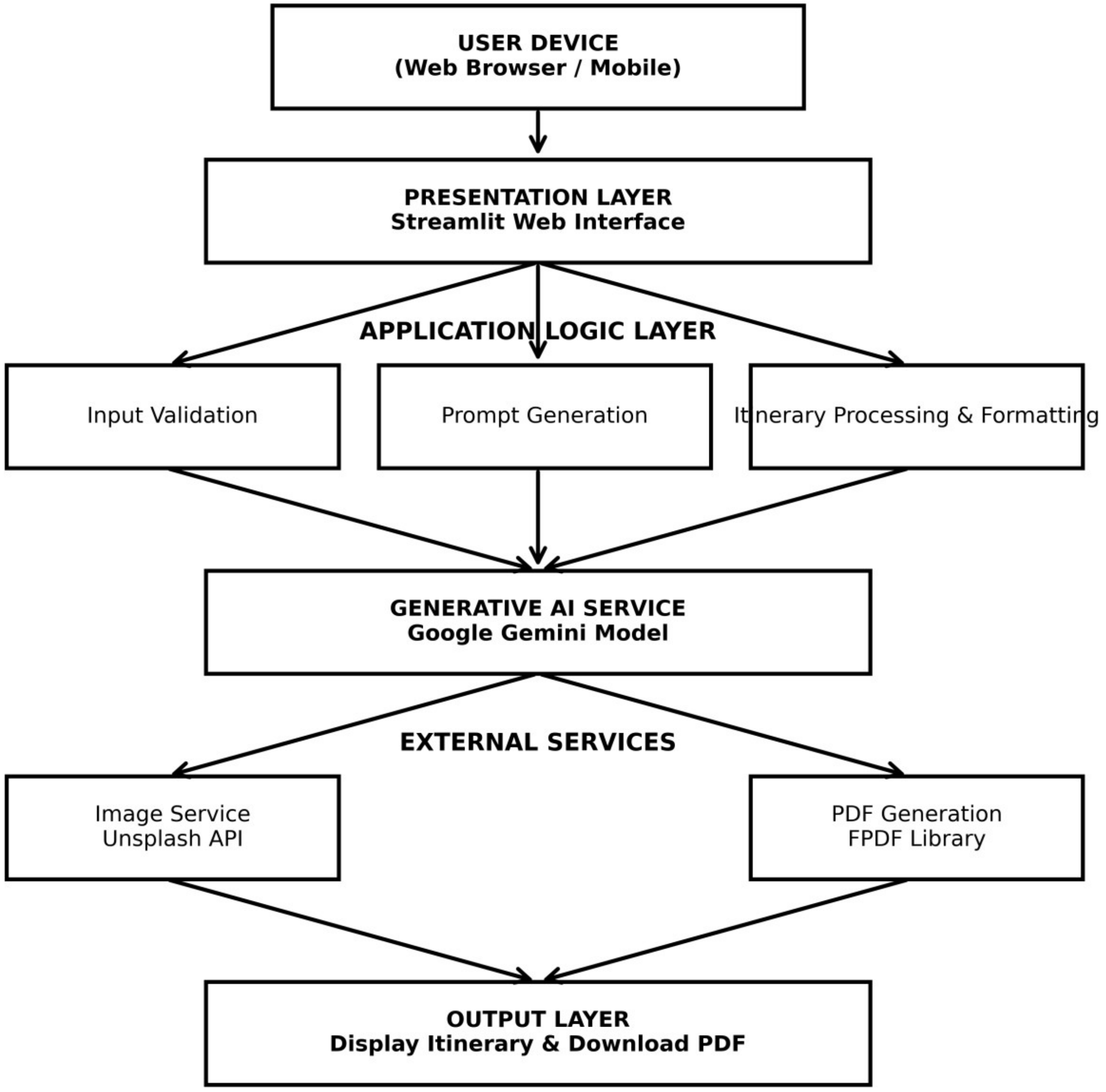
Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	31 January 2026
Team ID	LTVIP2026TMIDS26163
Project Name	Explore with AI: Custom Itineraries for Your Next Journey
Maximum Marks	4 Marks

Technical Architecture:

AI TRAVEL ITINERARY GENERATOR



USER DEVICE
(Web Browser / Mobile)

PRESENTATION LAYER
Streamlit Web Interface

APPLICATION LOGIC LAYER

Input Validation

Prompt Generation

Itinerary Processing & Formatting

GENERATIVE AI SERVICE
Google Gemini Model

EXTERNAL SERVICES

Image Service
Unsplash API

PDF Generation
FPDF Library

OUTPUT LAYER
Display Itinerary & Download PDF

Table-1 : Components & Technologies:

Table-1: Component & Technologies:

S.No	Component	Description	Technology
1.	User Interface	Web-based user interaction to enter trip details and view itineraries	Streamlit (App UI library), Python
2.	Application Logic-1	Logic to validate user input for trip details	Python, Streamlit Validation Functions
3.	Application Logic-2	Logic to dynamically construct prompts for AI generation	Python
4.	AI Model	AI model to generate itinerary based on user details	Gemini API (Google)
5.	Image Integration	Service/API to fetch destination-related images	Unsplash API
6.	PDF Generation	Logic to format itinerary content and build PDF	Python, PyPDF2
7.	File Storage	Temporary storage and download of generated PDF	Python
8.	PDF Download Feature	Fetch destination image Using Unsbash API	Python
9.	File Storage	Temporary storage and download of generated PDF	Python

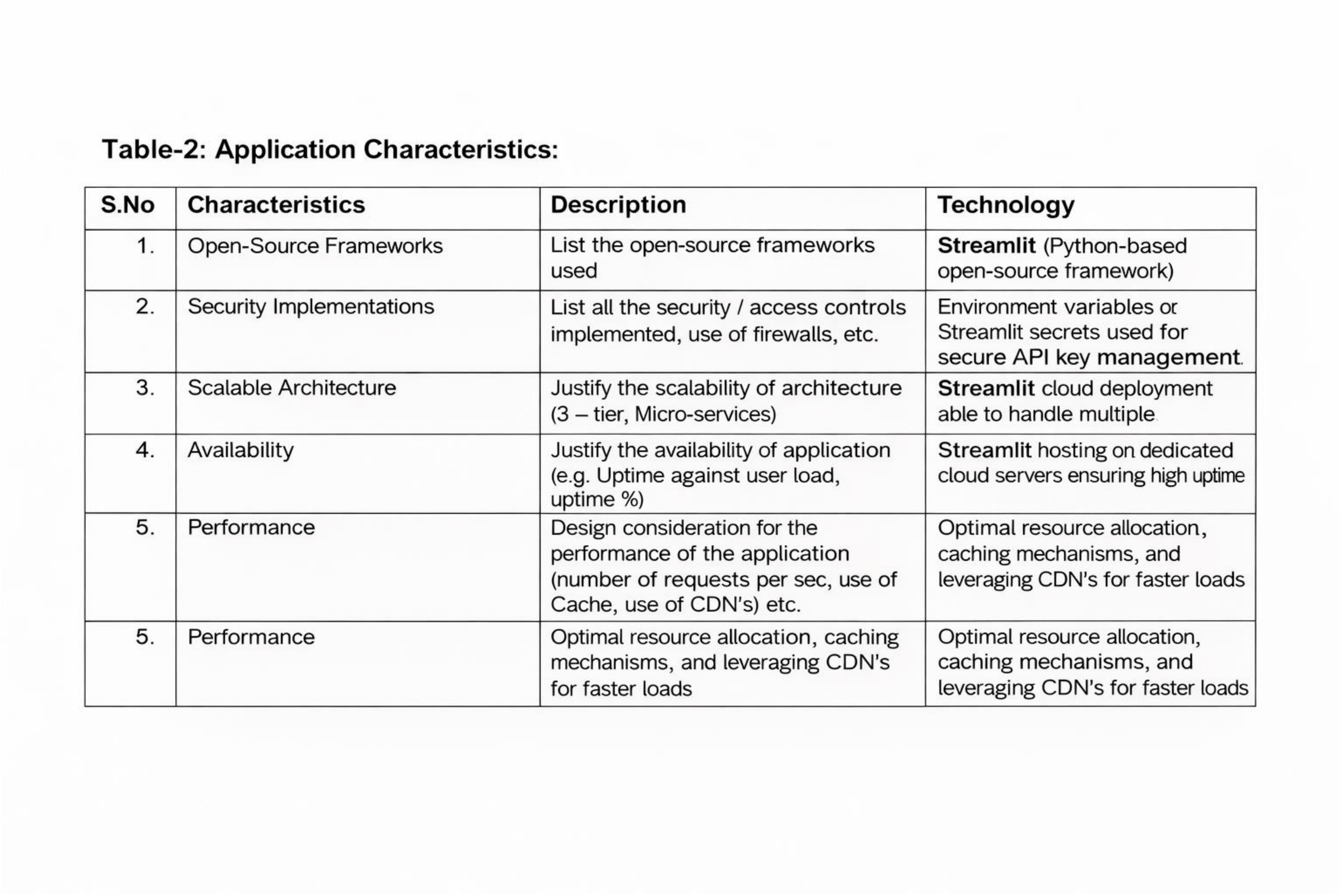


Table-2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source Frameworks	List the open-source frameworks used	**Streamlit** (Python-based open-source framework)
2.	Security Implementations	List all the security / access controls implemented, use of firewalls, etc.	Environment variables or Streamlit secrets used for secure API key management.
3.	Scalable Architecture	Justify the scalability of architecture (3 – tier, Micro-services)	**Streamlit** cloud deployment able to handle multiple.
4.	Availability	Justify the availability of application (e.g. Uptime against user load, uptime %)	**Streamlit** hosting on dedicated cloud servers ensuring high uptime
5.	Performance	Design consideration for the performance of the application (number of requests per sec, use of Cache, use of CDN's) etc.	Optimal resource allocation, caching mechanisms, and leveraging CDN's for faster loads
5.	Performance	Optimal resource allocation, caching mechanisms, and leveraging CDN's for faster loads	Optimal resource allocation, caching mechanisms, and leveraging CDN's for faster loads