlit allows the interface to accept user input along with prompt and view the description generated in the main area of the page.
- A specialized prompt instructing the AI to act as a historian and provide details on the artifact's name, origin, time period, material, and cultural significance.
- Custom CSS is applied to create a "centered" layout with a professional "Georgia" serif font for the title and a dark "Verdana" theme for the sidebar.
- The error is displayed if the user does not upload any image while trying to generate a description.

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning:

- Week 1: Requirements gathering and tech stack selection.
- Week 2: UI design and CSS styling (Title, Sidebar, and Output Area).
- Week 3: Integration with Gemini API and error handling.
- Week 4: Testing, deployment on Streamlit Cloud, and documentation.

Week	Focus Area	Tasks Covered
Week 1	Requirements & Planning	Requirements gathering and tech stack selection
Week 2	UI Development	UI design and CSS styling (Title, Sidebar, and Output Area)
Week 3	Backend Integration	Integration with Gemini API and error handling
Week 4	Testing & Deployment	Testing, deployment on Streamlit Cloud, and documentation

6. FUNCTIONAL AND PERFORMANCE TESTING:

- **Case1:** When only image is uploaded

Gemini Historical Artifact Description App

Input Prompt (e.g., "Focus on its religious use"):

Choose an image of a historical artifact...



Drag and drop file here

Limit 200MB per file • JPG, JPEG, PNG

Browse files



f98b8117187169ac03ca81458574f44f.jpg 26.2KB



Uploaded Image

Generate Artifact Description



Description of the Artifact:

Based on the visual evidence, here's an analysis of the historical artifact:

Name of the artifact: Bas-relief sculpture of Lord Shiva (specifically, a four-armed seated manifestation).

Origin: This intricate stone carving originates from the **Hoysala Dynasty** of South India, specifically from the region of **Karnataka**. It is typical of the sculptural style found on the outer walls of their famous temples.

Historical Significance:

1. **Religious Devotion:** The sculpture unequivocally depicts Lord Shiva, one of the principal deities of Hinduism. His attributes – the axe (parashu) in his upper left hand, the cobra (naga) in his upper right, and the possible bull (Nandi, his vehicle) seen faintly at the lower left base – confirm his identity. This signifies the strong prevalence of Shaivism (worship of Shiva) during the Hoysala period.
2. **Artistic Excellence of the Hoysala Dynasty:** The carving is an outstanding example of Hoysala art,

The description is successfully generated in Case 1.

- **Case2: When image is uploaded along with prompt.**



Gemini Historical Artifact Description App

Input Prompt (e.g., 'Focus on its religious use'):

Analyze the uploaded image of a historical artifact. Identify the artifact, estimate its historical period, or

Choose an image of a historical artifact...



Drag and drop file here

Limit 200MB per file • JPG, JPEG, PNG

Browse files



f98b8117187169ac03ca81458574f44f.jpg 26.2KB



Uploaded Image

Generate Artifact Description



Description of the Artifact:

Here's an analysis of the historical artifact in the image:

- **Name of the Artifact:** Carved relief depicting **Vishnu**. More specifically, it shows Vishnu in a seated posture, likely *Lalitasana* (royal pose) or *Sukhasana* (relaxed pose), holding his characteristic attributes. This form of Vishnu is often referred to by specific names depending on the exact order and combination of his attributes (conch, discus, mace, lotus). Given the attributes clearly visible (chakra/discus in upper right, shankha/conch in lower right, gada/mace in lower left, and possibly a lotus or a gesture in upper left which is slightly obscured), it is definitively a form of Vishnu.
- **Origin:** The intricate carving style, particularly the detailed ornamentation, serene facial expression, and the use of soft chloritic schist (soapstone) which allows for such minute detail, is highly characteristic of **Hoysala art from Karnataka, South India**. Famous examples of such craftsmanship can be found in the temples of Belur, Halebidu, and Somanathpur.
- **Approximate Time Period:** This style of carving flourished during the **Hoysala Empire**, typically from the **12th to 13th century CE**. The zenith of Hoysala temple building and sculptural art was during this period.

The description is successfully generated in Case 2

importance of Vaishnavism during the Hoysala period.

2. **Artistic Masterpiece:** It represents the pinnacle of Hoysala sculptural art. Hoysala sculptors (known as *shilpis*) were renowned for their extraordinary skill in carving intricate details, jewelry-like ornamentation, and lifelike expressions from a relatively soft stone (chloritic schist). The precision and depth of carving, even in high relief, are remarkable.
3. **Architectural Context:** This relief would have been an integral part of a larger temple structure, adorning its exterior walls or pillars. Hoysala temples are known for being almost entirely covered in such narrative and iconic sculptures, depicting deities, mythological scenes, and celestial beings.
4. **Cultural Insight:** The carving provides valuable insight into the religious iconography, aesthetic values, and the high level of craftsmanship patronized by the Hoysala rulers. It speaks to a sophisticated culture that valued both religious devotion and artistic excellence.
5. **Material Usage:** The choice of chloritic schist (soapstone) was deliberate, as it allowed for the fine details that are a hallmark of Hoysala art, which would have been much harder to achieve with harder granites.

In summary, this Vishnu relief is a magnificent example of Hoysala artistry from 12th-13th century South India, embodying the religious devotion, advanced sculptural techniques, and rich cultural patronage of its time.

 Download Description as TXT

The description can be downloaded by clicking on the download button.

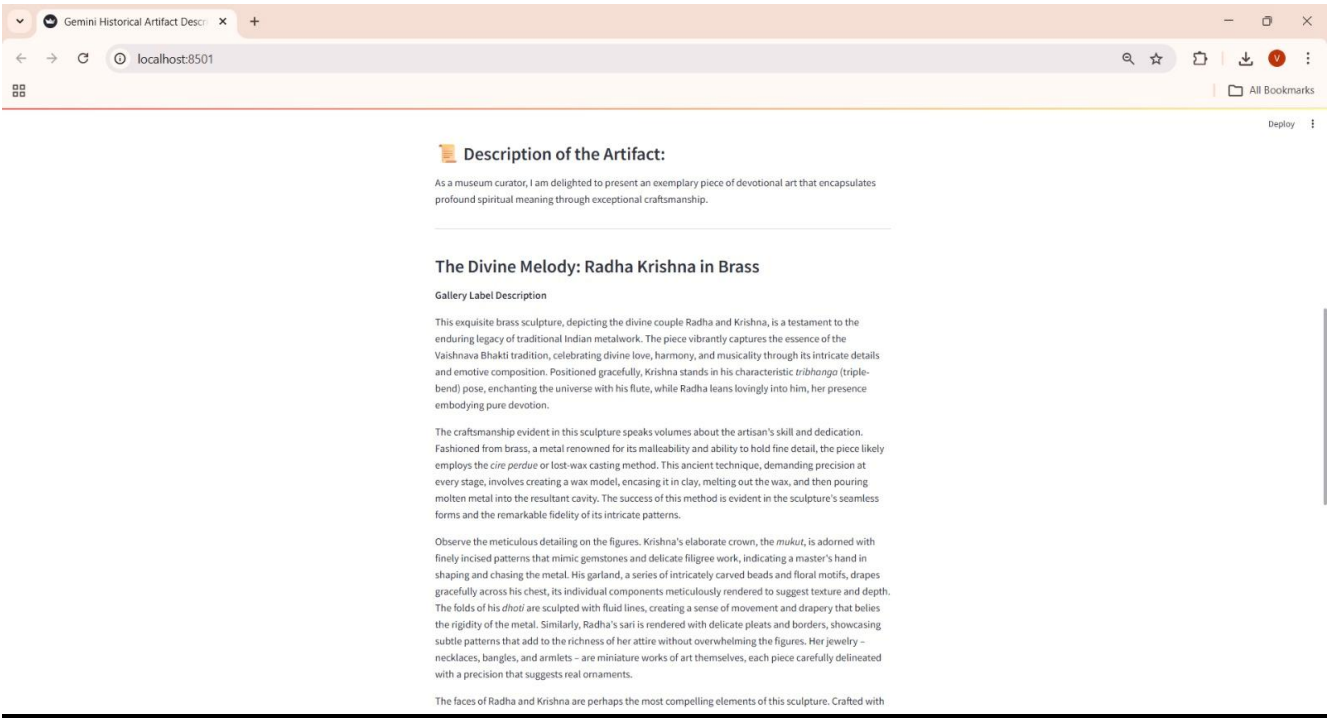
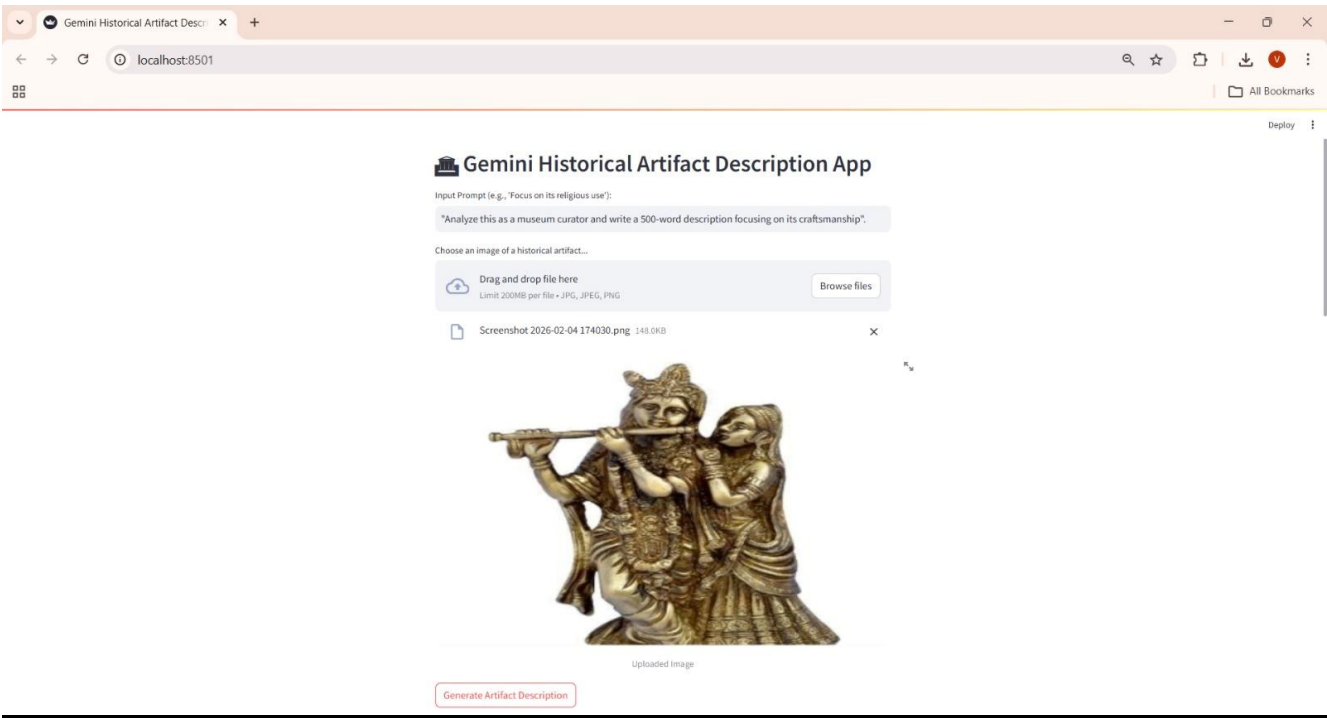
- **Case3 : When only prompt is given or when the image limit exceeds .**

In the above case, the interface produces an error.

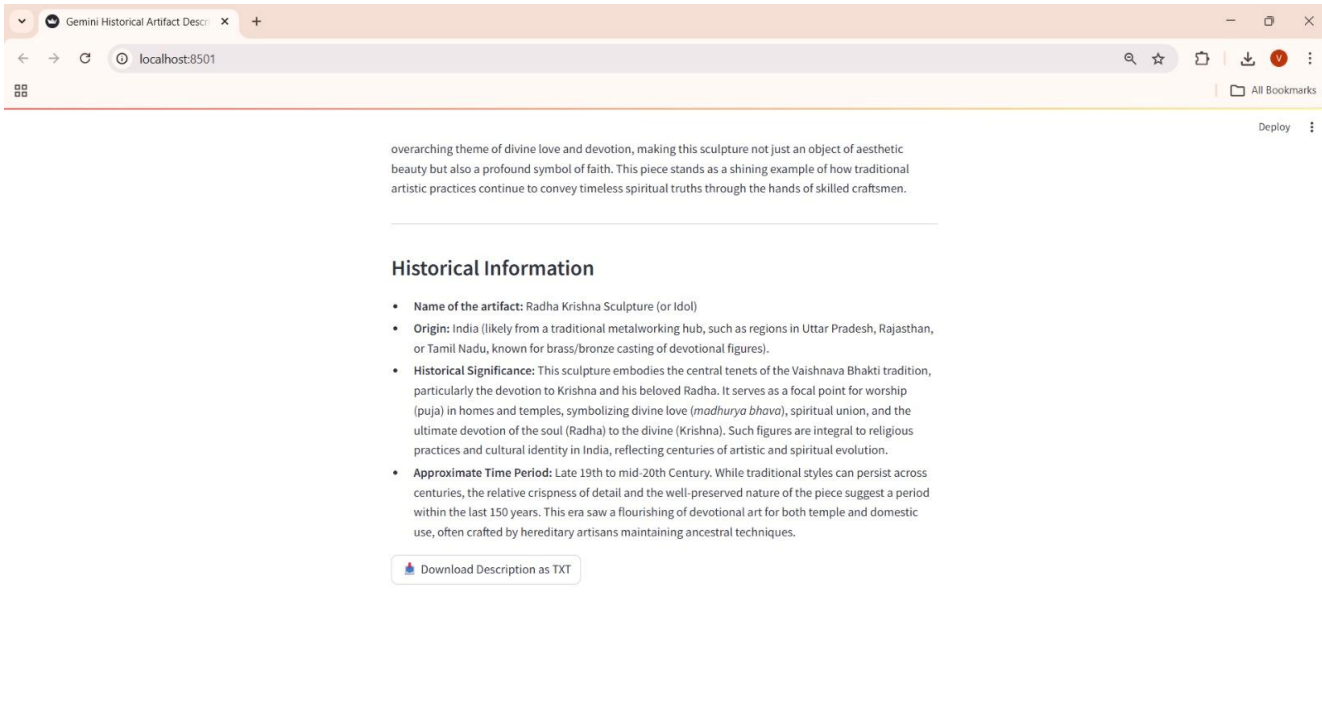


7. RESULTS

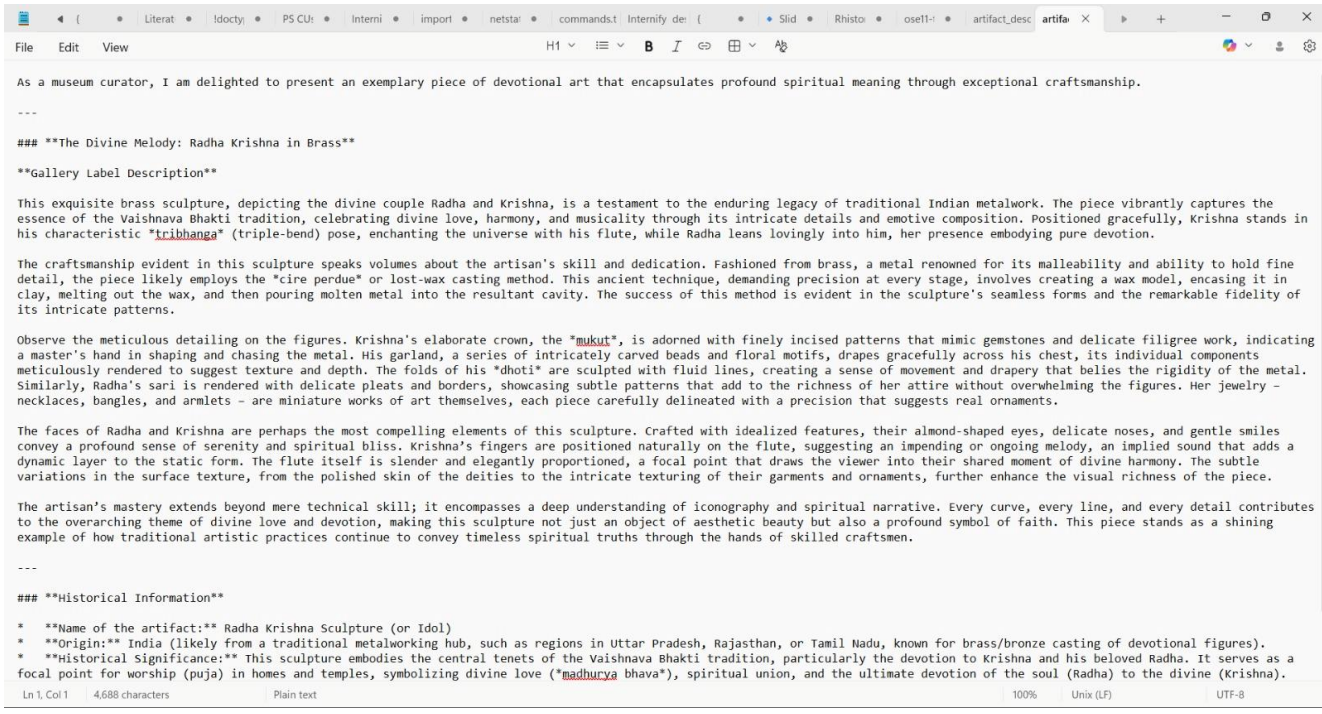
7.1 Output Screenshots:



Artifact Analysis: Successful identification of an Image artifact with detailed bullet points on name, period, and significance.



Download Feature: Functional button allowing the export of the description to artifact_description.txt.



8. ADVANTAGES & DISADVANTAGES

ADVANTAGES:

- User-Friendly interface
- Generates the description within seconds
- Useful for people who want to educate themselves on artifacts and don't have the resources.
- Provides a valuable learning resource for those who lack access to professional historical experts or museum curators.

DISADVANTAGES:

- Dependency on an external API.
- AI might face the problem of hallucination without proper access to current data.
- Internet Requirement The application cannot function without a stable internet connection to communicate with the cloud-based AI.
- Image Quality Sensitivity The accuracy of the description depends heavily on the clarity and lighting of the uploaded artifact image.

9. **CONCLUSION:**

The Gemini Historical Artifact Describer demonstrates how generative AI can support cultural heritage documentation by creating detailed artifact descriptions from images and user prompts. By leveraging the Gemini 1.5 Flash model, the project bridges the gap between complex historical data and digital accessibility, providing a user-friendly and secure platform for educators, students, and the public. While the system remains dependent on external APIs and the potential for AI hallucinations, it offers a highly efficient resource for identifying the origin and cultural significance of global artifacts. Ultimately, this project serves as a scalable proof-of-concept for utilizing modern AI frameworks to preserve and share history at the ease of a user's fingertips.

10. **FUTURE SCOPE:**

- **Multilingual Support:** Implement automatic translation to provide artifact descriptions in multiple global languages for international accessibility.
- **Interactive AI Chatbot:** Integrate a conversational interface that allows users to ask follow-up questions for a deeper, interactive learning experience.
- **AR Visualization:** Develop Augmented Reality features to overlay historical data and 3D annotations directly onto artifacts through a mobile camera view.

11. APPENDIX:

- **Source Code:** Hosted on GitHub.
- **GitHub Link:** <https://github.com/LahariDarisi/gemini-historical-artifact-description-app>
- **Project Demo Link:** <https://gemini-historical-artifact-description-app-fbbmzghmx96joo6plkc.streamlit.app/>