**1.**

A Project Report On

**Folder Protector App**

Submitted in partial fulfillment of the requirement for the award of the degree

Bachelor of Computer Application (BCA)/ Bachelor of Science (IT)

Academic Year 2024 – 25

**SHREY PALAN 92200588062**

**PARAM KHANDAVI 92200588074**

**TRUSHA SALAVIYA 92200588137**

|  |
| --- |
| **Internal Guide** |
| RIDDHI JOSHI |



Rajkot-Morbi Road, At & PO : Gauridad, Rajkot 360 003. Gujarat. India.



**Faculty of Computer Applications (FCA)**



***This is to certify that the project work entitled***

***Folder Protector App***

***submitted in partial fulfillment of the requirement for the award of the degree of***

***Bachelor of Computer Application/ Bachelor of Science***

***(IT)***

***of the***

Marwadi University

***is a result of the bonafide work carried out by***

***SHREY PALAN (92200588062) PARAM KHANDAVI (92200588074) TRUSHA SAVALIYA(92200588137)***

***During the academic year 2024-25.***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **RIDDHI JOSHI** |  | **SUNIL BAJEJA** |  | **DR. SRIDARAN** |

**DECLARATION**

I/We hereby declare that this project work entitled **Folder Protector App**

is a record done by me.

I also declare that the matter embodied in this project is genuine work done by me and has not been submitted whether to this University or to any other University / Institute for the fulfillment of the requirement of any course of study.

Place: RAJKOT Date:

**PARAM KHANDAVI (92200588074) Signature:**

**TRUSHA SAVALIYA (92200588137) Signature:**

**SYREY PALAN (92200588062) Signature:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | | | | |  |  |
|  |  | **CONTENTS** | | | | |  |  |
|  | | | **Chapters** | **Particulars** | **Page No.** |  | |
| **1** | **SYNOPSIS** | 5 |
| **2**  2.1  2.2 | **PREAMBLE**  General Introduction Module description | 6-7 |
| **3**  3.1  3.2 | **TECHNICAL DESCRIPTION**  Hardware Requirement Software Requirement | 8 |
| **4**  4.1  4.2  4.3 | **SYSTEM DESIGN AND DEVELOPMENT**  Use Case-Diagram Activity Diagram Dataflow Diagram | 9-11 |
| **5** | **TESTING** | 12-19 |
| **6** | **CONCLUSION** | 20 |
| **7** | **LEARNING DURING PROJECT WORK** | 21 |
| **8** | **BIBLIOGRAPHY** | 22 |
| 4 | | | | | | | |
|  |  |  | | | | |  |  |

# SYNOPSIS

The provided Python code implements a simple graphical user interface (GUI) application using the Tkinter library to protect a folder with a password. The application allows users to select a folder, set a password for it, and lock or unlock the folder based on the provided password. Below is a detailed breakdown of the functionality:

**2.1 General Information**

The Folder Protection App is a simple graphical user interface (GUI) application built

Using Python's Tkinter library. Its primary purpose is to allow users to protect a specific folder on their computer by setting a password. When the folder is protected, it becomes hidden or access-restricted, and can only be unlocked by entering the correct password.

**Key Features:**

1. **Folder Selection**:

Users can easily browse and select any folder on their system that they wish to protect.

1. **Password Protection:**

The application enables users to set a password that must be entered to unlock the protected folder. This feature is essential for preventing

Unauthorized access.

1. **Folder Locking Mechanism:**

Once a password is set, the application employs system-specific commands to hide or restrict access to the selected folder. This makes the folder Invisible or inaccessible from standard file navigation.

1. **Unlocking Capability:**

Users can unlock the folder by entering the correct password, restoring access and visibility. This functionality is crucial for retrieving

Files and data stored within the protected folder.

1. **Cross-Platform Compatibility:**

The application is designed to work on multiple operating systems, including Windows and Linux/Mac, by using appropriate system commands for folder management.

**2.2 MODULE DESCRIPTION**

**Overview:**

The Folder Protection Application is a Python module that provides a graphical user interface (GUI) for users to protect folders on their local file system with a password. The application allows users to select a folder, set a password for it, and subsequently unlock the folder by entering the correct password. This module is built using the Tkinter library, which facilitates the creation of user-friendly desktop applications.

**Dependencies:**

1. Python:

The application is developed in Python and requires a compatible version

1. Tkinter:

This is the standard GUI toolkit for Python and is included with most Python installations.

1. OS and Platform Libraries:

These standard libraries are used for interacting with the operating system and managing file attributes.

**Key Components:**

Class: FolderProtectionApp

The main class that encapsulates all functionality related to the folder protection application.

Responsible for initializing the GUI, handling user interactions, and managing folder protection logic.

**Methods:**

**\_\_init\_\_(self, root):**

Initializes the main application window and sets up the initial state of the application, including attributes for the protected folder and password.

**create\_widgets(self):**

Creates and arranges the GUI elements, including labels, entry fields, and buttons for user interaction.

**browse\_folder(self):**

Opens a file dialog to allow users to select a folder and updates the GUI with the selected folder path.

**set\_password(self):**

Validates user input for the password, sets the password, and locks the selected folder by changing its attributes.

**lock\_folder(self):**

Implements the logic to hide or restrict access to the selected folder based on the operating system.

**unlock\_folder(self):**

Prompts the user for the password and unlocks the folder if the correct password is entered, restoring access to the folder.

**open\_folder(self, folder\_path):**

Opens the protected folder in the system's file explorer, allowing the user to view its contents.

**User Interface Elements:**

Labels, entry fields, and buttons are created to facilitate user interactions, such as selecting a folder, entering a password, and unlocking the folder.

# TECHNICAL DESCRIPTION

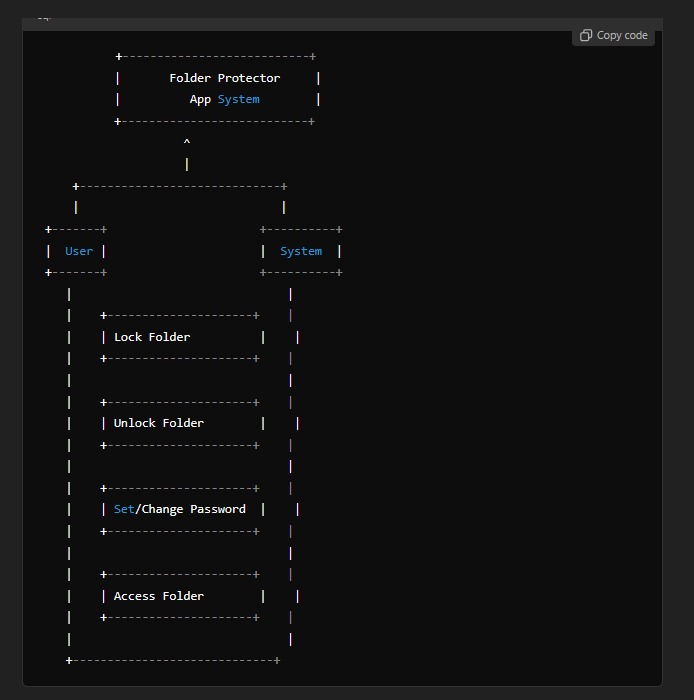
|  |  |
| --- | --- |
|  | |
|  |  |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
|  | |
|  |  |
|  |  |
|  |  |

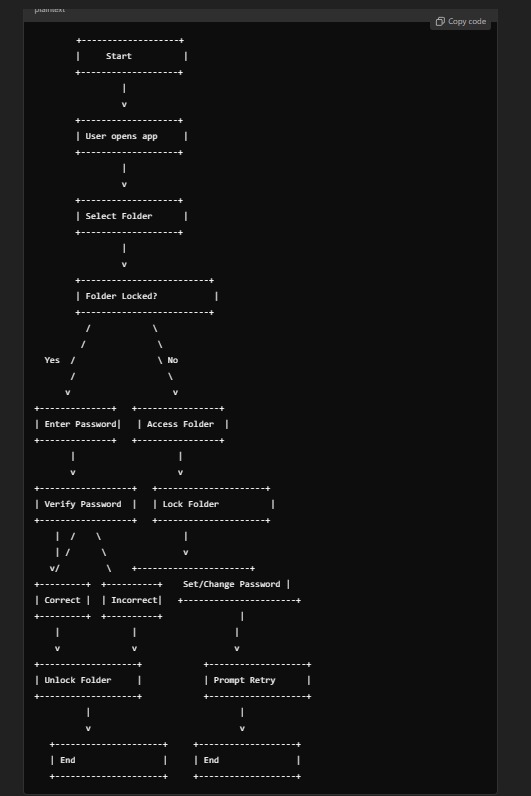
|  |  |
| --- | --- |
| HARDWARE REQUIREMENT | |
| HARD DISK | MINIMUM 80GB |
| RAM | MINIMUN 1GB |
| ACCESORIES | KEYBOARD, MOUSE AND MONITOR |
| GRAPHIC DESIGNING | INTEGRATED GRAPHICS OR DEDICATED GRAPHICS CARD WITH AT LEAST LUS512 MB VRAM |

|  |  |
| --- | --- |
| SOFTWARE REQUIREMENT | |
| OPERATING SYSTEM | WINDOWS 11 |
| CODE EDITOR | VISUAL STUDIO |
| DATABASE | MY SQL |

1. **SYSTEM DESIGN AND DEVELOPMENT**
   1. **USE-CASE DIAGRAM**



* 1. **ACTIVITY DIAGRAM**



* 1. **DATAFLOW DIAGRAM**



# TESTING

* 1. **SOURCE CODE**

import os

import tkinter as tk

from tkinter import messagebox, filedialog

import subprocess

import platform

class FolderProtectionApp:

def \_\_init\_\_(self, root):

self.root = root

self.root.title("Folder Protection App")

self.root.geometry("500x300")

self.protected\_folder = None

self.password = None

# Create GUI elements

self.create\_widgets()

def create\_widgets(self):

# Folder selection

self.folder\_label = tk.Label(self.root, text="Select Folder to Protect:")

self.folder\_label.pack(pady=10)

self.folder\_entry = tk.Entry(self.root, width=50)

self.folder\_entry.pack(pady=10)

self.browse\_button = tk.Button(self.root, text="Browse", command=self.browse\_folder)

self.browse\_button.pack(pady=5)

# Password setting

self.password\_label = tk.Label(self.root, text="Set Password to Protect Folder:")

self.password\_label.pack(pady=10)

self.password\_entry = tk.Entry(self.root, width=50, show="\*")

self.password\_entry.pack(pady=10)

self.set\_password\_button = tk.Button(self.root, text="Set Password", command=self.set\_password)

self.set\_password\_button.pack(pady=10)

# Unlock Folder

self.unlock\_button = tk.Button(self.root, text="Unlock Folder", command=self.unlock\_folder, state=tk.DISABLED)

self.unlock\_button.pack(pady=10)

def browse\_folder(self):

folder\_selected = filedialog.askdirectory()

if folder\_selected:

self.folder\_entry.delete(0, tk.END)

self.folder\_entry.insert(0, folder\_selected)

self.protected\_folder = folder\_selected

def set\_password(self):

if not self.protected\_folder:

messagebox.showerror("Error", "Please select a folder first!")

return

password = self.password\_entry.get()

if not password:

messagebox.showerror("Error", "Password cannot be empty!")

return

self.password = password

self.password\_entry.config(state=tk.DISABLED)

self.set\_password\_button.config(state=tk.DISABLED)

self.unlock\_button.config(state=tk.NORMAL)

# Lock the folder (hide it or restrict access)

self.lock\_folder()

messagebox.showinfo("Success", "Password set! Folder is now protected.")

def lock\_folder(self):

if os.name == 'nt': # Windows

os.system(f'attrib +h +s "{self.protected\_folder}"')

elif os.name == 'posix': # Linux/Mac

os.system(f'chmod 000 "{self.protected\_folder}"')

else:

messagebox.showerror("Error", "Unsupported OS")

def unlock\_folder(self):

if not self.protected\_folder:

messagebox.showerror("Error", "No folder selected!")

return

password\_prompt = tk.simpledialog.askstring("Unlock Folder", "Enter password to unlock:", show="\*")

if password\_prompt == self.password:

# Unlock the folder (make it visible or restore access)

if os.name == 'nt': # Windows

os.system(f'attrib -h -s "{self.protected\_folder}"')

elif os.name == 'posix': # Linux/Mac

os.system(f'chmod 755 "{self.protected\_folder}"')

messagebox.showinfo("Success", "Password correct! Folder unlocked.")

self.open\_folder(self.protected\_folder)

else:

messagebox.showerror("Error", "Incorrect password!")

def open\_folder(self, folder\_path):

# Open the folder in the file explorer

if platform.system() == "Windows":

os.startfile(folder\_path)

elif platform.system() == "Darwin": # macOS

subprocess.call(["open", folder\_path])

elif platform.system() == "Linux":

subprocess.call(["xdg-open", folder\_path])

else:

messagebox.showerror("Error", "Unsupported OS")

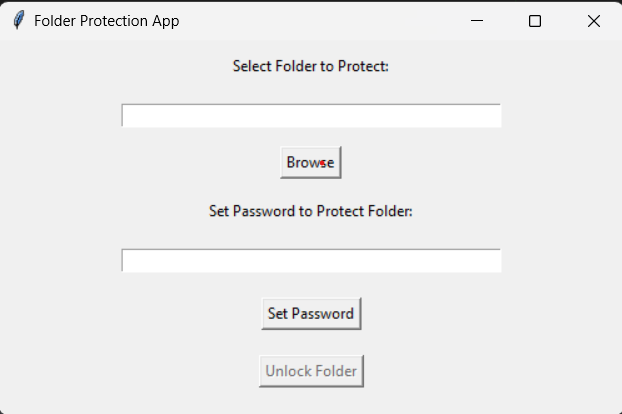
if \_\_name\_\_ == "\_\_main\_\_":

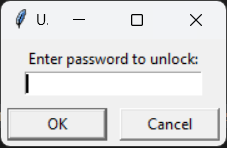
root = tk.Tk()

app = FolderProtectionApp(root)

root.mainloop()

**5.2 OUTPUT**





# CONCLUSION

The Folder Protection App is a basic implementation of folder protection

using password security. While it provides a simple interface and

basic functionality, it is essential to consider its limitations, especially

regarding security and platform compatibility. For more robust folder

protection, more advanced techniques and encryption methods would be

necessary.

# BIBLIOGRAPHY

* 1. **ONLINE REFERENCE**
  2. **OFFLINE REFERENCE**
     + **Param Khandvi**
     + **Shrey Palan**