INT-428(AI Essentials)

CA-3 (Skill-Based)

Submitted By:

S no.	Name	Registration No.	Roll No.	Section
1.	Sadasivuni Sai Vivek	12300999	02	K23WG
2.	Chukka Victor Prasad	12305255	50	K23WG
3.	Ann Mariya Rose Pereira	12306793	52	K23WG

Submitted To: **Mr. Vikas Sharma**

LOVELY PROFESSIONAL UNIVERSITY

1. Project Overview:

1.1 Title:

PHOTOGRAPHY GUIDANCE BOT

1.2 Introduction:

The Photography Assistant project is a desktop-based intelligent application designed to assist photographers in capturing optimal images by providing real-time suggestions based on current weather conditions and image parameters. The project integrates concepts of Artificial Intelligence, API integration, and image processing techniques, brought together in a Tkinter-based Python GUI.

The application features a clean full-screen login system, an interactive main interface, and core modules that support image upload, enhancement, RGB histogram visualization, and weather-aware photography tips. The interface is designed to be intuitive and accessible, even for amateur photographers.

The major objective of this project is to enhance the quality of photography by providing AI-driven suggestions in real time. By fetching live weather data via the OpenWeatherMap API, the application dynamically displays a weather-based popup that influences suggestions on ISO, aperture, lighting, and composition for best results.

Additionally, the tool enables users to analyze RGB

histograms of their images, helping them understand light balance and exposure settings better than conventional trial-and-error methods.
With this application, photography becomes more guided, intelligent, and weather-aware—helping users to make informed decisions before clicking the shutter.

1.3 Objective:

1. To develop a Photography Assistant desktop application using Python and Tkinter that assists users with intelligent photography tips.

2. To integrate real-time weather-based suggestions using OpenWeatherMap API for optimized photography based on current environmental conditions.

3. To design a user-friendly full-screen interface with authentication and a main dashboard for image enhancement and analysis.

4. To implement features such as RGB histogram visualization and basic image editing tools for analyzing and improving image quality.

5. To enhance the photography experience by bridging AI-driven insights with user creativity, ultimately leading to better composition and technical camera control.

2. Design and Architecture:

2.1 Interface Layer (Frontend):

- The **interface layer** of the Photography Assistant application is built using **Python's Tkinter library**, providing a graphical user interface (GUI) that is both interactive and user-friendly. The application launches in a **fullscreen mode with an authentication page**, ensuring secured access.

The dashboard consists of the following primary UI components:

- **Login Window**: A secure fullscreen login screen for user authentication before accessing the main module.
- **Main Dashboard**: After successful login, users can upload images, access image enhancement features, and receive smart photography tips.
- **Image Upload Section**: Allows users to select images from their local storage for analysis and enhancement.
- **Histogram Display Area**: Shows RGB histograms of uploaded images, helping users understand color distribution.
- **Real-Time Weather Popup**: Automatically fetches and displays **current weather conditions** as a popup at the top of the dashboard, assisting in outdoor photography planning.

- **Image Enhancement Tools**: Includes sliders and buttons for adjusting brightness, contrast, sharpness, and more.
- **Photography Tips Area**: Dynamic space providing suggestions based on image analysis and real-time weather inputs.

The frontend emphasizes **minimalism, clarity, and ease of use**, offering an intuitive experience for photographers of all skill levels.

2.2 Styling Layer:

- The **Styling Layer** in the Photography Assistant application is managed through the **Tkinter library's widget configuration and layout management system**. Although Tkinter does not support CSS like web platforms, it provides ample customization options using Python methods for styling and layout.

Key styling components include:

- **Theme Consistency**: A minimal and clean design is applied throughout the interface using a uniform color palette, padding, and font styling to maintain visual harmony.
- **Fullscreen Login Design**: The login window appears in fullscreen with customized fonts, input fields, and button styles to create a professional and secure entry point.
- **Real-Time Weather Popup Styling**: The weather popup is styled with a soft color background and

legible font, appearing as a non-intrusive top bar for better visibility and user interaction.

- **Image Display and Histogram Sections**: The layout ensures the image preview and RGB histogram are properly aligned and sized for clarity and usability.
- **Interactive Widgets**: Sliders, buttons, and text areas are styled using custom background colors, borders, and fonts for enhanced user experience.

The styling layer ensures that the application not only functions smoothly but also delivers a visually appealing and accessible environment for all users.

2.3 Logic Layer:

- The logic layer of the Photography Assistant desktop application is built entirely using Python, which handles core functionalities such as image processing, RGB histogram generation, and real-time weather-based suggestions. The application uses the Tkinter framework to integrate both GUI and backend logic, ensuring smooth interaction and real-time responsiveness. Image enhancements and RGB analysis are performed using the PIL (Pillow) module, while matplotlib is used to display dynamic histograms. Real-time weather data is fetched through the OpenWeatherMap API using the requests module, and context-based photography suggestions are shown in a non-

intrusive popup at the top of the interface. Additionally, the login and authentication system is managed through Python to ensure secure access within a full-screen UI. This logical architecture provides an interactive and efficient experience, with seamless integration between the visual interface and functional component.

2.4 Data Gathering Layer:

- The Photography Assistant desktop application gathers data dynamically through API integration to ensure real-time, accurate suggestions and analysis. Specifically, it utilizes the OpenWeatherMap API to retrieve current weather conditions, which are then used to provide contextual photography tips tailored to the ambient environment. The data is fetched using Python's requests module and processed instantly to be displayed via a popup on the application interface. This dynamic data gathering approach ensures that users receive up-to-date environmental information, enhancing the effectiveness of photography recommendations and overall user experience.

3. Implementation:

3.1 Python (Tkinter GUI Foundation)

- Python is used as the primary language for building the Photography Assistant application. The entire user interface is developed using the Tkinter library, which serves as the foundation for GUI layout and component arrangement
- It includes components like image upload buttons, login interface, RGB histogram canvas, and real-time photography suggestion panels.

```
from tkinter import *
from PIL import Image, ImageTk
import requests
import matplotlib.pyplot as plt
```

3.2 Tkinter Styling and Layout:

- Tkinter widgets are styled using attributes like bg, fg, font, and layout geometry managers such as pack(), grid(), and place() to maintain a responsive and structured appearance.
- Styling can be dynamically modified during runtime, and weather-based styling can also be applied for visual feedback.

```
frame = Frame(root, bg="#1f2937", padx=20, pady=20)

frame.pack(pady=10)
```

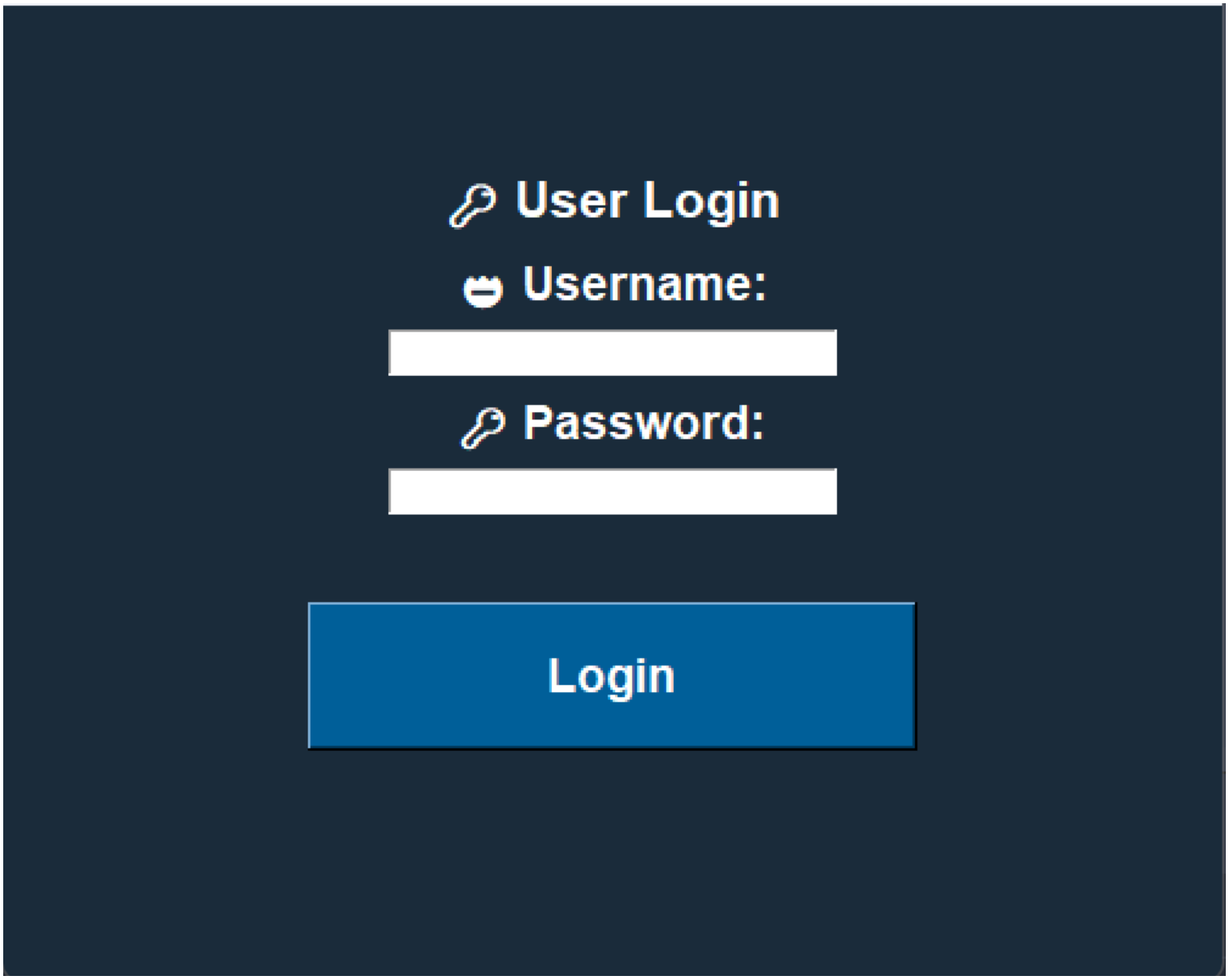
```
button = Button(frame, text="Upload Image", bg="#4ade80",
fg="white", font=("Arial", 12, "bold"))

button.pack()
```

3.3 Python Logic and API Integration:

- Python's procedural and object-oriented programming structure enables the implementation of all core logic including image enhancement, histogram generation, and live weather fetching
- API integration (such as OpenWeatherMap) allows real-time retrieval of weather data using the requests module, which helps provide photography tips based on current climatic conditions.

```
def get_weather(city):
    api_key = "your_api_key"
    url =
f"https://api.openweathermap.org/data/2.5/weather?q={city}&
appid={api_key}"
    response = requests.get(url)
    data = response.json()
    return data["weather"][0]["description"]
```

4. Dashboard Images:



Photography Guide

Lighting Tips:
• Use natural light for soft shadows.
• Golden hour (sunrise/sunset) provides warm tones.
• Avoid harsh midday light; use reflectors or diffusers.

Advanced Tip: Use ISO 100 and a shutter speed of 1/125s for optimal exposure.

Lighting | ISO | Shutter Speed | Aperture
Composition | Upload Image | Enhance Image | Histogram

Photography Guide

ISO Settings:
• Bright daylight: ISO 100-200
• Indoor/Night: ISO 800-3200
• Use a lower ISO for less noise, but higher ISO for low-light conditions.

*

Lighting | ISO | Shutter Speed | Aperture
Composition | Upload Image | Enhance Image | Histogram

Photography Guide

Shutter Speed Guide:
- 1/1000s for fast motion (sports, birds)
- 1/250s for portraits
- 1/30s for low light or creative blur.

Advanced Tip: Use a shutter speed of 1/125s for general outdoor photography.

Lighting | ISO | Shutter Speed | Aperture

Composition | Upload Image | Enhance Image | Histogram

Photography Guide

Aperture (f-stop) Guide:
- f/1.4 - f/2.8 for blurred backgrounds (portraits)
- f/4 - f/8 for balanced sharpness (landscapes)
- f/11+ for deep focus (architecture).

Advanced Tip: Use f/8 for landscapes to ensure depth of field.

Lighting | ISO | Shutter Speed | Aperture

Composition | Upload Image | Enhance Image | Histogram

Photography Guide

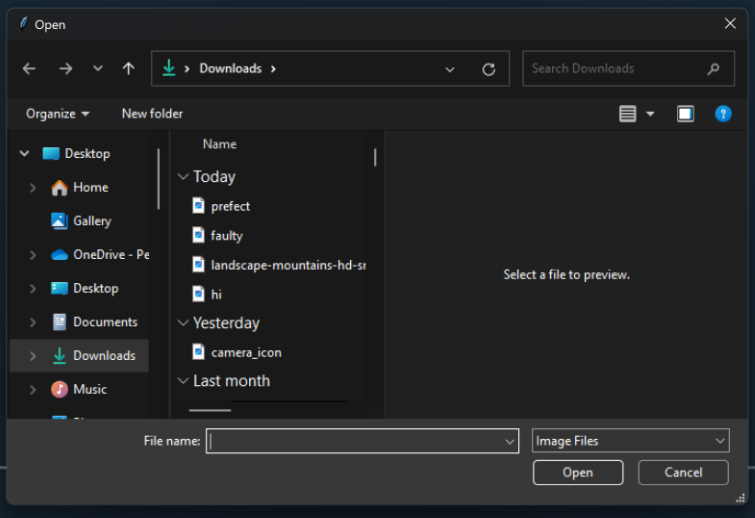
Composition Tips:
• Use the rule of thirds for balanced framing.
• Leading lines direct the viewer's eye.
• Experiment with perspectives for unique shots.

Lighting | ISO | Shutter Speed | Aperture

Composition | Upload Image | Enhance Image | Histogram

Photography Guide

Composition Tips:
• Use the rule of thirds for balanced framing.
• Leading lines direct the viewer's eye.
• Experiment with perspectives for unique shots.



Lighting | ISO | Shutter Speed | Aperture

Composition | Upload Image | Enhance Image | Histogram

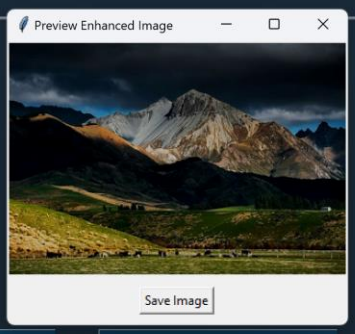
Photography Guide

Image uploaded successfully!
Image lacks contrast. Increase contrast by 20-30% using editing tools or decrease shutter speed slightly.

Lighting | ISO | Shutter Speed | Aperture
Composition | Upload Image | Enhance Image | Histogram

Photography Guide

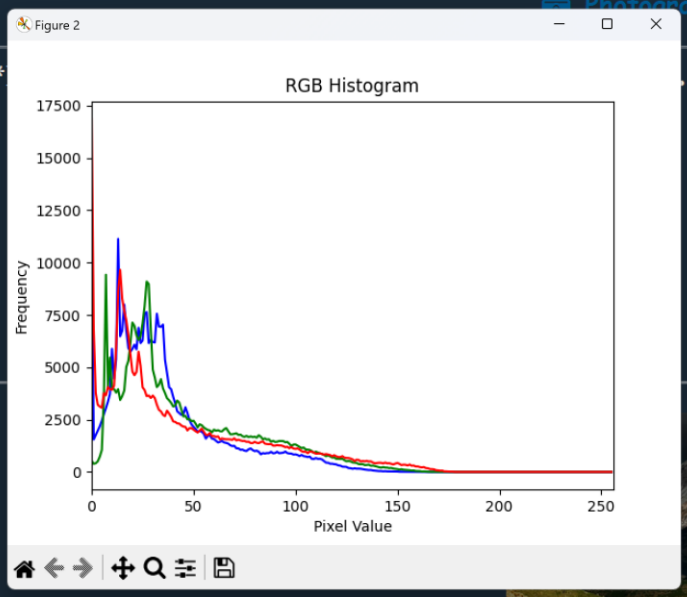
Image enhanced successfully! Previewing now...

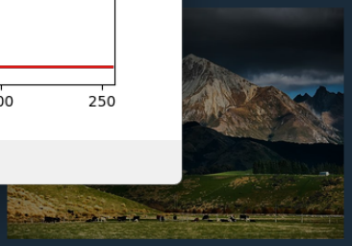


Lighting | ISO | Shutter Speed | Aperture
Composition | Upload Image | Enhance Image | Histogram

Photography Guide

Figure 2





Lighting | ISO | Shutter Speed | Aperture

Composition | Upload Image | Enhance Image | Histogram

5. Technologies & Tools Used:

1. Python

2. Tkinter

3. Pillow(PIL)

4. MatplotLib

5. Requests

6. OpenWeatherMap API

7. Cursor AI Code Editor

8. GitHub Copilot

9. Ttk

6. Conclusion:

- The Photography Assistant project successfully met all its primary objectives by delivering a real-time, interactive desktop application that provides personalized photography suggestions and tips based on both image analysis and live weather data.

- With the integration of Tkinter for GUI, Pillow for image enhancements, OpenWeatherMap API for real-time weather suggestions, and Matplotlib for RGB histogram visualization, the system ensures a comprehensive user experience for photographers of all skill levels.

- Finally, the project demonstrates the effective application of Artificial Intelligence, API integration, and Python-based GUI development in solving real-world challenges like photography optimization all within a lightweight desktop application environment, making it accessible and practical for everyday use.