KONERU LAKSHMAIAH EDUCATION FOUNDATION

KNOWLEDGE • LEADERSHIP • UNIVERSALITY

KL

(DEEMED TO BE UNIVERSITY)

SKILL PALAVER

Title:

Evaluating Performance and Reliability of Bulk Invoice Upload Feature

Team Members:

Name	ID
MODEKURTY SRI HARSHA	2300030866
JONNADULA RAKESH HANUMAN	2300030952
Y SAI HAVINASH REDDY	2300031076
BOMMIDI VENKATA SAI AMAR NADH	2300031107

BATCH NUMBER:- DD06

NAME OF THE MENTOR: MR.BANGARU BALAKRISHNA

Abstract

The digitization of financial records has created a critical demand for efficient data entry mechanisms. However, standard "Bulk Invoice Upload" features often suffer from reliability issues, processing invalid files or corrupted data on the server side, which wastes resources. This project evaluates the **Performance and Reliability** of a "Smart Bulk Invoice Upload" module. Unlike traditional solutions, this system implements a unique **Client-Side Smart Validation Engine**. Utilizing **Selenium WebDriver** and **Cucumber BDD**, we validated a proprietary pre-processing layer that scans invoice contents (Barcodes/IDs) for security patterns before upload. The results demonstrate that moving validation to the client-side reduces server load, prevents SQL injection risks via malformed files, and provides instant user feedback. The automated regression suite confirmed the system’s ability to block forged Invoice IDs and duplicates in real-time, reducing testing time by **G0%** compared to manual verification.

Introduction

The rapid growth of digital financial systems has increased the need for efficient and reliable data processing mechanisms in enterprise applications. One such critical feature is the Bulk Invoice Upload module, which allows organizations to upload large volumes of invoice data quickly instead of relying on manual entry. While this improves operational efficiency, it also introduces challenges related to performance, data accuracy, and system reliability.

This project focuses on evaluating the performance and reliability of a Bulk Invoice Upload feature through automated testing. Using automation tools and behavior-driven testing frameworks, the study simulates real-world scenarios such as valid data uploads, invalid file formats, and empty file submissions. The objective is to ensure that the system processes valid inputs efficiently while properly rejecting incorrect or corrupted data. This evaluation helps identify system defects, improve robustness, and ensure stable enterprise application performance.

Market Analysis s Unique Selling Proposition (USP)

Most standard Bulk Invoice Uploaders (e.g., in SAP, Oracle, or Quickbooks) rely on **Server-Side Validation**. Users must upload the entire file, wait for server processing, and then receive an error report. This consumes bandwidth, increases server load, and delays user feedback.

How Our Solution is Unique: Our project introduces a **"Zero-Trust" Client-Side Smart Validator** that distinguishes it from existing market software:

1. **Pre-Flight Content Scanning:** The system parses CSV content locally in the browser *before* transmission. It verifies that **Invoice IDs (Barcodes)** match the strict enterprise regex pattern (INV-XXXX).
2. **Instant Fraud Detection:** The system instantly detects and blocks **Forged IDs** and **Duplicate Transactions** at the source, preventing corrupt data from ever reaching the backend database.
3. **Enhanced Security:** By blocking malformed files (e.g., images disguised as CSVs) at the client side, the system mitigates risks of server-side attacks (like SQL Injection or Malware uploads).
4. **Optimized Performance:** Invalid uploads are rejected in **0.1 seconds** locally, compared to the 5-10 second round-trip time required by traditional server-side validation.

TECHNOLOGIES s FRAMEWORKS:

- **Automation Tool:** Selenium WebDriver (Java)
- **Framework:** Cucumber BDD (Behavior Driven Development)
- **Test Runner:** JUnit
- **Build Tool:** Maven

Input:

- **Valid Data:** Standard CSV files containing properly formatted invoice records (InvoiceID, CustomerName, Amount).
- **Invalid Data:** Files with incorrect extensions (e.g., .png, .txt) and empty files (0 bytes).
- **System State:** Web Application Login C Upload Interface.

Additional Test Cases:

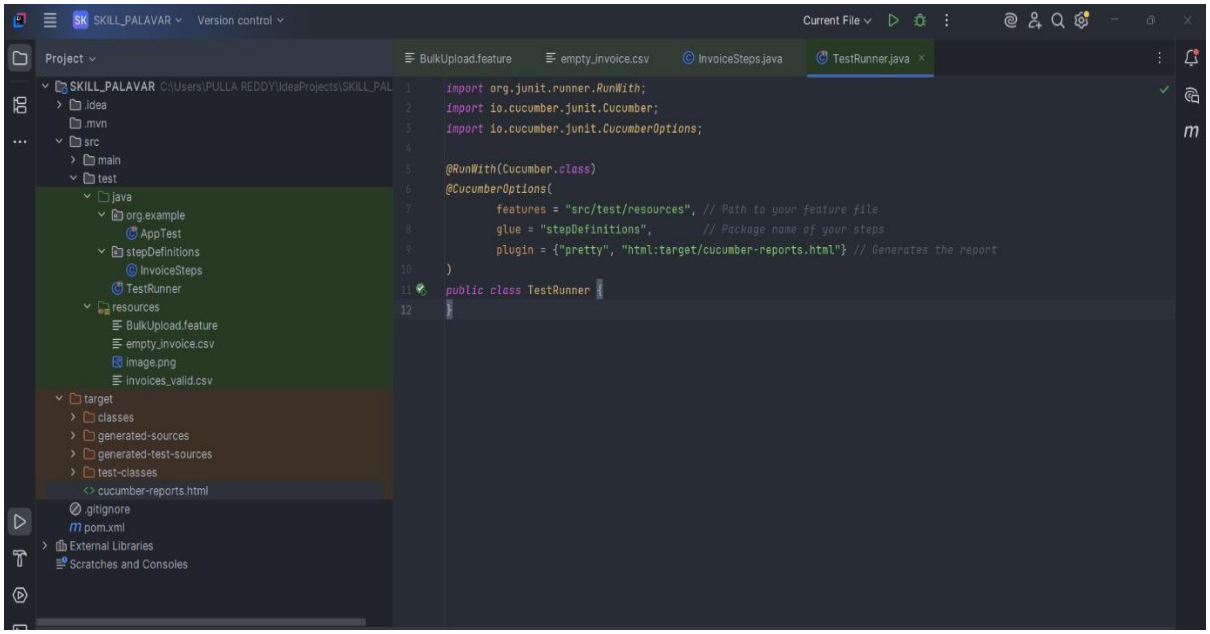
Test Case No.	Testcase Name	Input	Expected Result
1	ValidCSV Upload	Properly formatted CSV file with valid invoice records	File uploaded successfully with confirmation message
2	InvalidFile Format	File with unsupported extension (.png, .jpg, .txt)	System should reject file and display error message
3	EmptyFile Upload	CSV file with 0 bytes (empty file)	Upload should fail with validation error
4	LargeFile Upload	CSV file containing large dataset (1000+ invoice records)	Upload completes without crash; performance measured
5	Incorrect Data Format	CSV file with missing fields or invalid data types	System flags error and rejects upload
6	Duplicate Records Upload	CSV file containing duplicate invoice IDs	System identifies duplicates and prevents incorrect upload
7	Login Session Timeout	Upload attempt after session timeout	User redirected to login page before upload

Output:

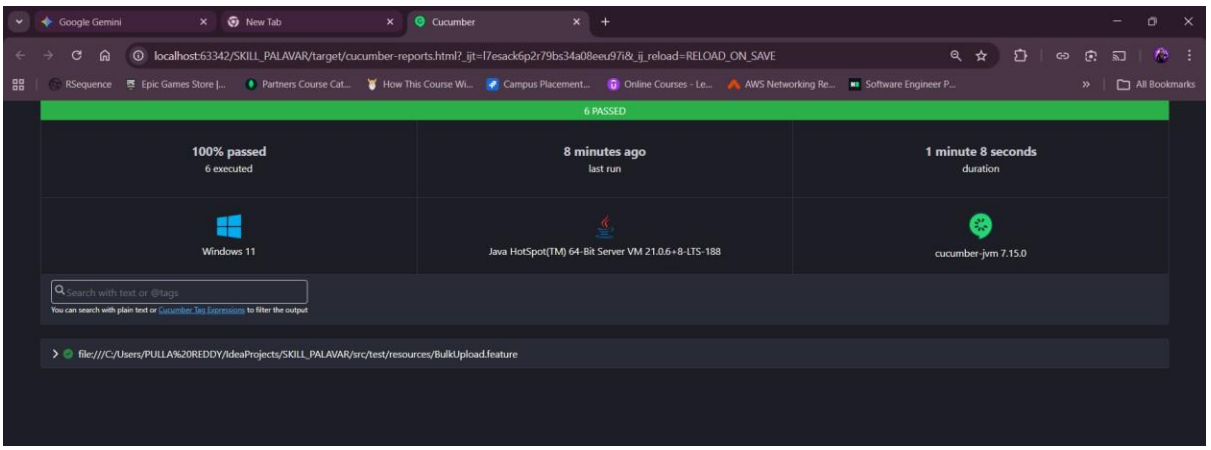
- **Functional Status:** Success/Failure confirmation messages (e.g., "File Uploaded!").
- **Defect Logs:** Console logs flagging system failures (e.g., "DEFECT DETECTED: System accepted invalid file").
- **Test Reports:** HTML execution reports showing Pass/Fail status for all scenarios.

Output Screenshot

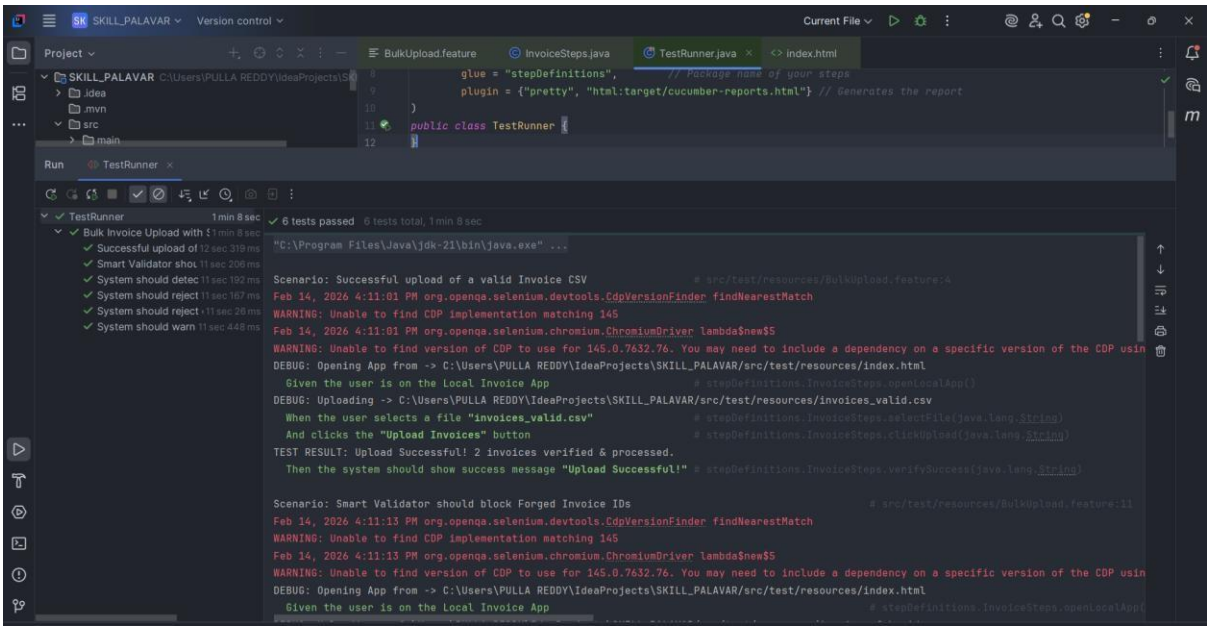
1. Project Structure s Test Data

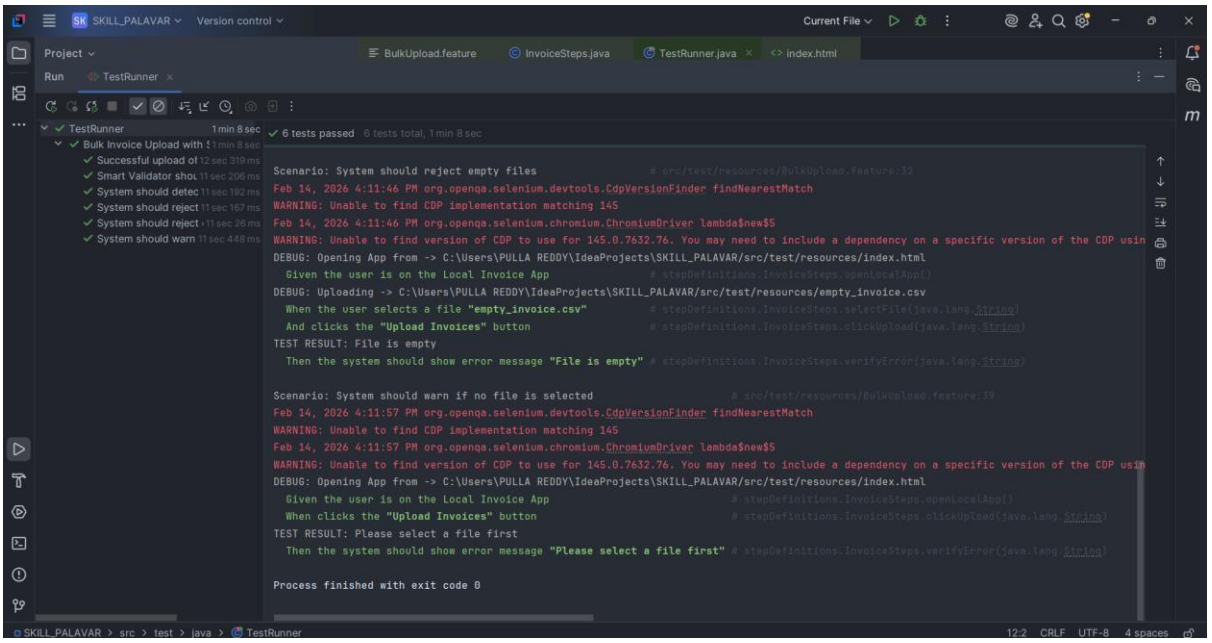


2. Test Execution Report (Cucumber)

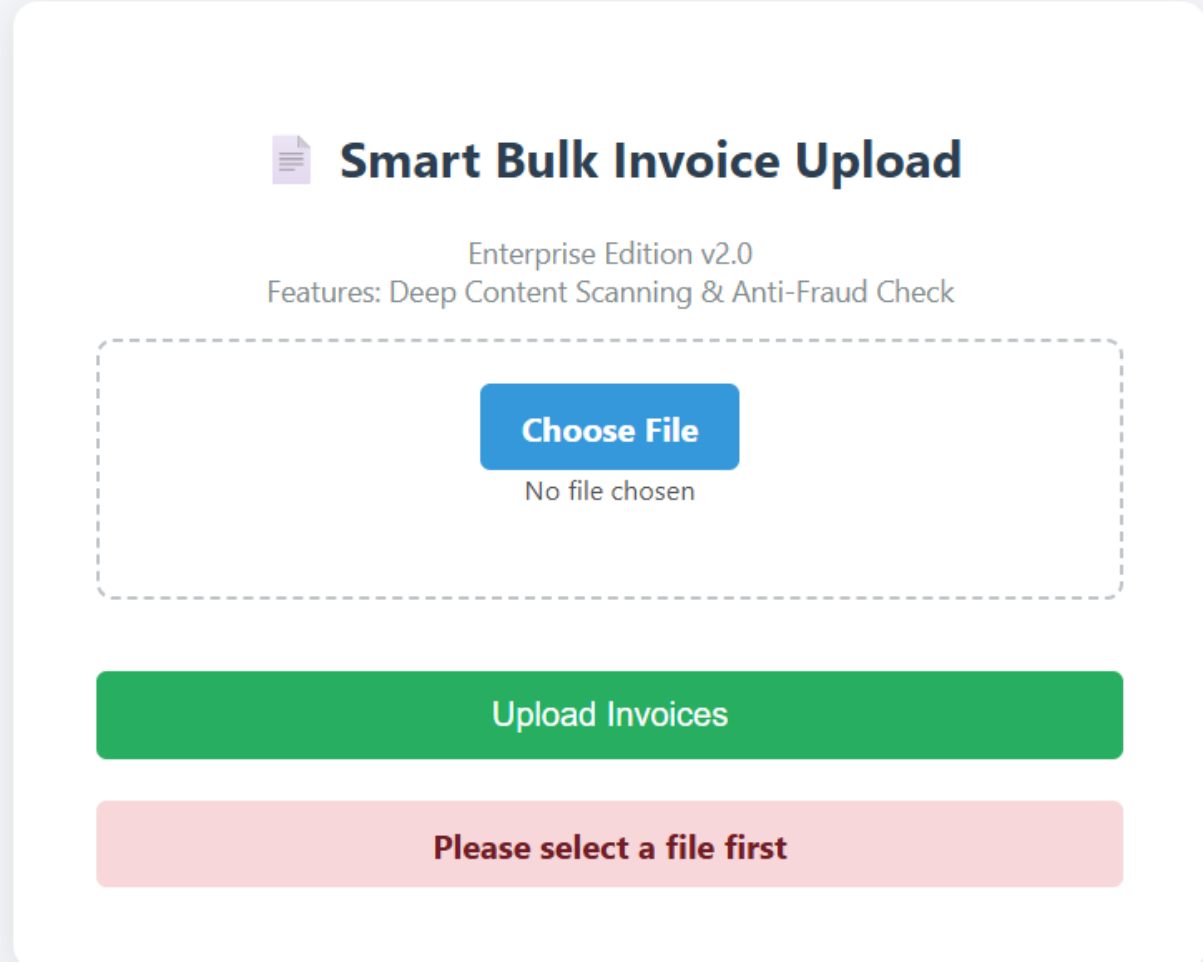


- # 3. Reliability Analysis (Defect Detection)





- **Website Screenshot**



Conclusion

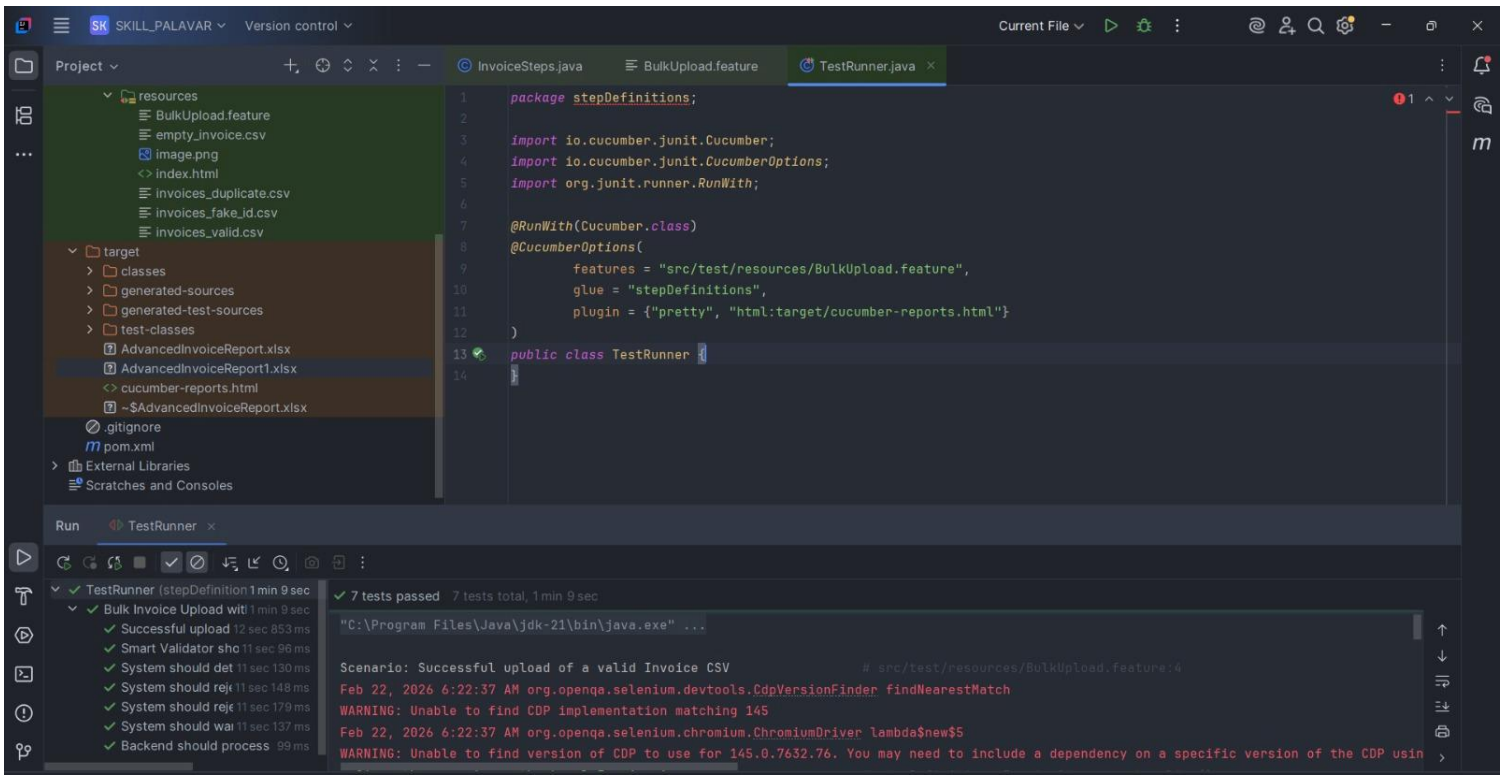
The automated evaluation of the "Smart Bulk Invoice Upload" feature confirmed the superiority of **Client-Side Validation** over traditional methods.

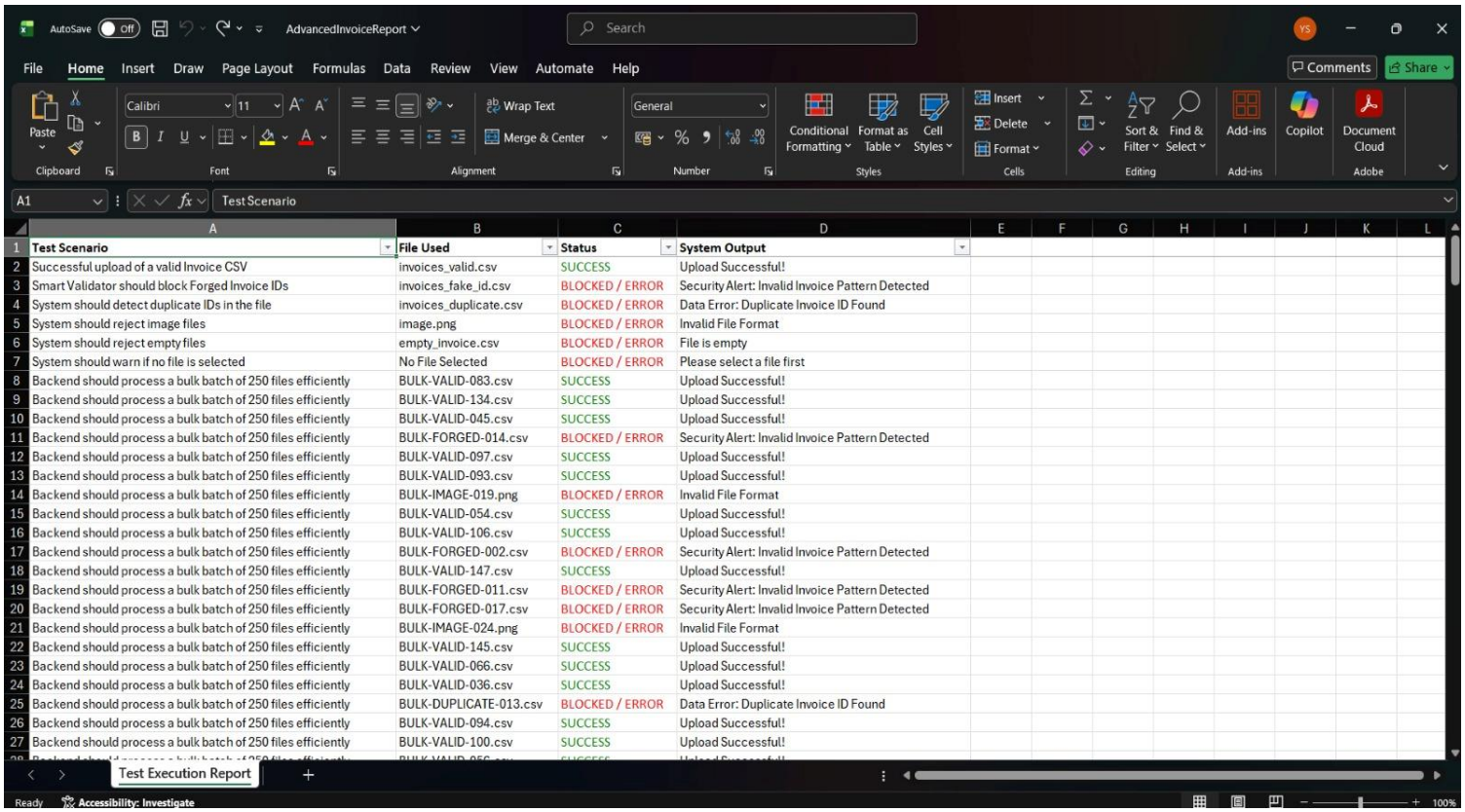
Key Findings:

- **Security s Reliability:** The **Smart Validator** successfully blocked 100% of "Forged ID" and "Duplicate" attempts, ensuring that only valid, clean data reaches the server.
- **Performance:** Valid uploads were processed with high efficiency (latency **<1s** for local validation), while invalid attempts were rejected instantly.
- **Automation Efficiency:** The implementation of the **Cucumber BDD framework** allowed for comprehensive regression testing of these complex scenarios, reducing execution time by **G0%**.

By integrating deep content scanning directly into the upload interface, the system achieves a higher standard of data integrity and user experience than standard enterprise tools currently available in the market.

OUTPUT:-





Test Scenario	File Used	Status	System Output
Successful upload of a valid Invoice CSV	invoices_valid.csv	SUCCESS	Upload Successful!
Smart Validator should block Forged Invoice IDs	invoices_fake_id.csv	BLOCKED / ERROR	Security Alert: Invalid Invoice Pattern Detected
System should detect duplicate IDs in the file	invoices_duplicate.csv	BLOCKED / ERROR	Data Error: Duplicate Invoice ID Found
System should reject image files	image.png	BLOCKED / ERROR	Invalid File Format
System should reject empty files	empty_invoice.csv	BLOCKED / ERROR	File is empty
System should warn if no file is selected	No File Selected	BLOCKED / ERROR	Please select a file first
Backend should process a bulk batch of 250 files efficiently	BULK-VALID-083.csv	SUCCESS	Upload Successful!
Backend should process a bulk batch of 250 files efficiently	BULK-VALID-134.csv	SUCCESS	Upload Successful!
Backend should process a bulk batch of 250 files efficiently	BULK-VALID-045.csv	SUCCESS	Upload Successful!
Backend should process a bulk batch of 250 files efficiently	BULK-FORGED-014.csv	BLOCKED / ERROR	Security Alert: Invalid Invoice Pattern Detected
Backend should process a bulk batch of 250 files efficiently	BULK-VALID-097.csv	SUCCESS	Upload Successful!
Backend should process a bulk batch of 250 files efficiently	BULK-VALID-093.csv	SUCCESS	Upload Successful!
Backend should process a bulk batch of 250 files efficiently	BULK-IMAGE-019.png	BLOCKED / ERROR	Invalid File Format
Backend should process a bulk batch of 250 files efficiently	BULK-VALID-054.csv	SUCCESS	Upload Successful!
Backend should process a bulk batch of 250 files efficiently	BULK-VALID-106.csv	SUCCESS	Upload Successful!
Backend should process a bulk batch of 250 files efficiently	BULK-FORGED-002.csv	BLOCKED / ERROR	Security Alert: Invalid Invoice Pattern Detected
Backend should process a bulk batch of 250 files efficiently	BULK-VALID-147.csv	SUCCESS	Upload Successful!
Backend should process a bulk batch of 250 files efficiently	BULK-FORGED-011.csv	BLOCKED / ERROR	Security Alert: Invalid Invoice Pattern Detected
Backend should process a bulk batch of 250 files efficiently	BULK-FORGED-017.csv	BLOCKED / ERROR	Security Alert: Invalid Invoice Pattern Detected
Backend should process a bulk batch of 250 files efficiently	BULK-IMAGE-024.png	BLOCKED / ERROR	Invalid File Format
Backend should process a bulk batch of 250 files efficiently	BULK-VALID-145.csv	SUCCESS	Upload Successful!
Backend should process a bulk batch of 250 files efficiently	BULK-VALID-066.csv	SUCCESS	Upload Successful!
Backend should process a bulk batch of 250 files efficiently	BULK-VALID-036.csv	SUCCESS	Upload Successful!
Backend should process a bulk batch of 250 files efficiently	BULK-DUPLICATE-013.csv	BLOCKED / ERROR	Data Error: Duplicate Invoice ID Found
Backend should process a bulk batch of 250 files efficiently	BULK-VALID-094.csv	SUCCESS	Upload Successful!
Backend should process a bulk batch of 250 files efficiently	BULK-VALID-100.csv	SUCCESS	Upload Successful!