1. **Abstract:**

Duplicate bills can cause financial discrepancies and increase the risk of fraud. This project aims to develop an automated system that detects duplicate invoices using file hashing techniques. By implementing SHA-256 hashing, the system ensures that no bill is uploaded more than once. The system is designed with a user-friendly frontend and a Python Flask-based backend to handle file uploads and comparison efficiently. The solution is scalable and can be integrated with databases for long-term storage.

1. **Introduction**

In many organizations, duplicate invoice submissions lead to financial losses and inefficiencies. Manual checking of invoices is time-consuming and prone to human error. The objective of this project is to automate duplicate bill detection by comparing uploaded invoices using hash-based verification. The system enhances transparency, minimizes errors, and ensures financial accuracy.

1. **Problem Statement**

The problem of duplicate invoices arises due to unintentional resubmissions, fraud, or clerical errors. Traditional manual verification is inefficient and lacks reliability. A solution is required to automate this process and alert users when a duplicate bill is detected.

1. **Objective & Scope**

**Objective:**

* Detect duplicate invoices automatically
* Ensure error-free financial transactions
* Provide a seamless and efficient user interface

**Scope:**

* Applicable for businesses, hospitals, e-commerce platforms, and government sectors
* Can be integrated with databases and cloud storage
* Future enhancements may include AI-based fraud detection

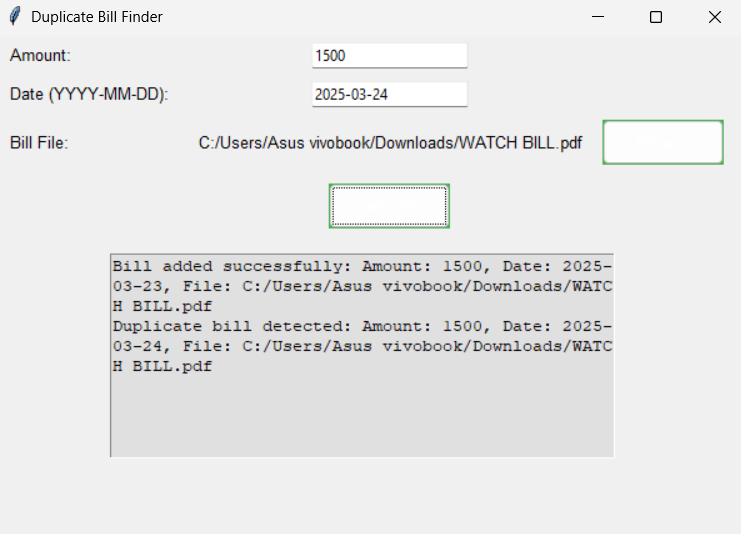
1. **Methodology (How It Works)**

* The user selects an invoice file and enters the amount and date.
* The system extracts file content and computes a SHA-256 hash.
* The hash is compared with existing hashes in the database.
* If a match is found, the system flags the bill as a duplicate.
* If unique, the bill is stored securely in the system.

1. **Tools & Technologies Used**

* **Frontend:** HTML, CSS, JavaScript
* **Backend:** Python (Flask)

1. **Results & Output**

****

1. **Challenges Faced & Future Scope**

**Challenges:**

* **Ensuring efficient file processing for large datasets**
* **Handling different file formats (PDF, PNG, JPEG)**
* **Optimizing real-time duplicate detection**

**Future Scope:**

* **Integration with databases for permanent storage**
* **AI-based anomaly detection for invoice fraud**
* **Mobile app development for on-the-go invoice uploads**

1. **Conclusion**

The Duplicate Bill Detection System effectively prevents duplicate invoice submissions by using hash-based verification. The project ensures accurate financial transactions, reduces human error, and increases efficiency. Future improvements can make the system even more robust with database integration and AI-driven fraud detection.

1. **References**

* gemini.google.com
* https://chatgpt.com/