

# Title: Detect and Analyze stenographic hidden data within digital files

## 1. Objective

The objective of this experiment was to detect and analyze steganographic content hidden within a digital file. This lab involved using a **custom-built Python-based steganography detection tool** to load a sample image, scan it for hidden data, and document the findings.

---

## 2. Tools and Environment Used

The following tools and environment components were used for this analysis:

- **Analysis Tool:** Custom Steganalysis Tool (Python-based GUI)
- **Key Libraries:** [PySide6, Pillow, matplotlib, numpy, scipy, exifread]
- **Sample File:**

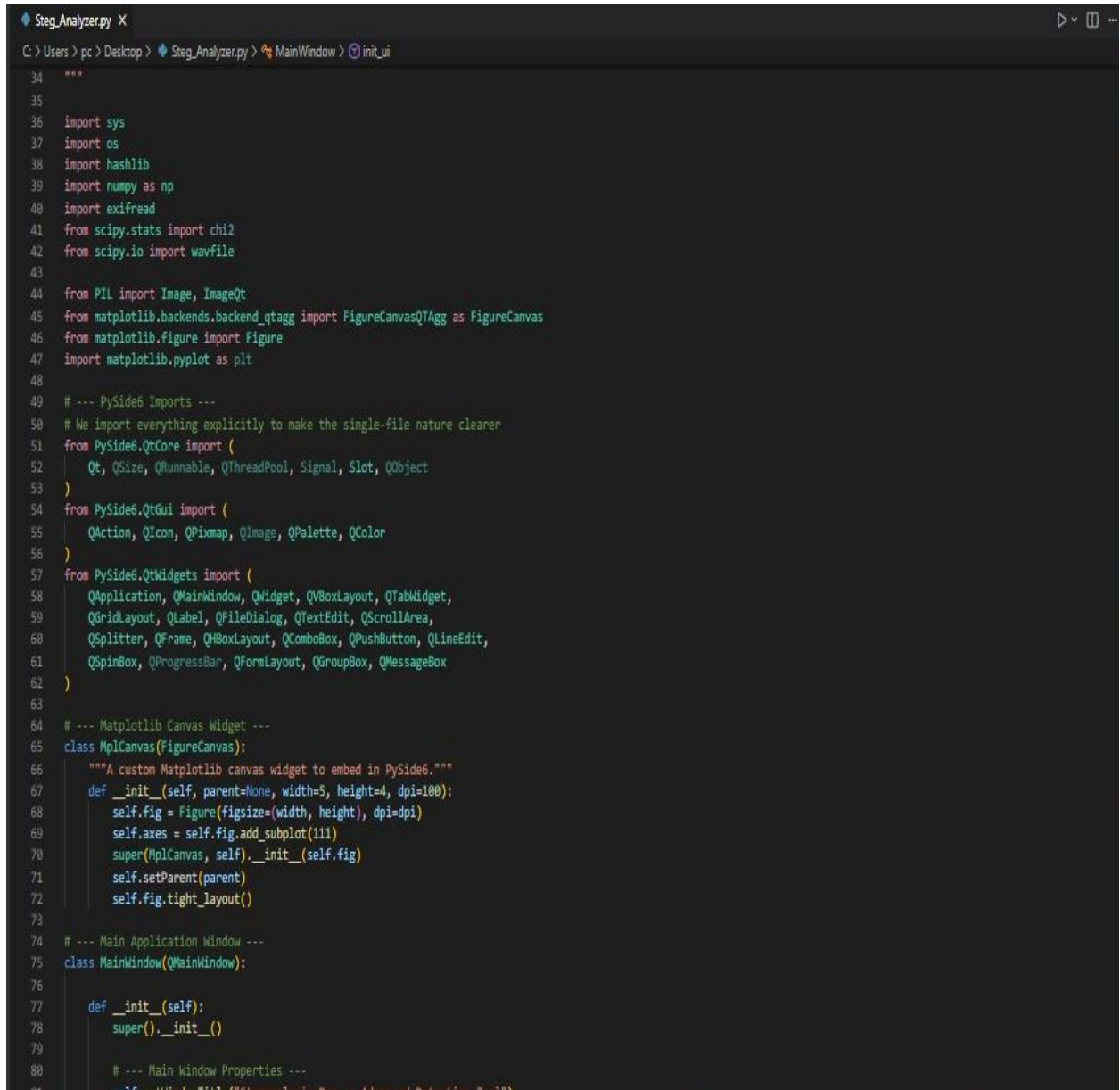


- **Operating System:** Window 10 64-bit
-

### 3. Step-by-Step Procedure

The following steps were performed using the custom analysis tool:

1. **Tool Launch:** The custom Python-based tool was launched.

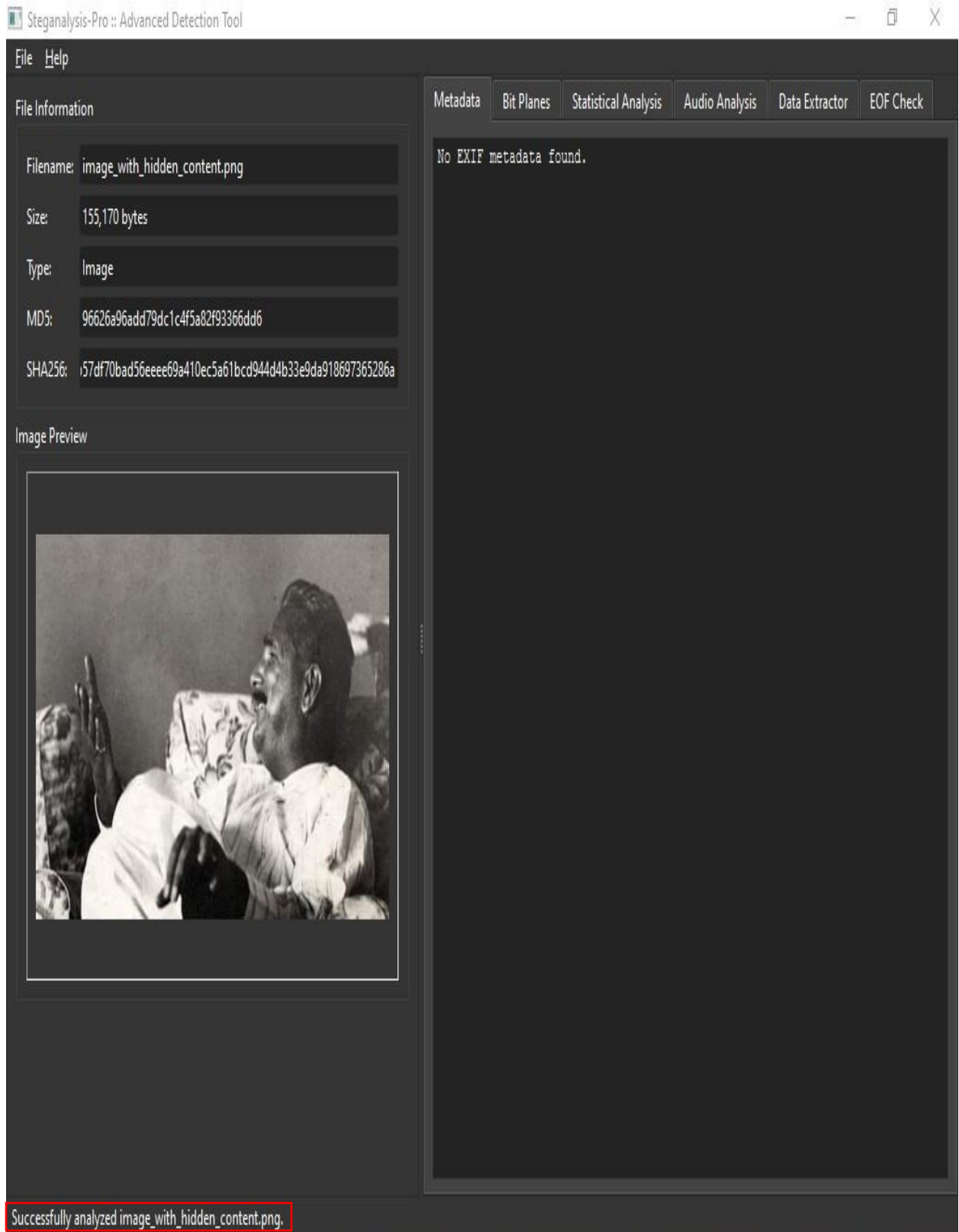


```
Steg_Analyzer.py X
C:\Users\pc\Desktop> Steg_Analyzer.py > MainWindow > init_ui

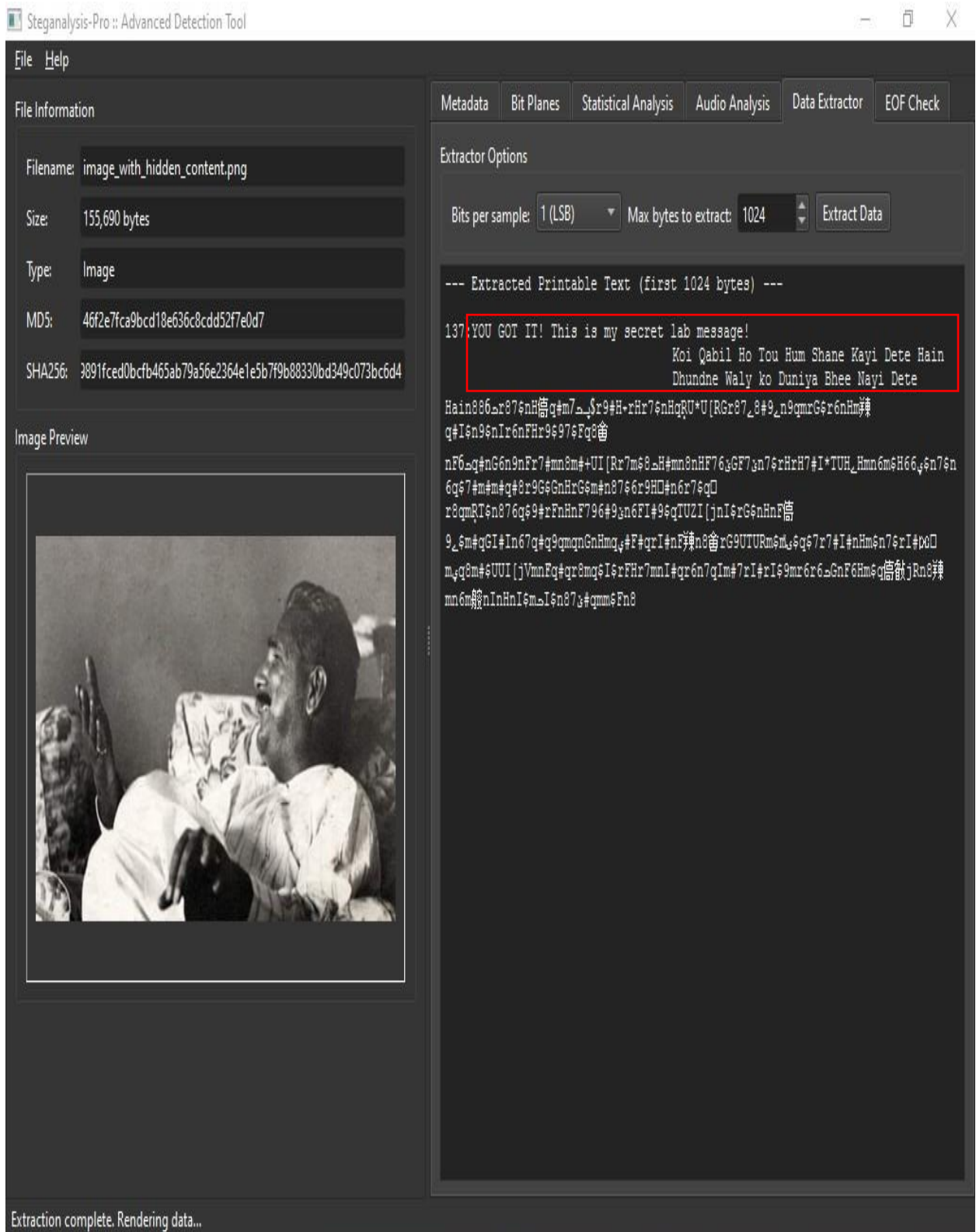
34 """
35
36 import sys
37 import os
38 import hashlib
39 import numpy as np
40 import exifread
41 from scipy.stats import chi2
42 from scipy.io import wavfile
43
44 from PIL import Image, ImageQt
45 from matplotlib.backends.backend_qtagg import FigureCanvasQTAgg as FigureCanvas
46 from matplotlib.figure import Figure
47 import matplotlib.pyplot as plt
48
49 # --- PySide6 Imports ---
50 # We import everything explicitly to make the single-file nature cleaner
51 from PySide6.QtCore import (
52     Qt, QSize, QRunnable, QThreadPool, Signal, Slot, QObject
53 )
54 from PySide6.QtGui import (
55     QAction, QIcon, QPixmap, QImage, QPalette, QColor
56 )
57 from PySide6.QtWidgets import (
58     QApplication, QMainWindow, QWidget, QVBoxLayout, QTabWidget,
59     QGridLayout, QLabel, QFileDialog, QTextEdit, QScrollArea,
60     QSplitter, QFrame, QGroupBox, QComboBox, QPushButton, QLineEdit,
61     QSpinBox, QProgressBar, QFormLayout, QGroupBox, QMessageBox
62 )
63
64 # --- Matplotlib Canvas Widget ---
65 class MplCanvas(FigureCanvas):
66     """A custom Matplotlib canvas widget to embed in PySide6."""
67     def __init__(self, parent=None, width=5, height=4, dpi=100):
68         self.fig = Figure(figsize=(width, height), dpi=dpi)
69         self.axes = self.fig.add_subplot(111)
70         super(MplCanvas, self).__init__(self.fig)
71         self.setParent(parent)
72         self.fig.tight_layout()
73
74 # --- Main Application Window ---
75 class MainWindow(QMainWindow):
76
77     def __init__(self):
78         super().__init__()
79
80         # --- Main Window Properties ---
81         self.setWindowTitle("Steg_Analyzer: Stego- & Advanced Detection Tool")
```

2. **File Loading:** The image file was loaded into the tool using its "Load Image" functionality.

3. **Analysis:** The tool's "Analyze" or "Detect" function was executed. This initiated the script to scan the image for hidden stenographic data.



4. **Results Review:** The tool's output (e.g., in a text box, status bar, or pop-up) was reviewed to determine if hidden content was found.



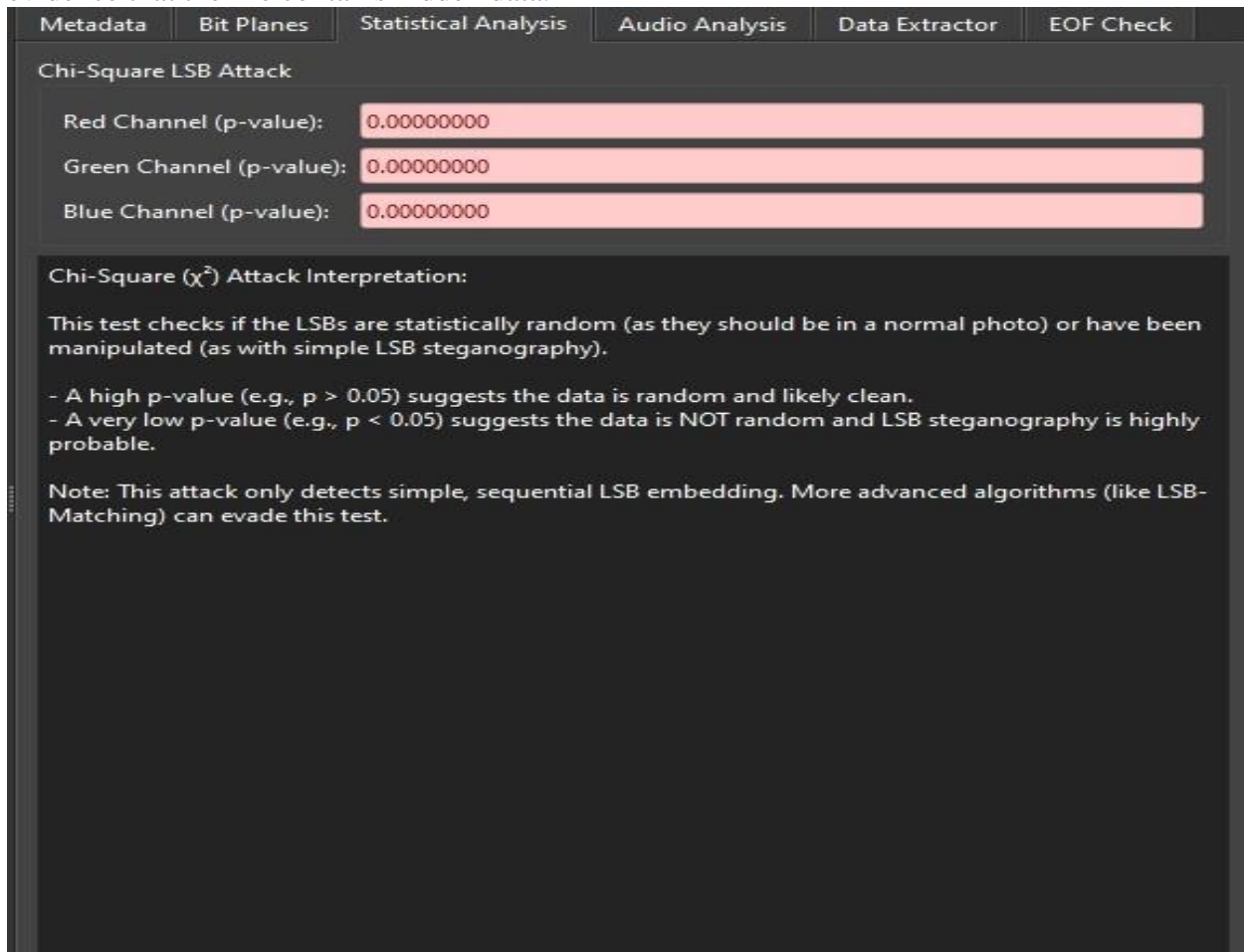
---

## 4. Results and Findings

The analysis of the sample image file using the custom tool yielded the following results:

- **Detection Status:** Hidden content was found and successfully extracted.
- **Hidden Content Type:** You Got It! This is my secret message.
- **Analysis Details:** The tool performed a Chi-Square ( $\chi^2$ ) attack on the image's Least Significant Bits (LSBs) to detect statistical anomalies. The p-value results strongly indicate data manipulation:
  - Red Channel (p-value): 0.00000000
  - Green Channel (p-value): 0.00000000
  - Blue Channel (p-value): 0.00000000

As per the test's interpretation, a p-value below 0.05 suggests the data's LSBs are not statistically random and that LSB steganography is "highly probable"<sup>5</sup>. These results provide clear forensic evidence that the file contains hidden data.



---

## 5. Conclusion

This experiment successfully demonstrated the application of a custom Python tool to detect and analyze steganography. The tool not only extracted a hidden text message ("You Got It! This is my secret message.") But also provided statistical proof of data manipulation.

The Chi-Square analysis confirmed that the image's LSBs were not random ( $p\text{-value} = 0.0$ ), a clear indicator of LSB steganography. This lab confirms the custom tool is effective for both extracting hidden content and forensically identifying the methods used to embed it.