PROJECT REPORT: FILE INTEGRITY CHECKER
Project Title: File Integrity Checker using Python

OBJECTIVE:

The aim of this project is to develop a tool that monitors changes in files by calculating and comparing their hash values. This ensures that any unauthorized modification, deletion, or corruption of files is quickly detected, helping maintain file security and integrity.

PROBLEM STATEMENT:

In today's digital environment, files are vulnerable to unintentional changes, malicious tampering, or corruption. It is difficult to manually track if files have been altered or deleted. There is a need for an automated tool that can monitor files and notify users of any changes in a reliable way.

PROPOSED SOLUTION:

We have developed a **File Integrity Checker** in Python that works in two simple steps:

1. **Hash Generation:** The tool scans a selected folder and calculates a unique hash value for every file. These hash values are stored securely in a JSON file.
2. **Integrity Verification:** At any later time, the user can run the tool again to compare current file hashes with the previously stored ones. If any file is modified, deleted, or replaced, the tool will detect it and show a warning.

This tool also provides a simple GUI and stores a detailed log of all checks for future reference.

TECHNOLOGIES USED:

Technology	Purpose
Python	Core programming language
hashlib	To create hash values (SHA256)
os	For reading files and directories
json	To save and read hash data
tkinter	To create the GUI
datetime	To log date and time of integrity checks

HOW IT WORKS:

Step 1: Generate Hashes

- User selects a folder through the GUI.
- The program reads all files in the folder.
- A **SHA-256 hash** is generated for each file.
- These hashes are stored in a file named hashes.json.

Step 2: Check Integrity

- The tool reads the stored hashes.
- It recalculates the current hashes of the files.
- If the hash matches, the file is safe.
- If not, the file is either changed or missing.
- Results are displayed to the user and saved in log.txt.

TEST CASE EXAMPLE:

Suppose you have three files in a folder:
project.docx
summary.pdf
data.csv
On Day 1, you run the tool and it stores the hash of all 3 files.
On Day 3:

- project.docx is modified.
- summary.pdf is deleted.
- data.csv is unchanged.

You run the checker again and see:

```
[!] File changed: project.docx
[!] File missing: summary.pdf
[OK] File intact: data.csv
```

BENEFITS OF THE PROJECT:

- Easy to use with GUI
- Detects file tampering and loss
- Saves logs for auditing
- Useful for students, professionals, and cybersecurity learners

REAL-WORLD APPLICATIONS:

- Securing legal documents
- Monitoring software project files
- Protecting exam or confidential data
- Cybersecurity and digital forensics

FUTURE ENHANCEMENTS:

- Email alert when a file is changed
- Filter to include/exclude file types (e.g., only .txt or .pdf)
- Cloud backup of hashes.json and log.txt
- Automatic scheduled checks using Task Scheduler or cron jobs

CONCLUSION:

This File Integrity Checker is a simple yet powerful tool built using Python. It ensures that important files remain unchanged unless authorized. With a clear interface and smart functionality, it can be used across various fields to protect digital content.

README.md

# File Integrity Checker

A simple desktop tool built in Python to monitor and detect unauthorized changes, deletions, or tampering in your files using hash values.

C FOLDER STRUCTURE

Organize your files like this:

C:\FileIntegrityChecker\

```
├── dist\
│   └── file_integrity_gui.exe    <-- Your compiled EXE (via PyInstaller)
├── hashes.json                   <-- Auto-generated when running
├── log.txt                       <-- Auto-generated when checking
└── file_integrity_gui.py         <-- Your Python script
```

FEATURES

- Detects if any file is modified, deleted, or replaced
- Uses SHA-256 hashing for strong integrity checking
- Provides a user-friendly GUI (no coding knowledge required)
- Saves results in a `log.txt` file with timestamps
- Standalone `.exe` version available – no need for Python installation

TECHNOLOGIES USED

- `Python`
- `hashlib` – file hashing
- `tkinter` – GUI
- `json` – saving hash data
- `os`, `datetime` – file and time handling
- `pyinstaller` – to create .exe
- `Inno Setup` – to create installer (optional)

HOW TO RUN

Option 1: Run from Python (for developers)

1. Install Python (https://python.org)
2. Open **Command Prompt** and navigate to the project folder:

   cd C:\FileIntegrityChecker
3. Run the script:

   python file_integrity_gui.py

Make sure you have tkinter installed (usually included by default in standard Python installations).

Option 2: Run the .exe File (No Python Needed)

1. Go to the dist/ folder.
2. Double-click on *file_integrity_gui.exe.*

It will open the GUI instantly, like any Windows app.

Option 3: Run via CMD using .exe

1. Open **Command Prompt**.
2. Navigate to the dist/ folder:
   cd C:\FileIntegrityChecker\dist
3. Run:
   file_integrity_gui.exe

AUTO-GENERATED FILES

- *hashes.json*: Stores hashes of files from the selected folder.
- *log.txt:* Stores logs of integrity check results.

You don't need to create them manually — they are created when the tool is run.

NOTES

- Only file contents are checked (not file names).
- Safe for any file type: .txt, .pdf, .exe, .docx, etc.
- Does not modify or delete any file — only reads and reports.

DEVELOPER INFO:

Name: **Ashirwad Shukla**
Field: Cybersecurity & Software Development
Tool Type: Desktop GUI Application