A Project Report On

SMART INVOICE ANALYZER

Submitted in partial fulfilment of the requirement for the award of the Degree in

Bachelor of Technology

in

Computer Science and Engineering

Submitted By

SHAIK ASIFA	**(21031A0548)**
NELLURI KRANTI VARDHAN	**(21031A0536)**
DWARAPUREDDI LALITHA	**(21031A0514)**
V V S S MOHAN RAMANAM	**(22035A0507)**
MAILAMURI FATHIMA	**(22035A0504)**

Under the esteemed Guidance of

Dr.G.MADHAVI

ASSISTANT PROFESSOR & HOD(i/c)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

UNIVERSITY COLLEGE OF ENGINEERING NARASARAOPET

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY KAKINADA

NARASARAOPET – 522601, ANDHRA PRADESH, INDIA

2024-2025

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

UNIVERSITY COLLEGE OF ENGINEERING NARASARAOPET

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY KAKINADA

NARASARAOPET – 522601, ANDHRA PRADESH, INDIA

2024-2025

CERTIFICATE

This is to certify that the thesis entitled **"SMART INVOICE ANALYZER"** that is being submitted by **SHAIK ASIFA (21031A0548), NELLURI KRANTI VARDHAN (21031A0536), DWARAPUREDDI LALITHA (21031A0514), V V S S MOHAN RAMANAM (22035A0507), MAILAMURI FATHIMA (22035A0504)** in partial fulfilment for the award of Bachelor of Technology in Computer Science and Engineering to the University College of Engineering Narasaraopet, Jawaharlal Nehru Technological University Kakinada is a record of bonafide work carried out by them under my guidance and supervision.

The results embedded in this thesis have not been submitted to any other university/Institute for the award of any degree / diploma.

PROJECT SUPERVISOR

Dr. G. MADHAVI
Assistant Professor and HOD(i/c)
Department of CSE
UCEN-JNTUK

HEAD OF THE DEPARTMENT

Dr. G. MADHAVI
Assistant Professor and HOD(i/c)
Department of CSE
UCEN-JNTUK

EXTERNAL EXAMINER

DECLARATION

We here by declare that the work described in the project report entitled **"SMART INVOICE ANALYZER"** which is submitted by us in partial fulfillment of the requirements for the award of Bachelor of Technology in the department of **COMPUTER SCIENCE AND ENGINEERING** to the University College of Engineering Narasaraopet, Jawaharlal Nehru Technological University Kakinada, Andhra Pradesh, is the result of work done by us under the guidance of **Dr. G. MADHAVI**, Assistant Professor and HOD(i/c). The work is original and has not been submitted for any Degree/Diploma of this or any other university.

Name of the Student	Roll No.	Signature
SHAIK ASIFA	**21031A0548**	
NELLURI KRANTI VARDHAN	**21031A0536**	
DWARAPUREDDI LALITHA	**21031A0514**	
V V S S MOHAN RAMANAM	**22035A0507**	
MAILAMURI FATHIMA	**22035A0504**	

Place:

Date:

ACKNOWLEDGEMENT

It is needed with a great sense of pleasure and immense sense of gratitude that we acknowledge the help of these individuals. We owe many thanks to many people who helped and supported us during the entire course of this project. We take privilege to express our heartfelt gratitude to our project supervisor and our Head of the Department **Dr. G. MADHAVI,** Assistant Professor & HOD (i/c) and our Project Coordinator **Mrs. B. JYOTHI,** Assistant Professor (c) of CSE Department, for their valuable suggestions and constant motivation that greatly helped us in successful completion of the project.

We express our sincere thanks to our beloved Vice Principal **Dr. Y. S. Kishore Babu,** UCEN JNTUK and we extend our thanks to Principal **Prof. CH. Srinivasa Rao**, UCEN JNTUK for providing us with a good infrastructure and environment to carry out this project. We are thankful to all faculty members for extending their kind cooperation and assistance. Finally, we are extremely thankful to our parents and friends for their constant help and moral support.

Project Associates

Shaik Asifa	**21031A0548**
Nelluri Kranti Vardhan	**21031A0536**
Dwarapureddi Lalitha	**21031A0514**
V V S S Mohan Ramanam	**22035A0507**
Mailamuri Fathima	**22035A0504**

TABLE OF CONTENTS

ABSTRACT

The rapid expansion of digital transactions has heightened the need for automated invoice processing solutions. Traditional methods rely on manual data entry and rule-based systems, making them inefficient for handling large volumes of invoices, leading to errors, delays, and increased operational costs. The **SMART INVOICE ANALYZER** is an AI- powered system that overcomes these limitations by automating invoice extraction, translation, analysis, and categorization. Utilizing OCR and the Gemini API, it extracts essential invoice details such as Invoice Number, Invoice Date, Product Name, Company, Quantity, Unit Price, Total Price, Customer Name, and Fraud Indicators from invoices in image or PDF formats. Unlike the existing system, which lacks intelligent fraud detection, the proposed system enhances security by identifying fraudulent invoices in English, though results for other languages may be limited. It also introduces an interactive dashboard providing real-time graphical insights and analytical trends, allowing businesses to monitor transactions effectively. Additionally, the system includes an invoice generation module using ReportLab, ensuring structured and professional formatting. A core feature is product demand and frequency analysis, enabling businesses to track purchasing patterns using a dataset of 10,000 invoices covering electronic gadgets like laptops, TVs, ACs, printers, mobiles, refrigerators, washing machines, and tablets. This AI-driven solution surpasses traditional approaches by integrating automation, multi-language support, fraud detection, and data analytics, thereby enhancing operational efficiency, reducing manual workload, and empowering businesses with actionable financial insights..

LIST OF FIGURES

LIST OF TABELS

LIST OF ABBREVIATIONS

S.NO	ABBREVIATION	EXPANSION
1	**OCR**	Optical Character Recognition
2	**PDF**	Portable Document Format
3	**CSV**	Comma-Separated Values
4	**MSE**	Mean Squared Error

CHAPTER1
INTRODUCTION

1. INTRODUCTION

1.1 Introduction to SMART INVOICE ANALYZER:

The Smart Invoice Analyzer is an advanced AI-driven solution designed to revolutionize invoice processing by automating data extraction, translation, fraud detection, analysis, and generation. With businesses handling invoices in multiple languages and formats, traditional processing methods often lead to delays, inefficiencies, and errors. This system leverages OCR and the Gemini API to accurately extract key invoice details from PDF and image formats, ensuring multi-language support for seamless cross-border transactions. It incorporates fraud detection mechanisms to identify suspicious invoices, while an interactive dashboard provides real-time visual insights, product demand analysis, and categorization based on a custom dataset of 10,000 invoices covering various electronic gadgets. Additionally, the system enables businesses to generate structured invoices using ReportLab, ensuring professional formatting and consistency. By integrating AI-powered automation, multi-language capabilities, and intelligent analytics, the Smart Invoice Analyzer Tool enhances financial accuracy, minimizes manual workload, improves decision-making, and optimizes business operations.

1.2 Significance of the project:

Smart Invoice Analyzer is a significant innovation in automated invoice processing, addressing key challenges businesses face in handling invoices across multiple languages and formats. Traditional invoice management systems struggle with inconsistencies in data extraction, leading to inefficiencies and errors in financial operations. By leveraging OCR and the Gemini API, this tool ensures accurate extraction of essential invoice details, allowing businesses to process invoices seamlessly regardless of their language. Additionally, the tool integrates fraud detection mechanisms, analyzing invoice patterns to identify anomalies and mitigate financial risks. This enhances the security and reliability of financial transactions, preventing fraudulent activities that could impact business operations.

Beyond invoice extraction, the system provides real-time insights into product demand and frequency, helping businesses track purchasing trends and optimize inventory management. The interactive dashboard offers visual analytics, allowing users to make data-driven decisions based on invoice trends. Furthermore, the invoice generation feature, powered by ReportLab, ensures consistency and professionalism in invoice formatting.

1.3 Advantages and Disadvantages of Smart Invoice Analyzer:

The Smart invoice Analyzer offers several advantages, making invoice processing more efficient and intelligent. It supports multi-language invoice extraction, allowing businesses to handle invoices from various regions without language barriers. By leveraging OCR and Gemini AI, the system ensures accurate data extraction, minimizing human effort and errors. The fraud detection feature enhances financial security by identifying suspicious transactions, while the real-time dashboard provides valuable insights into product demand and purchasing trends. Additionally, the tool automates invoice categorization and enables structured invoice generation using ReportLab, ensuring professional formatting. This automation significantly reduces processing time and costs, making it highly scalable for businesses of all sizes.

However, the system also has some limitations. Since fraud detection is based on an English-language dataset, its effectiveness may be limited for non-English invoices. The accuracy of OCR-based extraction can be affected by low-quality images or handwritten invoices, leading to occasional errors. Businesses using legacy financial systems may face integration challenges, requiring technical expertise for seamless adoption. Additionally, fraud detection algorithms may sometimes flag genuine invoices as fraudulent, leading to false positives. As the system is cloud-based, it depends on a stable internet connection for optimal performance. Despite these challenges, the Smart Invoice Analyzer remains a powerful solution for streamlining invoice processing, enhancing financial security, and providing businesses with valuable data-driven insights.

1.4 Summary

The Smart Invoice Analyzer is an AI-driven system that automates invoice extraction, translation, analysis, and fraud detection. Using OCR and the Gemini API, it accurately extracts key details from invoices in multiple formats and languages. The system features a dynamic dashboard for real-time insights into product demand, frequency, and financial trends. It also enables invoice generation using ReportLab, ensuring structured documentation. A custom dataset of 10,000 invoices supports fraud detection and product analysis.

CHAPTER-2
LITERATURE SURVEY

2. LITERATURE SURVEY

2.1 Literature Survey:

[1] Advancements in Document Analysis and Recognition: Insights from ICDAR 1993 The 2nd International Conference on Document Analysis and Recognition (ICDAR '93) took place from October 20 to 22, 1993, in Tsukuba Science City, Japan. This biennial conference served as a platform for researchers and practitioners to discuss advancements in document analysis and recognition technologies. The proceedings encompassed a wide array of topics, including character recognition, image processing, and neural network applications. For instance, one study explored the use of rubber band techniques for text character recognition and representation. Another paper presented a system for hand-printed character recognition utilizing artificial neural networks. The conference also featured research on converting raster images to object representations using knowledge-based image processing. Additionally, ergonomic improvements in text error detection and prevention within desktop publishing systems were discussed. The event highlighted the growing interest in applying neural networks to document analysis tasks. Overall, ICDAR '93 contributed significantly to the progression of document analysis research.Vatukoula Vishnu Teja Reddy and Thumma Dhyanchand In human-computer interaction, virtual mouse is implemented with fingertip recognition and hand gesture tracking based on image in a live video is one of the studies. The main objective is to find the solution for the finger tracking in the real world and the cursor control of a computer is still performed physically. The proposed system describes virtual mouse control using fingertip identification and hand gesture recognition..

[2] I. A. S. Ahmad and M. Man The paper "Multiple Types of Semi-structured Data Extraction Using Wrapper for Extraction of Image using DOM (WEID)" by I. A. S. Ahmad and M. Man, presented at the 2016 Regional Conference on Science, Technology, and Social Sciences in Singapore, introduces a method for extracting images from semi-structured web data using the Document Object Model (DOM). The authors developed an automated web extractor named WEID, implemented in PHP, capable of extracting images from web pages. Experimental results demonstrated the feasibility and effectiveness of this approach is successfully extracting images from web pages.

[3] E. Alfonseca and S. Manandhar In the paper "An Unsupervised Method for General named Entity Recognition and Automated Concept Discovery" by E. Alfonseca and S. Manandhar, presented at the 1st International Conference on General WordNet in 2002, the authors propose an unsupervised approach to named entity recognition and concept discovery. This method does not rely on labeled data or manually constructed gazetteers, making it adaptable to different languages

and domains. The approach leverages the aspect distributional properties of words and their co-occurrence patterns identify named entities and discover new concepts within a given text.

2.2 TRADITIONAL METHODS

[1] M. Ali, G. Tan, and A. Hussain propose a Bidirectional Recurrent Neural Network (BiRNN) model for Arabic Named Entity Recognition (NER) in their 2018 study. The model utilizes Bidirectional Long Short-Term Memory (BiLSTM) networks to capture past and future contexts in Arabic text. NER is a crucial task in (NLP), essential for extracting names of people, organizations, and locations. Due to Arabic's inherent complex morphology and lack of diacritics, traditional meth ods lack with high accuracy. The authors implement deep learning techniques to enhance recognition & classification performance. Their experimental results show significant improvement in Arabic NER compared to rule-based and statistical models. This research contributes to advancing Arabic NLP applications such as machine translation and information retrieval.

[2] J. Andersson 2020 report explores automatic invoice data extraction using a constraint satisfaction problem (CSP) approach. Invoices contain structured and semi-structured data, making automation challenging without predefined template. By formulating the extraction as CSP, the system identifies constraints among invoice elements like dates, amounts, and item descriptions. A constraint solver is used to match extracted text with predefined rules, reducing manual data entry efforts. The approach is designed to handle variations across different invoice formats without actual explicit training on each type.Experiments suggest that CSP-based extraction improves efficiency, accuracy, and also scalability in the invoice processing. This research contributes to automating financial document analysis, aiding businesses in invoice management

2.3 Issues in Multi-Language Invoice Processing and Data Extraction

Handling Multi-Language Invoices:

Invoice processing systems must accurately extract information from invoices written in various languages. However, existing solutions often struggle with recognizing different scripts, leading to misinterpretation of key details such as product descriptions, pricing, and tax information. Ensuring high accuracy across diverse languages remains a significant challenge.

Background Noise in Scanned Documents:

Invoices scanned or captured as images often contain noise, such as blurred text, skewed alignments, or faded sections. Such distortions can hinder Optical Character Recognition (OCR) accuracy, affecting the reliability of extracted data**.**

Processing Large Volumes Efficiently:

Businesses receive invoices in bulk, and processing thousands of invoices manually is time-consuming. Even automated solutions must be optimized for speed and efficiency to handle large datasets without delays in financial reporting.

Integrating Real-Time Insights and Demand Analysis:

While extracting invoice data is crucial, businesses also need meaningful insights from it. Many existing solutions lack advanced analytics for tracking product demand, frequency of purchases, and financial trends, limiting their ability to provide actionable business intelligence.

2.4 MOTIVATION

With businesses handling invoices in multiple languages and formats, existing automated systems struggle with accurate extraction, translation, and analysis. This leads to inefficiencies in financial tracking and decision-making. The **Smart Invoice Analyzer** aims to bridge this gap by leveraging **OCR and Gemini AI** to automate multi-language invoice processing while providing actionable insights on product demand and financial trends . By streamlining invoice management and enhancing data accuracy, this solution empowers businesses to optimize operations, reduce manual intervention, and make informed financial decisions efficient.

2.5 PROBLEM STATEMENT

Invoice processing is a critical business function, yet existing systems lack automation, multi-language support, and fraud detection, leading to inefficiencies. The **Smart Invoice Analyzer** addresses these challenges using **OCR, AI-driven translation, and fraud detection** to extract, translate, and analyze invoices accurately. It enhances financial management by identifying fraudulent invoices, providing product demand insights, and streamlining invoice generation, making the process faster, smarter, and more efficient for global businesses.

2.6 SUMMARY

Through this review, we identified the challenges in existing invoice processing systems, particularly in areas such as fraud detection, multi-language extraction, and product insights. While there is room for improvement in these areas, we aim to overcome these issues by implementing more efficient fraud detection algorithms, improving data extraction capabilities using advanced OCR and AI tools, and providing accurate, real-time product insights. We conclude that the **SMART INVOICE ANALYZER** system will address these challenges effectively, offering a reliable solution without these limitations.

CHAPTER 3
SYSTEM ANALYSIS

3. SYSTEM ANALYSIS

3.1 System Analysis

System analysis is the process of examining a system's structure, functionality, and interactions to optimize its efficiency and achieve business objectives. It involves breaking down complex processes into smaller components to improve decision-making, streamline workflows, and enhance overall system performance. In modern financial management, invoice processing plays a crucial role in maintaining accurate records, tracking transactions, and ensuring financial transparency. However, traditional automated solutions often struggle with challenges such as multi-language support, fraud detection, and structured data extraction. The Smart Invoice Analyzer is designed to address these challenges by integrating OCR, AI-powered translation, fraud detection, and analytical insights into a single, automated system.

The project involves multiple stages, including invoice image processing, text extraction, translation, fraud detection, and product demand analysis. The system extracts key invoice details such as invoice number, invoice date, product name, company, quantity, unit price, total price, customer name, and fraudulent indicators from scanned images or PDFs. By leveraging Gemini AI and OCR technology, it ensures accurate extraction of invoice data, even from multilingual invoices, facilitating seamless processing for global businesses. The extracted information is then analyzed to provide useful insights into financial trends, product demand, and potential fraudulent activities.

One of the critical aspects of this system is fraud detection, where invoice patterns are analyzed using a custom dataset containing 10,000 invoices. The system identifies anomalies and inconsistencies in invoice data, helping businesses prevent financial losses due to fraudulent transactions. Additionally, the product insights module allows users to track the demand and frequency of purchases across various electronic gadgets such as laptops, TVs, mobiles, and more. This feature aids businesses in making data-driven decisions regarding inventory management and sales forecasting.

Another essential feature of the Smart Invoice Analyzer is its invoice generation module, which utilizes ReportLab to create structured and professional invoices. This ensures uniformity in invoice formats, improving readability and compliance with financial regulations. Furthermore, the system provides an interactive dashboard where users can visualize invoice trends, financial patterns, and purchasing behaviors through graphical representations and reports.

By eliminating manual inefficiencies, enhancing multilingual capabilities, and providing intelligent financial insights, the Smart Invoice Analyzer Tool revolutionizes invoice management. It reduces processing time, minimizes human errors, and enhances financial security by detecting fraudulent invoices. This innovative solution is designed to make invoice processing faster, smarter, and more accessible for businesses operating on a global scale, ultimately improving their financial operations and decision-making processes.

3.2 Existing System

The existing invoice processing systems primarily rely on either manual data entry or semi-automated tools that extract information using basic OCR technology. In traditional methods, businesses receive invoices in various formats, such as scanned images, PDFs, and handwritten documents, which require significant human effort for verification and entry into accounting systems. This manual process is time-consuming, error-prone, and inefficient, especially when handling large volumes of invoices. Even with basic automation tools, accuracy is often compromised due to poor image quality, variations in invoice formats, and lack of advanced data validation techniques.

Figure 3.2 : Existing Systems

Furthermore, most existing systems lack robust multilingual support, making it challenging for global businesses to process invoices in different languages. Many solutions are designed to handle only structured invoices with predefined formats, limiting their flexibility in real- world applications. Additionally, fraud detection mechanisms in current systems are minimal, relying mainly on manual reviews rather than AI-driven analysis, increasing the risk of financial

discrepancies. Another limitation is the absence of advanced analytics, preventing businesses from gaining insights into purchasing trends, product demand, and financial patterns.

Overall, the existing invoice processing solutions have limitations in automation, accuracy, multilingual capabilities, fraud detection, and data-driven decision-making. These gaps highlight the need for an advanced, AI-powered system like the Smart Invoice Analyzer Tool, which integrates intelligent extraction, translation, fraud detection, and analytics to streamline invoice management efficiently.

3.2.1 Disadvantages of Existing System

Manual Data Entry and Errors

- Traditional invoice processing relies heavily on manual data entry, which is time-consuming and prone to human errors.
- Incorrect data entry can lead to financial discrepancies and inefficiencies in business operations.

Slow Processing Speed

- Handling a large volume of invoices manually or using semi-automated tools results in significant delays.
- Businesses may face late payments or processing bottlenecks due to slow invoice verification.

Limited OCR Accuracy

- Many existing systems use basic OCR technology, which struggles with varying invoice formats, handwritten text, and low-quality scans.
- Extracted data often requires manual correction, reducing the efficiency of automation

Lack of Multilingual Support

- Most current invoice processing tools cannot extract and translate invoice details from multiple languages.
- This limitation makes it difficult for global businesses to process international invoices effectively.

Minimal Fraud Detection

- Existing systems lack advanced fraud detection mechanisms, making it difficult to identify fraudulent invoices.
- Businesses must rely on manual checks to detect anomalies, increasing the risk of financial fraud.

No Real-Time Insights or Analytics

- Traditional systems do not provide interactive dashboards or data analysis features.
- Businesses miss out on valuable insights related to purchasing trends, product demand, and financial forecasts.

Inefficient Invoice Generation

- Many existing solutions do not offer structured invoice generation features.
- Users often need to create invoices manually, leading to inconsistencies in formatting and presentation.

3.3 Proposed System

The Smart Invoice Analyzer Tool is an advanced AI-driven invoice processing system designed to address the inefficiencies of traditional invoice management. By integrating OCR, AI-based translation, fraud detection, and analytical insights, the system automates invoice extraction, categorization, and analysis. Unlike conventional methods, it supports multi-language invoices, making it suitable for global business operations.

This system extracts key invoice details such as invoice number, invoice date, product name, company, quantity, unit price, total price, customer name, and fraudulent indicators from scanned images or PDFs using OCR and the Gemini API. Custom dataset of 10,000 invoices to identify potential anomalies. Additionally, product insights and demand analysis enable businesses to track purchasing patterns and forecast trends efficiently.

The system also includes an interactive dashboard for real-time monitoring of invoice trends and financial data visualization. Furthermore, it features an invoice generation module using ReportLab, ensuring structured and professional invoice creation. By leveraging AI, automation, and analytics, the Smart Invoice Analyzer Tool improves efficiency, enhances financial security,

and provides organizations with data-driven insights for better decision- making. The system can be broken down into five main components

- Invoice Extraction
- Fraud Detection
- Product Insights & Demand Analysis
- Interactive Dashboard
- Invoice Generation

Figure 3.3 Workflow of Smart Invoice Analyzer

3.3.1 Advantages of Proposed System

Automated Invoice Processing – Eliminates manual effort by extracting and analyzing invoice details using AI and OCR, improving efficiency and accuracy.

Multi-Language Invoice Extraction – Supports invoices in different languages by translating key details into English, enabling global usability.

Fraud Detection – Identifies potentially fraudulent invoices using machine learning trained on a custom dataset, enhancing financial security**.**

Product Insights and Demand Analysis – Helps businesses track product demand, purchase frequency, and market trends over time.

Real-Time Data Visualization – Provides interactive dashboards with graphical insights into invoice trends, financial patterns, and product analytics.

Invoice Generation – Enables structured invoice creation with a professional format using ReportLab, ensuring consistency in business transactions.

Time and Cost Efficiency – Reduces processing time and operational costs by automating invoice handling and analysis.

User-Friendly Interface – Designed with a simple and intuitive interface.

3.4 Summary

This chapter deals with problem statement that we have stated after performing literature survey, existing system and its drawbacks, the system that we have proposed and its advantages and how our proposed system can overcome the draw backs of existing system.

CHAPTER 4
SYSTEM REQUIREMENTS SPECIFICATIONS

4. SYSTEM REQUIREMENTS SPECIFICATIONS

4.1 Functional Requirements

Functional Requirements talks about a functionality that need to be carried out by a system or its component. A functionary is considered as a specification of behavior between outputs and inputs.The Users or Actors identified in Project are Admin and user.

USER	FUNCTIONALITIES
User	• Uploads invoices • Requests invoice translation • Checks if an invoice is fraudulent • Views extracted invoice details • Searches for product insights • Generates new invoices • Analyzes invoice trends and demand
Application	• Extracts text using OCR and Gemini API • Translates multi-language invoices • Analyzes invoices for fraud detection • Displays invoice summaries and product demand insights • Provides a dynamic dashboard with visual analytics • Generates structured invoices using ReportLab • Allows exporting of invoice reports (CSV/PDF)

Table 4.1 : User and Functionalities

4.2 Non-Functional Requirements

Non-functional Requirements states criteria that can be used to evaluate the operation of a system, rather than specific behaviors. They are different from Functional Requirements which define specific behavior or functionalities.

These are the quality constraints that our system satisfies according to the project

Portability

The ability of software to be transferred from one machine or system to another. A computer software application is considered portable to a new environment if the effort required to adapt it to the new environment is within reasonable limits.

Security

The protection of computer systems and information from harm, theft, and unauthorized use. It is the concept of implementing mechanisms in the construction of security to help it remain Functional (or resistant) to attacks

Maintainability

The ease with which a software system or component can be modified to correct faults, improve performance or other attributes, or adapt to a changed environment. A software product that is easy to extend with new functionality.

Reliability

The ability of a system or component to function under stated conditions for a specified period of time. It relates to operation rather than design of the program, and hence it is dynamic rather than static.

Scalability

The measure of a system's ability to increase or decrease in performance and cost in response to changes in application and system processing.

4.3 Hardware Requirements

In any software application, hardware requirements define the necessary physical computer resources essential for smooth operation. These requirements ensure that the system functions efficiently and meets the performance expectations of users. Hardware specifications typically include components such as the processor, memory (RAM), storage, graphics card, and other peripherals. The hardware capabilities play a crucial role in determining the system's speed, responsiveness, and ability to handle complex computations**.**

Ensuring proper hardware compatibility is vital for seamless integration and execution of software applications, particularly those involving AI-powered tools, data processing, and automation, like the Smart Invoice Analyzer Tool.

Proceesor	:	Intel Core i5/i7
RAM	:	8GB (Minimum), 16GB (Recommended)
Storage	:	256GB SSD (Minimum), 512GB SSD or higher

4.4 Software Requirements

Any software application defines the most common requirements known as the physical computer resources or hardware, A hardware requirements list regularly come with hardware compatibility list (HCL), particularly in case of operating systems. An HCL contains tested, compatible, and sometimes incompatible hardware devices for a particular operating system or application.

Operating System	**:**	Window XP/Windows10, Linux Windows Vista/Windows 7
Languages	**:**	Python
Python Libraries	**:**	pytesseract, pdfplumber, googletrans
IDE used	**:**	Visual Studio code

4.4.1 PYTHON

Python is an easy to learn, powerful programming language. It has efficient high- level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms.

The Python interpreter and the extensive standard library are freely available insource or binary form for all major platforms from the Python Web site, https://www.python.org/, and may be freely distributed. The Python interpreter is easily extended with new functions and data types implemented in C or C++ (or other languages callable from C). Python is also suitable as an extension language for customizable applications. This tutorial introduces the reader informally to the basic concepts and features of the Python language and system. It helps to have a Python

interpreter handy for hands-on experience, but all examples are self-contained, so the tutorial can be read off-line as well. For a description ofstandard objects and modules, see library-index. reference-index gives a more formal definition of the language. To write extensions in C or C++, read extending- index and c- api-index.

There are also several books covering Python in depth. If you do much work on computers, eventually you find that there's some tasks you'dlike to automate. For example, you may wish to perform a search-and-replace over a large number of text files, or rename and rearrange a bunch of photo files in a complicated way. Perhaps you'd like to write a small custom database, or a specialized GUI application, or a simple game. If you're a professional software developer, you may have to work with several C/C++/Java libraries but find the usual write/compile/test/re-compile cycle is too slow. Perhaps you're writing a test suite for such a library and find writing the testing code a tedious task. Or maybe you've written a program that could use anextension language, and you don't want to design and implement a whole new language for your application. Python is just the language for you. You could write a Unix shell script or Windows batch files for some of these tasks, but shell scripts are best at moving around filesand changing text data, not well- suited for GUI applications or games. You could write a C/C++/Java program, but it can take a lot of development time to get even a first-draft program. Python is simpler to use, available on Windows, Mac OS X, and Unix operating systems, and will help you get the job done more quicklyBecause of its more general data types Python is applicable to a much larger problem domain than Awk or even Perl, yet many things are at least as easy in Python as in those languages. Python allows you to split your program into modules that can be reused in other Python programs.

It comes with a large collection of standard modules that you can use as the basis of your programs — or as examples to start learning to program in Python. Some of these modules provide things like file I/O, system calls, sockets, and even interfaces to graphical user interface toolkits like Tk Python is an interpreted language, which can save you considerable time during program development because no compilation and linking is necessary. The interpreter can be used interactively, which makes it easy to experiment with features of the language, to write throw-away programs, or to test functions during bottom-up program development. It is alsoa handydesk calculator. Python enables programs to be written compactly and readably. Programs written in Python are typically much shorter than equivalent C, C++, or Java programs, for several reasons.
Python is extensible: if you know how to program in C it is easy to add a new built-in function or module to the interpreter, either to perform critical operations at maximum speed, or to link Pythonprograms to libraries that may only be available in binary form (such as a vendor- specific graphics library).

4.4.2 APPLICATIONS OF PYTHON

Web Development

It is one of the most astonishing applications of Python. This is because Python comes up with a wide range of frameworks like Django, Flask, Bottle, and a lot more that provide ease to developers. Furthermore, Python has inbuilt libraries and tools which make the web development process completely effortless. Use of Python for web development also offers:

- Amazing visualization
- Convenience in development
- Enhanced security
- Fast development process

Machine Learning and Artificial Intelligence

Machine Learning and Artificial Intelligence are the hottest subjects right now. Python along with its inbuilt libraries and tools facilitate the development of AI and ML algorithms. Further, it offers simple, concise, and readable code which makes it easier for developers to write complex algorithms and provide a versatile flow. NumPy for complex data analysis

- Keras for Machine learning
- SciPy for technical computing
- Seaborn for data visualization

Data Science

Data science involves data collection, data sorting, data analysis, and data visualization provides amazing functionality to tackle statistics and complex mathematical calculations. The presence of in-built libraries provides convenience to data science professionals. Some of the popular libraries that provide ease in the data science process are TensorFlow, Pandas, and Socket learning. These libraries provide an ecosystem for fine-tuning data models, data pre-processing, and performing complex data analysis.

Game Development

With the rapidly growing gaming industry Python has proved to be an exceptional option for game development. Popular games like Pirates of the Caribbean, Bridge commander, and Battlefield 2 use Python programming for a wide range of functionalities and addons. The presence of popular 2D and 3D gaming libraries like pygame, panda3D, and Cocos2D make the game development process completely effortless.

Audio and Visual Applications

Audio and video applications are undoubtedly the most amazing feature of Python.Python is equipped with a lot of tools and libraries to accomplish your task flawlessly. Applications that are coded in Python include popular ones like Netflix, Spotify, andYouTube. This can be handled by libraries like

- Mingus
- OpenCV

Software Development

Python is just the perfect option for software development. Popular applications like Google, Netflix, and Reddit all use Python. This language offers amazing features like:

- Inbuilt libraries and frameworks to provide ease of development.
- Enhanced code reusability and readability
- High compatibility
- Platform independence

4.4.3 Why Python:

Python is incredibly easy to learn and use for beginners and newcomers in the industry. The language is the most accessible among all the programming languages available because it has simplified syntax that is not complicated at all and gives more emphasis on natural language. Due to its ease of learning and usage, Python codes can easily be written and executed much faster than other programming languages.

One of the main reasons why Python's popularity has exponentially grown is due to its simplicity in syntax so that it could be easy to read and developed by amateur professionals as well. Some of the most popular libraries available in Python are NumPy and SciPy, Django, and others that are used for different purposes.

Why OCR (pytesseract):

OCR (Optical Character Recognition) is crucial for extracting text from invoices that exist as scanned images or non-selectable PDFs. Many invoices are received as image-based documents, making it impossible to copy or process the text directly. pytesseract, built on Tesseract OCR, helps in recognizing and converting printed or handwritten text into machine-readable format. This enables accurate data extraction from invoices where standard text-parsing methods fail. OCR

ensures that even low-quality scans can be processed efficiently, improving automation in invoice management.

Figure 4.4.1 OCR

Why pdfplumber:

pdfplumber is specifically designed for extracting structured text from **digitally generated PDFs** where the text is already selectable. Unlike OCR, which processes images, pdfplumber can directly read and extract text, tables, and structured data from a PDF file without losing formatting accuracy. This makes it much faster and more precise when dealing with digital invoices. By using pdfplumber, the system avoids unnecessary image conversions, leading to improved efficiency in extracting invoice details with high accuracy.

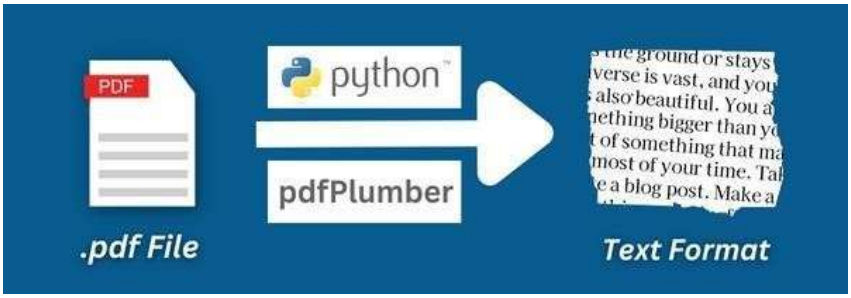


Figure 4.4.2 pdfPlumber

4.5 DataSet Analysis:

4.5.1 Overview of the Dataset

The dataset used in the Smart Invoice Analyzer Tool plays a crucial role in enabling accurate invoice extraction, fraud detection, product demand analysis, and invoice generation. It consists of 10,000 invoices in PDF and image formats, containing structured details essential for financial processing and decision-making.

4.5.2 Dataset Structure

The dataset used in the Smart Invoice Analyzer Tool consists of structured invoice-related attributes essential for extraction, analysis, and fraud detection.

Attribute Name	Description
Invoice_No	Unique identification number for each invoice.
Invoice Date	The date when the invoice was generated.
Product_Name	The name of the product purchased.
Company	The manufacturer or vendor of the product.
Quantity	Number of units purchased in the invoice.
Unit_Price	Price per unit of the product.
Total_Price	The total cost (Quantity × Unit Price).
Description	Brief details about the product.
Usage	Specifies how the product is intended to be used.
How_to_Use	Instructions for using the product.
Customer_name	Name of the customer or organization billed.
Fraudulent	ndicates whether the invoice is genuine or fraudulent.

Table 4.5.2 : Dataset Structure

4.5.3 Role of the Dataset in the Project

The dataset is used in multiple aspects of the Smart Invoice Analyzer Tool, such as:

- **Invoice Extraction**: Helps train the OCR and Gemini API models to extract invoice deta- ils accurately.
- **Fraud Detection**: By analyzing patterns in fraudulent invoices, the model identifies suspicious invoices.
- **Product Demand Analysis**: The dataset is used to track product sales trends, helping businesses understand market demand.
- **Multi-Language Support**: The tool extracts details from invoices in multiple languages and translates them into English.
- **Invoice Generation**: The dataset serves as a reference for structuring and generating invoices professionally.

Importance of a Custom Dataset:

A custom dataset is essential for improving the accuracy of AI-based financial tools. Unlike pre-existing datasets, this dataset is curated specifically for invoice-related applications, ensuring

that the machine learning models receive high-quality, domain-specific training data. The fraud detection model is particularly dependent on this dataset, as it helps distinguish between genuine and suspicious invoices based on previous fraudulent patterns.

4.6 Challenges in Dataset Handling:

While using the dataset, certain challenges must be addressed, including:

- Data Quality: Ensuring that extracted invoice details are correct and consistent.
- Multi-Language Processing: Handling non-English invoices with OCR and translation without losing important details.
- Fraud Detection Limitations: Since fraud indicators vary across different businesses, more labeled data may be required for better fraud identification.

4.7 Support Vector Classifier (SVC) :

The Support Vector Classifier (SVC) is a powerful machine learning algorithm used for fraud detection in invoices. It operates by finding the optimal hyperplane that best separates fraudulent and legitimate invoices in a high-dimensional space. This method enhances accuracy and robustness, making it effective for identifying fraudulent transactions.

In the Smart Invoice Analyzer, the SVC model is trained on a custom dataset containing 10,000 invoices, including various attributes such as Invoice Number, Invoice Date, Product Name, Company, Quantity, Unit Price, Total Price, Customer Name, and Fraudulent Indicator. The model 25nvoice25 historical invoice patterns and identifies anomalies that may indicate fraud. It considers multiple factors such as unusual pricing, abnormal frequency of certain products, and discrepancies in company details to classify an invoice as fraudulent or legitimate.

The key advantages of using SVC for fraud detection include:

- **High Accuracy**: Finds the optimal decision boundary for precise classification.
- **Robustness to Noise**: Effectively handles outliers and complex patterns in data.
- **Feature Importance Analysis**: Identifies which invoice attributes contribute the most to fraud detection.

By implementing this machine learning approach, the system enhances financial security by preventing fraudulent transactions and providing businesses with data-driven insights into suspicious invoices.

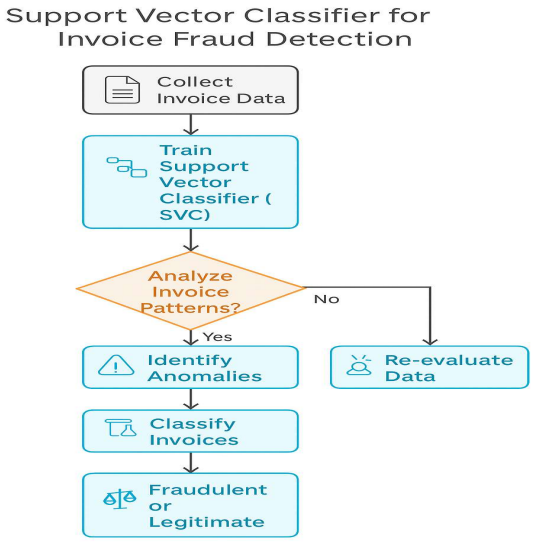


Figure 4.6.1 Support Vector Classifier for Fraud Detection

4.8 Summary :

In this chapter we discussed about the system requirements like what hardware is required and what are the software requirements one must have to know to use our model along with functional and non-functional requirements

CHAPTER 5
SYSTEM DESIGN

5. SYSTEM DESIGN

5.1 Introduction

System design is the process of transforming project requirements into a structured representation of software architecture, ensuring that all functional and non-functional aspects are addressed. It serves as a blueprint for developing a scalable, efficient, and high- performance system. The Smart Invoice Analyzer Tool is designed to automate invoice extraction, fraud detection, product insights, and invoice generation, integrating OCR, Gemini API, machine learning, and natural language processing to provide an end-to-end solution for invoice management. The Smart Invoice Analyzer Tool follows a structured process to handle invoices efficiently. First, the user uploads an invoice to the system. The tool then extracts relevant details using Optical Character Recognition (OCR). Next, fraud detection algorithms analyze the extracted data to identify potential issues. Once verified, an invoice summary is generated, highlighting key financial details. The system also provides product insights based on the extracted data. If needed, the invoice can be translated into different languages for better accessibility. Finally, users can generate a new invoice with updated details. This streamlined process ensures accuracy, security, and efficiency in invoice management.

The system follows a modular approach, where each component is designed to handle specific functionalities such as invoice data extraction, fraud classification, product analysis, and multi-language translation. The user uploads an invoice in image or PDF format, and the OCR module extracts key details. The extracted information is processed by the fraud detection model, which uses a Random Forest Classifier to determine if the invoice is legitimate. Additionally, the system provides real-time invoice summaries, product insights, and visualization dashboards, ensuring transparency and easy financial tracking. For seamless operations across multiple languages, automated translation features are incorporated, allowing businesses to Interpret non-English invoices effortless.

The system also includes an invoice generation module, built using ReportLab, which enables users to generate structured invoices with enhanced formatting. A custom dataset of 10,000 invoices is utilized to train models for fraud detection and product demand analysis. Through this structured design approach, the Smart Invoice Analyzer Tool optimizes invoice processing, minimizes manual effort, and provides businesses with actionable insights.System Architecture

5.2 System Architecture

The architecture of the system is a conceptual model that defines the arrangement and interaction of various components that make up the Smart Invoice Analyzer Tool. The overall system architecture, as shown in Figure 4.2, consists of multiple core modules, including Invoice Pre-processing, Data Extraction, Fraud Detection, Multi-Language Translation, and Invoice Generation.

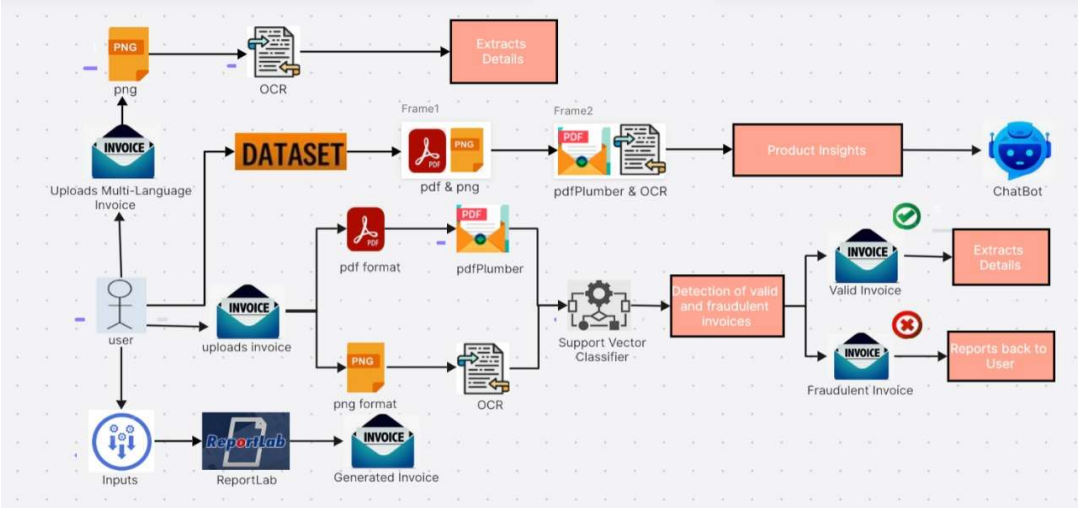


Figure 5.2.1 System Architecture

5.3 Modules

Modules are used primarily to group different functionalities that contribute to the overall working of the Smart Invoice Analyzer Tool. Each module is designed to handle a specific task related to invoice processing, fraud detection, translation, and insights.

The modules used in Smart Invoice Analyzer Tool are:

a) Fraud Detection
b) Multi-Language Invoice Extraction
c) Product Insights
d) Invoice Generation

5.3.1 FRAUD DETECTION

- This module is responsible for detecting fraudulent invoices.
- The user uploads an invoice, and the system extracts relevant details from it.
- The extracted details are compared with a custom dataset of **10,000 invoices** to identify anomalies.
- If fraudulent patterns are detected, the invoice is marked as fraudulent; otherwise, a summary of the invoice is generated.

5.3.2 MULTI-LANGUAGE INVOICE EXTRACTION

- This module allows users to upload invoices in different languages.
- The system extracts key details such as Invoice Number, Date, Product Name, Company, Quan- Total Price, and Status.
- It translates the extracted text into English using AI-powered translation.
- The system categorizes invoices and provides invoice-related insights.

5.3.3 PRODUCT INSIGHTS

- Users can input a product name to retrieve insights.
- The system provides the product description, frequency of appearance in invoices, and demand time.
- The insights are based on data extracted from previously processed invoices.

5.3.4 INVOICE GENERATION

- This module allows users to generate structured invoices.
- The system uses ReportLab to create professional invoices in PDF format.

5.4 UML Diagrams used in Design

Design is the first step in the development phase for an engineered product or system. Design is the place where quality is fostered in software development. Design is the only way that we can accurately translate a user's requirements into a finished software product or system. Software design serves as the foundation for all software engineers and software maintenance steps that follow. Without design, we risk building an unstable design-one that will fail when small changes are made, one that may be difficult to test, and one whose quantity cannot be accessed until late in the software engineering process.

Taking software requirements specification document of analysis phase as input to the design phase we have drawn Unified Modelling Language (UML) diagrams. UML depends on the visual modelling of the system. Visual modelling is the process of taking the information from the model and displaying it graphically using some sort of standards set of graphical elements. UML Diagrams are drawn using the Star UML Diagrammed Software. Complexity is better understood when it is displayed visually rather than written textually. By producing visual models of a system, one can understand how system works on several levels and can model the interactions between the users and the system. Each UML diagram is designed to let developers and customers view a software system from a different perspective and in varying degrees of abstraction.

5.4.1 Use Case Diagram:

Use Case Diagram captures the system's functionality and requirements by using actors and use cases. Use Cases model the services, tasks, function that a system needsto perform. It models how an external entity interacts with the system to make it work. Use cases represent high-level functionalities and how a user will handle the system. Actors can be defined as something that interacts with the system. Actors can be a human user, some internal applications, or may be some external applications. When we are planning to draw a Use Case diagram, we should have the following itemsidentified. Functionalities to be represented as use case

- Actors
- Relationships among the use cases and actors.

There can be following five Associations in Use case diagrams

- Association between actor and use case
- Generalization of an actor
- Extend between two use cases
- Include between two use cases Generalization of a use case

5.4.2 Scenarios

A Scenario is a specific sequence of actions and interactions between actors andthe system it is also called a use case instance. It is one particular story of using a system, orone path through the use case; for example, the scenario of successfully purchasing items withcash, or the scenario of failing to purchase items because of a credit payment denial. Informally then, a use case is a collection of related success and failure scenarios thatdescribe an actor using a system to support a goal.

5.4.3 Use case diagram representing overview of the System

Actors identified are Admin and User. Associations may be unidirectional or bidirectional. An undirected association relationship is an association that is navigable in onlyone direction and in which the control flows from one classifier to another only one of the association ends specifies navigability. The association is represented by a line with arrow atone end. A bidirectional association relationship is an association that is navigable both directions and in which the control flows from both classifiers to another .The association is represented by a normal line between use case and an actor.

There are 3 actors present in the diagram

I. USER

- Sign-up
- Sign-in
- Use Invoice Generator
- Upload Invoices
- Detect Fraud
- Utilize Product Insights

II. Chatbot

- Process Query
- Fetch Response

III. System

- Store Data
- Retrieve Data
- Download Invoice
- Extract Data

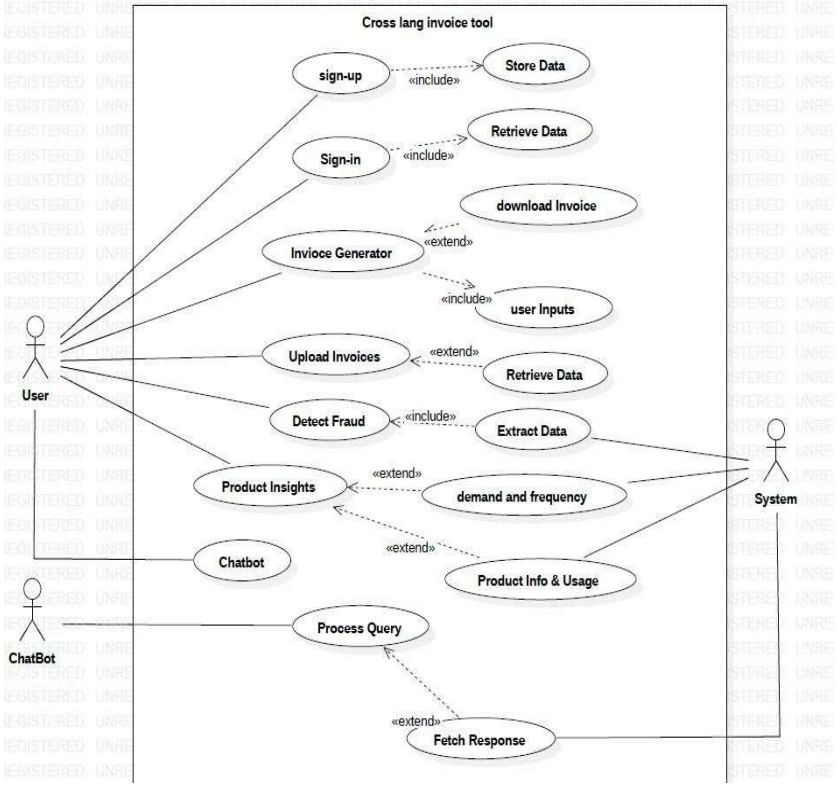


Figure 5.4.3 Use Case Diagram

5.4.4 Class Diagram:

Class Diagrams helps to design the structural model of the system by showing classes, attributes, operations and relationships between classes.The Class Diagram shown in fig:5.3.2.1 displays the classes that represent different users with their respective attributes and functionalities. It also displays classes that represent different services provided by the Proposed System.

Based on your provided class diagram, here's a description of the classes in similar format to your examples:

CLASS: USER

Here it contains the information related to the user of the application:

- Username
- Email
- Password
- Confirm Password
- Signup()
- Login()
- Storedata()
- Retrievedata()

CLASS: INVOICE GENERATOR

Here it contains the information related to invoice generation:

- Generator_invoices
- Invoice_data
- Log
- Industry
- Item_label
- Theme_color
- Template_style
- Collect_user_inputs()
- Generate_invoice()
- Download_invoice()

CLASS: INVOICE EXTRACTOR

Here it contains the information related to invoice extraction:

- InvoiceData
- Fraud Status
- Extract_data()
- Detect_fraud()
- Get_invoice_info()

CLASS: INVOICE EXTRACTOR BY IMAGE

Here it contains the information related to invoice extraction from images:

- Extract_Data()
- Detect_fraud()
- Get_invoice_info()

CLASS: INVOICE EXTRACTOR BY PDF

Here it contains the information related to invoice extraction from pdfs:

- Extract_Data()
- Detect_fraud()
- Get_invoice_info()

CLASS:PRODUCT INSIGHTS

Here it contains the information related to product insights:

- Product_data
- Quantity
- Price
- Invoices
- Get_demand_analysis()
- Get_frequency_analysis()
- Product_description & info()

CLASS:CHATBOT

Here it contains the information related to the chatbot functionality:

- Query
- Response
- Process_query()
- Fetch_response()

CLASS: DASHBOARD

Here it contains the information related to the dashboard display:

- Amount
- Dates
- Amount_in_period()
- Paid_and_unpaids()

CLASS: SYSTEM

Here it contains the information related to the system resources:

- Operating System
- Memory
- Processor
- Response()

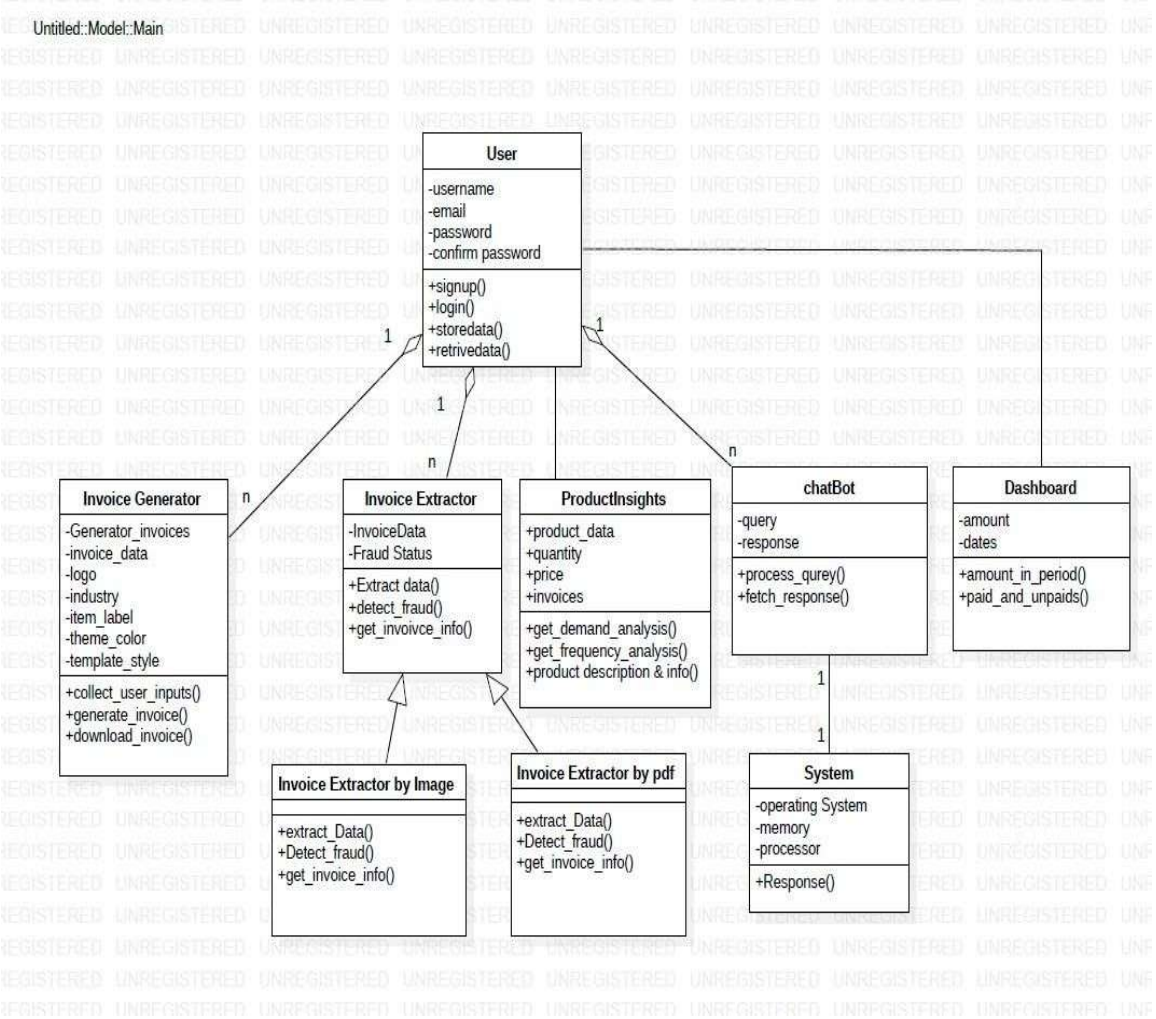


Figure 5.4.4 class diagram

5.4.5 Sequence Diagram:

Sequence Diagram displays the time sequence of the objects participating in the interaction. This consists of the vertical dimension (time) and horizontal dimension (different objects).

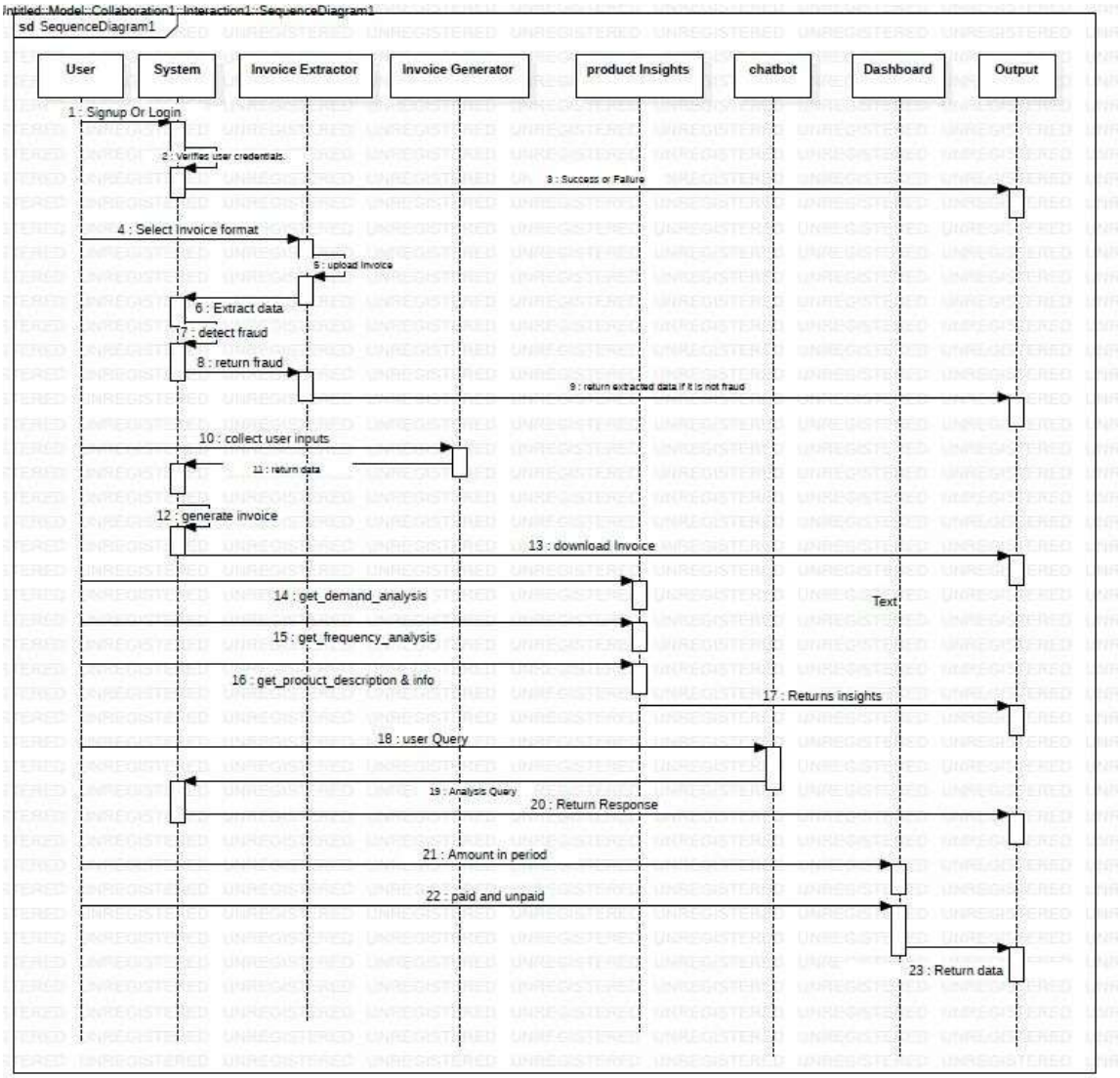


Figure 5.4.5: Sequence diagram

5.4.6 Activity Diagram

Activity Diagrams describe how activities are coordinated to provide a service which can be at different levels of abstraction. Typically, an event needs to be achieved by some operations, particularly where the operation is intended to achieve a number of different things that require coordination, or how the events in a single use caserelate to one another, in particular, use cases where activities may overlap and require coordination. It is also suitable for modelling how a collection of use cases coordinates torepresent business workflows.

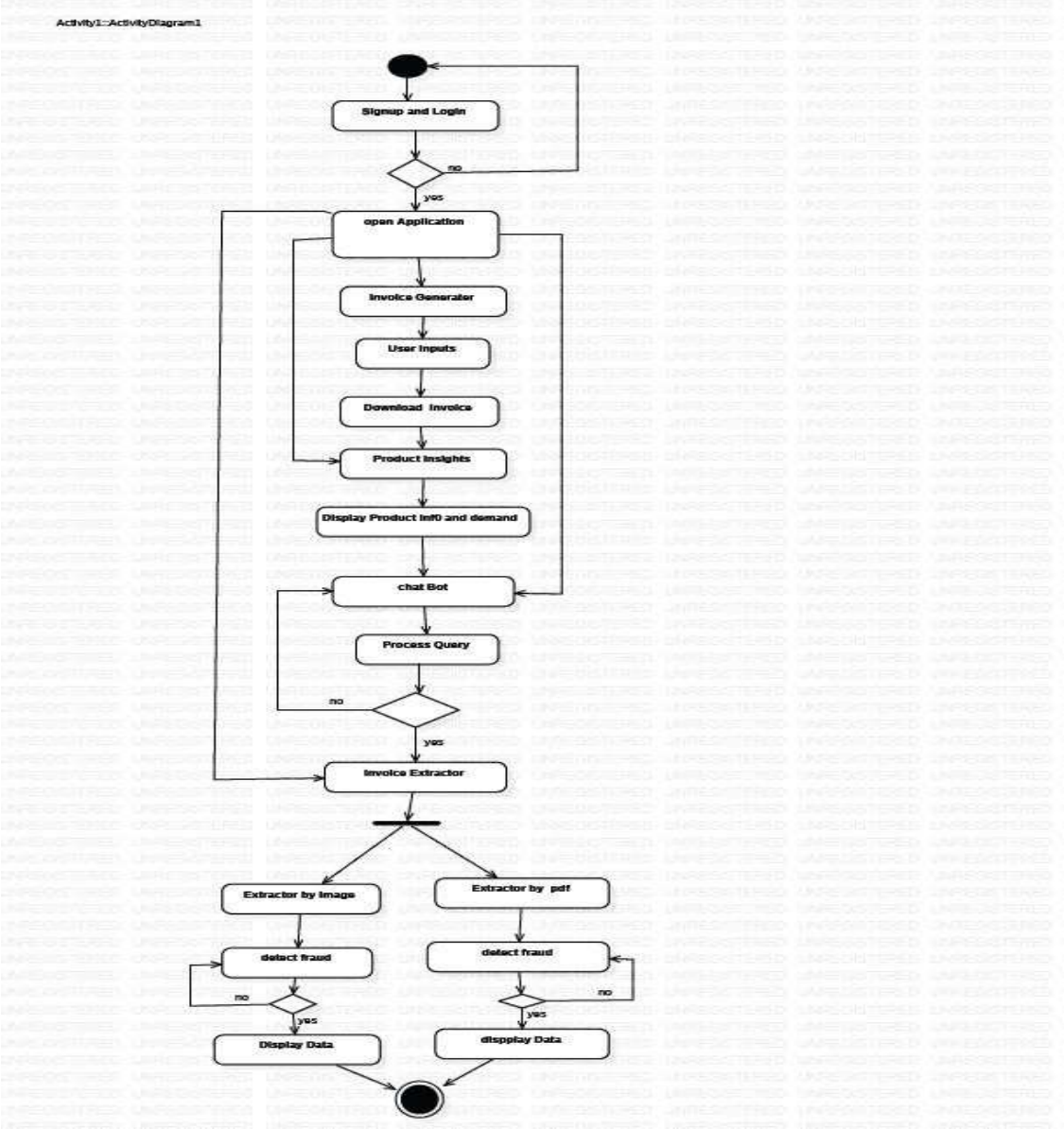


Figure 5.4.6 : Activity diagram

5.5 Summary

After analyzing the interactive diagrams and data flow diagrams of the whole project we have a clear understanding of the procedure that have to be done. We have all the requirements needed for the project and are ready for the implementation phase.

CHAPTER 6
SYSTEM IMPLEMENTATION

6. SYSTEM IMPLEMENTATION

6.1 INTRODUCTION:

System implementation is the process of putting the designed system into action or making it functional. It involves the developing the system in a way that meets the requirements and objectives specified during the system analysis and design phase. It involves writing code for the system based on the design specifications. In this chapter all the required libraries are explained briefly.

6.2 PROJECT MODULES:

6.2.1 Streamlit

Description:

Streamlit is an open-source Python framework that allows the rapid development of interactive web applications. It simplifies UI creation for data science and machine learning applications, offering seamless integration with the Python libraries such as NumPy, Pandas, Matplotlib, Scikit-learn, and more.

Application in SMART INVOICE ANALYZER TOOL:

Streamlit serves as the core framework for building the web interface of the system. It provides an intuitive UI for users to upload invoices, view extracted data, and analyze invoice trends through dynamic dashboards. The framework also makes the multi-language translation feature, allowing real-time conversion of invoice details into many different languages. The fraud detection system is seamlessly integrated, displaying whether an invoice is fraudulent or legitimate instantly. Additionally, Streamlit enables interactive visualization for product demand analysis using graphs and charts. With its lightweight design and Python compatibility, it ensures smooth user experience.

INSTALLATION:

```
pip install Streamlit
```

6.2.2 Datetime

Description:

The datetime module provides functionalities for handling date and time, allowing users

Application in Smart Invoice Analyzer :

The system uses datetime to record invoice timestamps, ensuring that each uploaded invoice is associated with an accurate date and time. It is also used for categorizing invoices based on their due dates, allowing automated reminders for overdue payments.

Additionally, datetime assists in fraud detection by analyzing invoice patterns based on historical timestamps.

6.2.3 io

Description:

The io module in Python provides tools for handling in-memory file operations without needing physical storage. It is particularly useful for reading and writing files in binary or text format.

6.2.4 qrcode

Description:

The qrcode module allows for generating QR codes, which can store information such as text, URLs, or structured data**.**

Application in SMART INVOICE ANALYZER:

The project generates QR codes for invoices, allowing businesses to quickly verify and retrieve invoice details by scanning the code. This feature is particularly useful for tracking invoice authenticity and simplifying record-keeping.

INSTALLATION:

pip install qrcode

6.2.5 ReportLab

Description:

ReportLab is a Python library for creating PDFs, supporting elements like text, images, tables, and barcodes**.**

Application in SMART INVOICE ANALYZER :

This library is used for generating invoices in PDF format with well-structured layouts. It also allows the inclusion of business details, product lists, invoice totals, and QR codes.

INSTALLATION:

pip install reportlab

6.2.6 OS

Description:

The os module provides functions to interact with the operating system, including file handling and directory management.

Application in SMART INVOICE ANALYZER:

The project uses os to manage invoice storage, create necessary directories for saving

extracted data, and handle file operations dynamically.

6.2.7 Streamlit Components Description:

Streamlit Components allow developers to integrate custom HTML, CSS, and JS into Streamlit applications, providing more interactive UI elements.

Application in SMART INVOICE ANALYZER:

It is used to embed custom invoice summary sections, create interactive dashboards,and enhance user experience by displaying invoice details in a visually appealing format.

6.2.8 Pandas Description:

Pandas is a powerful data analysis library in Python that provides data structures such as DataFrames and Series for efficient data manipulation.

Application in SMART INVOICE ANALYZER:

Pandas is used to store and process invoice data, analyze trends, and filter fraudulent invoices.

6.2.9 Matplotlib

Description:

Matplotlib is a Python plotting library that allows users to create visualizations

Application in SMART INVOICE ANALYZER :

It is used to generate invoice fraud detection graphs, product demand trends, and invoice distribution charts, helping businesses analyze data effectively.

6.2.10 Google Generative AI (Gemini API)

Description:

Google Generative AI (Gemini API) allows AI-powered text generation, insights extraction, and intelligent responses.

Application in SMART INVOICE ANALYZER TOOL:

This API is integrated for automatic invoice summarization and multi-language invoice text translation, enhancing the system's ability to process diverse invoices efficiently.

6.2.11 PyTesseract (OCR)

Description:

PyTesseract is an OCR(Optical Character Recognition) tool that extracts text from images.

Application in SMART INVOICE ANALYZER TOOL:

It is used to extract text from invoice images, allowing seamless processing of scanned or photographed invoices.

INSTALLATION

pip install pytesseract

6.2.12 Pickle

Description:

Pickle is a Python module for serializing and deserializing objects.

Application in SMART INVOICE ANALYZER TOOL:

Pickle is used to save trained fraud detection models, ensuring that predictions can be made without retraining the model every time

6.3 THE CODE OF OUR PROJECT

First all the required modules are imported into the file where the functions and their functionality id defined.After each function is defined with its functionality with the help of the improved modules.

It is as follows:

```
import streamlit as st
import datetime
import io
import qrcode
from reportlab.lib.pagesizes import letter
from reportlab.pdfgen import canvas
from reportlab.lib import colors
from reportlab.lib.utils import ImageReader
import re
import os
import base64
import streamlit.components.v1 as components
import pandas as pd
import matplotlib.pyplot as plt
import google.generativeai as genai
import asyncio
import plotly.express as px
from PIL import Image
from googletrans import Translator
import pytesseract
import pickle
import numpy as np
from pathlib import Path
import hashlib
```

Figure 6.3.1 importing the modules

```python
def input_image_setup1(uploaded_files):
    image_parts = []
    for uploaded_file in uploaded_files:
        if uploaded_file is not None:
            image = Image.open(uploaded_file)
            byte_data = uploaded_file.getvalue()
            image_parts.append({
                "mime_type": uploaded_file.type,
                "data": byte_data,
                "image": image
            })
    return image_parts
```

Figure 6.3.2 Uploading the invoices

```python
def extract_key_points(image, prompt):
    key_points_prompt = """
    You are an expert in understanding invoices.
    Please extract the following key points from the invoice:
    - Invoice Number
    - Invoice Date
    - Total Amount
    - Billed To
    - Due Date (if available)
    """
    model = genai.GenerativeModel('gemini-1.5-flash')
    response = model.generate_content([key_points_prompt, image, prompt])
    return response.text
```

Figure 6.3.3 Extracting Key points from the Invoice

```python
def extract_product_details(image, prompt):
    product_details_prompt = """
    Extract the following details about the products in the invoice:
    - Product Name
    - Quantity
    Provide the information in structured English, in the following format:
    Product Name: [Product Name], Quantity: [Quantity],
    product Name: [description],product Name: [item],Quantity: [Qty]
    """
    model = genai.GenerativeModel('gemini-1.5-flash')
    response = model.generate_content([product_details_prompt, image, prompt])
    return response.text
```

Figure 6.3.4 Extracting Product Details

```python
async def detect_language_async(translator, text):
    return await translator.detect(text)

async def translate_text_async(translator, text, src, dest):
    return await translator.translate(text, src=src, dest=dest)

def translate_text(text, target_language):
    language_code = language_mapping.get(target_language, "en")
    async def translate_workflow():
        detected_language = (await detect_language_async(translator, text)).lang
        if detected_language != language_code:
            translated_text = (await translate_text_async(
                translator, text, src=detected_language, dest=language_code
            )).text
            return translated_text
        else:
            return text
    try:
        loop = asyncio.get_event_loop()
        if loop.is_closed():
            asyncio.set_event_loop(asyncio.new_event_loop())
            loop = asyncio.get_event_loop()
        return loop.run_until_complete(translate_workflow())
    except RuntimeError:
        new_loop = asyncio.new_event_loop()
        asyncio.set_event_loop(new_loop)
        return new_loop.run_until_complete(translate_workflow())
```

Figure 6.3.5 Translating Text


```python
async def detect_language_async(translator, text):
    return await translator.detect(text)

async def translate_text_async(translator, text, src, dest):
    return await translator.translate(text, src=src, dest=dest)

def translate_text(text, target_language):
    language_code = language_mapping.get(target_language, "en")
    async def translate_workflow():
        detected_language = (await detect_language_async(translator, text)).lang
        if detected_language != language_code:
            translated_text = (await translate_text_async(
                translator, text, src=detected_language, dest=language_code
            )).text
            return translated_text
        else:
            return text
    try:
        loop = asyncio.get_event_loop()
        if loop.is_closed():
            asyncio.set_event_loop(asyncio.new_event_loop())
            loop = asyncio.get_event_loop()
        return loop.run_until_complete(translate_workflow())
    except RuntimeError:
        new_loop = asyncio.new_event_loop()
        asyncio.set_event_loop(new_loop)
        return new_loop.run_until_complete(translate_workflow())
```


Figure 6.3.6 Categorizing of Invoices

```python
def generate_product_quantity_bar_chart(products, quantities):
    fig, ax = plt.subplots(figsize=(10, 6))
    ax.bar(products, quantities, color='skyblue')
    ax.set_xlabel('Product Name')
    ax.set_ylabel('Quantity')
    ax.set_title('Product Quantities')
    plt.xticks(rotation=45, ha='right')
    st.pyplot(fig)

def generate_product_quantity_pie_chart(products, quantities):
    fig, ax = plt.subplots(figsize=(8, 8))
    ax.pie(quantities, labels=products, autopct='%1.1f%%', startangle=140, colors=plt.cm.Paired.colors)
    ax.set_title('Products and Their Quantities', fontsize=16)
    st.pyplot(fig)
```

Figure 6.3.7 Generating Graphs

```python
with tabs[3]:
    df1 = ""
    for i in range(total):
        status = df.loc[i, "status"]
        category = df.loc[i, "category"]
        highlight_box(f"Category: {category}")
        highlight_box(f"Status: {status}", color="lightgreen" if status == "Paid" else "lightcoral")

        key_response = df['Key Points'].values[i]
        st.subheader(f"Essential Information from Image{i+1}:")

        translated_details_lines = key_response.split("\n")

        # Extracting key details
        for line in translated_details_lines:
            try:
                if "Invoice Number" in line:
                    invoice_number = line.split(":")[1].strip("**")
                elif "Invoice Date" in line:
                    invoice_date = line.split(":")[1].strip("**")
                elif "Total Amount" in line:
                    total_amount = line.split(":")[1].strip("** ₹")
                elif "Due Date" in line:
                    due_date = line.split(":")[1].strip("**")
            except Exception as e:
                print(f"Error processing line '{line}': {e}")
```

```
        df1 = pd.DataFrame([{
            "Invoice Number": invoice_number,
            "Invoice Date": invoice_date,
            "status" : status,
            "Total Amount": total_amount
        }])
        # Displaying the DataFrame

        st.dataframe(df1)

        data_to_store = {
            "Id": invoice_number,
            "date": invoice_date,
            "status": status,
            "amount": total_amount,
        }
        save_to_pickle1(data_to_store)


with tabs[4]:
    col1,col2 = st.columns(2)
    with col1:
        st.subheader('Bar Graphs')
        for i in range(total):
            products = df.loc[i, "products"]
            quantities = df.loc[i, "quantities"]
            if products and quantities:
                generate_product_quantity_bar_chart(products, quantities)
```

```
    with col2:
        st.subheader('pie Graphs')
        for i in range(total):
            products = df.loc[i,"products"]
            quantities = df.loc[i, "quantities"]
            if products and quantities:
                generate_product_quantity_pie_chart(products, quantities)

with tabs[5]:
    df_new = df.drop(columns=['products', 'quantities'])
    st.subheader("Extracted Data:")
    st.table(df_new)

    # Provide a download button for the extracted data
    st.download_button(
        label="Download Extracted Data as CSV",
        data=df.to_csv(index=False),
        file_name="extracted_data.csv",
        mime="text/csv"
    )

    if st.button("Dashboard and Chatbot"):
        st.session_state.page = "Dashboard"
        st.rerun()
```

```python
def chatBot():
    with st.container():
        col1,col2,col3,col4,col5,col6 = st.columns(6)
        with col6:
            col7,col8,col9 = st.columns(3)
            with col9:
                if st.button("⬅ BACK"):
                    st.session_state.page = "Home"
                    st.rerun()
```

Figure 6.3.8 Dashboard

```python
# Function to get AI response
def get_gemini_response(message):
    response = chat.send_message(message)
    return response["content"]

def answer_user_question_directly(image, question):
    prompt = f"Given the invoice image, answer the following question: {question}"
    response = get_gemini_response(prompt)
    return response

# Chatbot Tab
with tabs[2]:
    input_text = st.text_input(
        "Enter your question (e.g., 'What is the invoice date?' or 'Who is the invoice billed to?')",
        key="input",
        placeholder="e.g., Extract invoice date or billing name"
    )
    image_data = input_image_setup1(load['uploaded_files'])
    for idx, image in enumerate(image_data):
        user_answer = answer_user_question_directly(image["image"], input_text)
        if input_text == "":
            continue
        st.subheader(f"Answer to Your Question: {input_text}")
        highlight_box(user_answer, color="lightyellow")
```

Figure 6.3.9 Chatbot Functionality

```python
import streamlit as st
import re
import os
import csv
import pandas as pd
import matplotlib.pyplot as plt
import pdfplumber
import seaborn as sns
import pickle
import numpy as np
from datetime import datetime, timedelta

# Initialize session state
if "page" not in st.session_state:
    st.session_state.page = "Home"

def main():
    if st.session_state.page == "Home":
        home()
    elif st.session_state.page == "pdf":
        pdf_extractor()
    elif st.session_state.page == "image":
        image_extractor()

def home():
    # Load dataset
    df = pd.read_csv("invoices2.csv")

    # Convert 'Invoice Date' to datetime format
    df["Invoice Date"] = pd.to_datetime(df["Invoice Date"], errors="coerce")

    # Streamlit UI
    st.title("Product Insights Dashboard")

    # Step 1: Select Date Range
    date_options = ["Past 7 Days", "Past 30 Days", "Past 60 Days", "Past 90 Days", "All Time"]
```

```python
def home():
    # Load dataset
    df = pd.read_csv("invoices2.csv")

    # Convert 'Invoice Date' to datetime format
    df["Invoice Date"] = pd.to_datetime(df["Invoice Date"], errors="coerce")

    # Streamlit UI
    st.title("Product Insights Dashboard")

    # Step 1: Select Date Range
    date_options = ["Past 7 Days", "Past 30 Days", "Past 60 Days", "Past 90 Days", "All Time"]
    selected_range = st.selectbox("Select Date Range:", date_options)

    # Get the start date based on selection
    end_date = datetime.today()
    if selected_range == "Past 7 Days":
        start_date = end_date - timedelta(days=7)
    elif selected_range == "Past 30 Days":
        start_date = end_date - timedelta(days=30)
    elif selected_range == "Past 60 Days":
        start_date = end_date - timedelta(days=60)
    elif selected_range == "Past 90 Days":
        start_date = end_date - timedelta(days=90)
    else:
        start_date = df["Invoice Date"].min()  # All-time data

    # Filter data by date
    filtered_df = df[(df["Invoice Date"] >= start_date) & (df["Invoice Date"] <= end_date)]

    # Step 2: Select Product Category
    categories = ["Laptop", "Mobile", "Printer", "Tablet", "TV", "AC", "Washing Machine", "Refrigerator"]
    selected_category = st.selectbox("Select a Product Category:", categories)
```

```
fig, ax = plt.subplots(figsize=(12, 6))
sns.barplot(x=product_counts.index, y=product_counts.values, palette="viridis", ax=ax)
ax.set_xlabel("Product Models")
ax.set_ylabel("Number of Invoices")
ax.set_title(f"{selected_category} Sales Count ({selected_range})")
ax.set_xticklabels(product_counts.index, rotation=90)

# Show plot in Streamlit
st.pyplot(fig)

# Step 4: Select Specific Product
selected_product = st.selectbox("Select a Product:", product_counts.index)

# Get product details
product_info = category_df[category_df["Product_Name"] == selected_product].iloc[0]

# Step 5: Display Product Details
st.subheader(f"Details of {selected_product}")
st.write(f"**Product Name:** {selected_product}")
st.write(f"**Description:** {product_info['Description']}")
st.write(f"**Usage:** {product_info['Usage']}")
st.write(f"**How to Use:** {product_info['How_to_Use']}")

# Step 6: Calculate Demand Percentage
selected_product1 = product_counts.get(selected_product, 0)
percentage = ((selected_product1 / max_count) * 100).astype(int)
st.subheader(f"Demand of {selected_product}: {percentage}%")
st.progress(percentage / 100)
```

Fig 6.3.10 code for Product demand, product frequency

6.4 SUMMARY:

Hence, this chapter clearly explains all the implementation steps required to dothis project. We have implemented the model, now we have to test the model under various testcases.

CHAPTER 7
SYSTEM TESTING

7. TESTING

7.1 INTRODUCTION

Testing is carried out on a fully developed system to identify whether the systemworks according to the requirements specified and also to identify and correct the errors. It canbe either done manually or using automated tools.

7.1.1 Goals of Software Testing

- **Verification:** It allows testers to confirm that the software meets the various business and technical requirements specified before the development of the project.
- **Validation**: Confirms that the software performs as expected and as per requirements of the clients after the development of the project. It involves comparing thefinal output with the expected output and then making necessary changes if there is a difference.
- **Defects**: The important purpose of testing is to find different defects in the software to prevent its failure or crash during implementation. Defects if left undetected or unattended can harm the functioning of the software.
- **Transparency**: With the assistance of reports generated during the process of software testing, testers can accumulate a variety of information related to the software and the steps taken to prevent failure.
- **Quality Analysis**: Testing helps improve the quality of software by constantly verifying design, code and outcome.
- **Compatibility**: It helps to validate application's compatibility with the implementation environment, various devices, operating systems, user requirements among other things.
- **Performance**: It helps to ensure that system performs efficiently by testing under different scenarios.

7.2 Testing Methods

- **Static testing**: It is also known as Verification in Software testing. Verification is the processcarried out before the development of the project to verify whether the project is being developed according to the requirements or not.
- **Dynamic testing**: It is also known as Validation in Software testing. Validation is the processcarried out after the development of the project to confirm that the project is developed according to requirements or not.

7.3 Testing Approaches

There are three types of testing approaches

- **White Box testing**: It is also called as Glass Box/Clear Box/Structural testing. White Box testing is based on application's internal code structure. In White Box testing, an internal perspective of the system, as well as programming skills is used to design test.
- **Black Box testing**: It is also called as Behavioral / Specification-Based/Input Output Test It is a software testing method in which testers evaluate the functionality of the software undertest without looking at the internal code structure.

Figure 7.3 White-Box and Black Box Testing

7.3.1 Testing Levels

- **Unit testing**: It is done to check whether the individual modules of the source code are working properly i.e., testing each and every unit of the application separately by the developer in the developer's environment.
- **Integration testing**: Integration testing is the process of testing the connectivity or data transfer between a couple of units tested modules. It is also called String testing. It is sub divided into Top-Down approach, Bottom- Up approach and Sandwich approach.
- **System testing**: It is also called end to end testing. It is Black Box testing. Testing the fully integrated application this is also called as end to end scenario testing. Verify testing of every input in the application to check for desired outputs
- **Acceptance testing**: To obtain customer sign-off so that software can be delivered and payments received. Types of Acceptance testing are Alpha, Beta and gamma testing.

7.3.2 Testing Strategy

The testing strategy employed is **Functional testing**.Functional testing is a type of Software testing that validates the software system against the functional requirements/specifications

7.4 Sample Test Cases

Test Case 1: Unit Testing in User Authentication Module

Input: User enters valid username and password and clicks Login

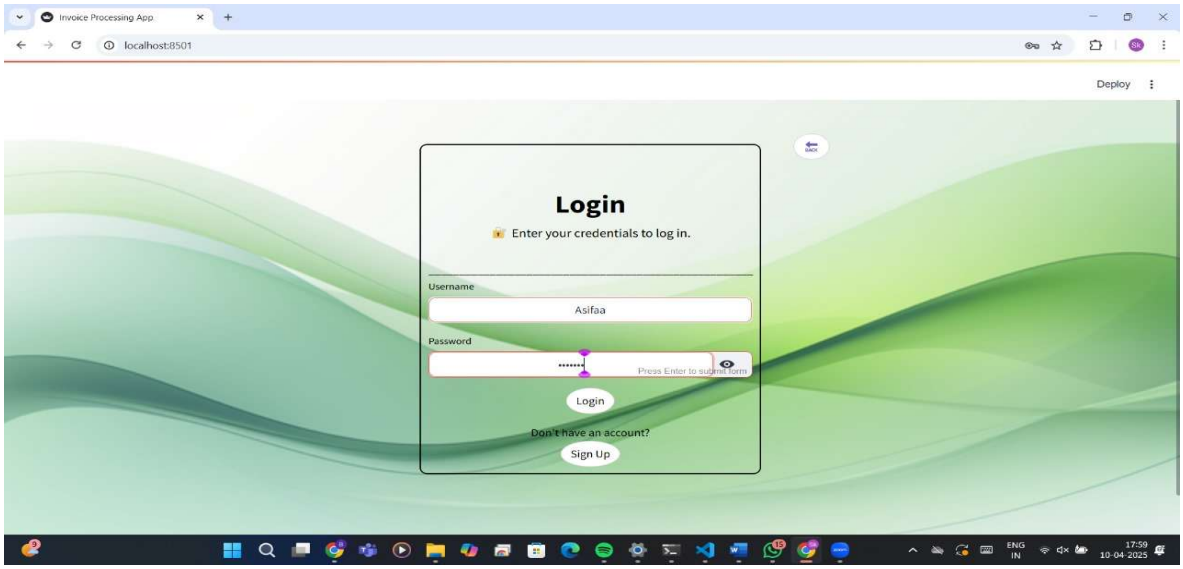


Figure 7.4.1 : User entering valid credentials

Expected Output: System shows a pop-up message like Login Successful.

Observed Output: Pop-up message displayed login Successful

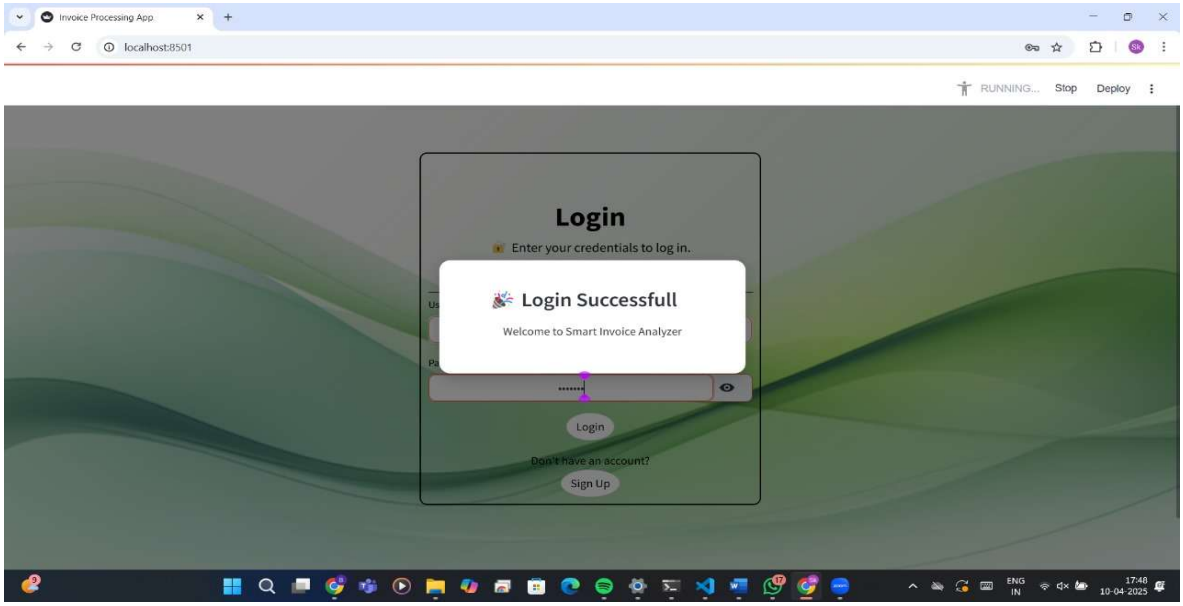


Figure 7.4.2 : Pop up message showing Sucessful Login

Test Case 2: Validation Testing in User Authentication Module

Input: User enters invalid username and password and clicks Login

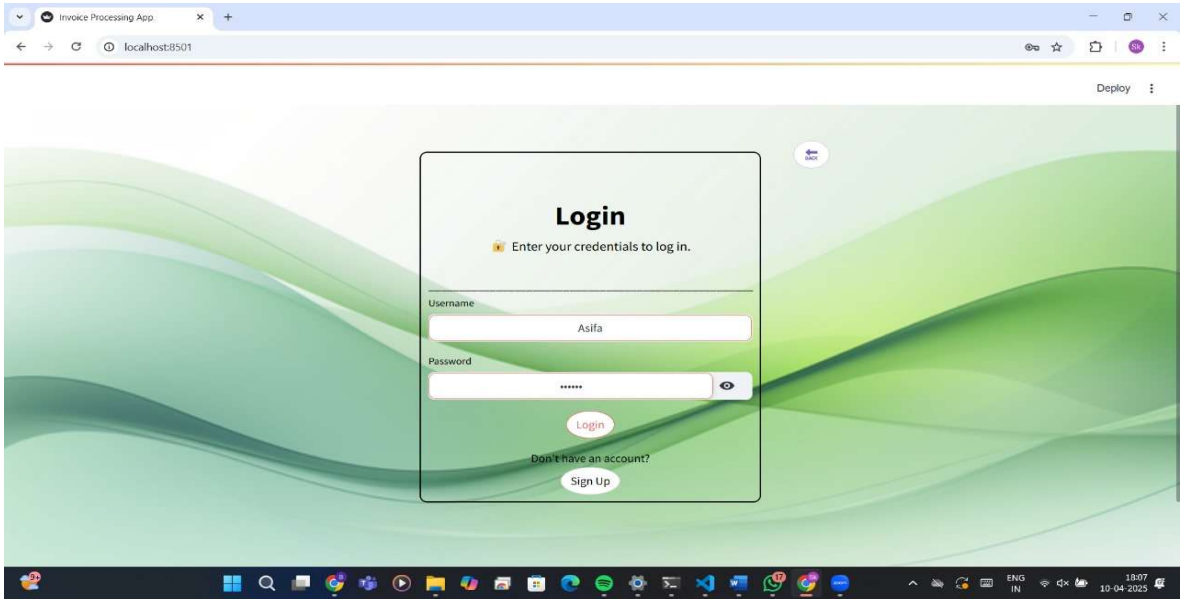


Figure 7.4.3 : User entering Invalid Credentials

Expected Output: System shows an alert message invalid username or password

Observed Output: Alert message displayed Sucessfully

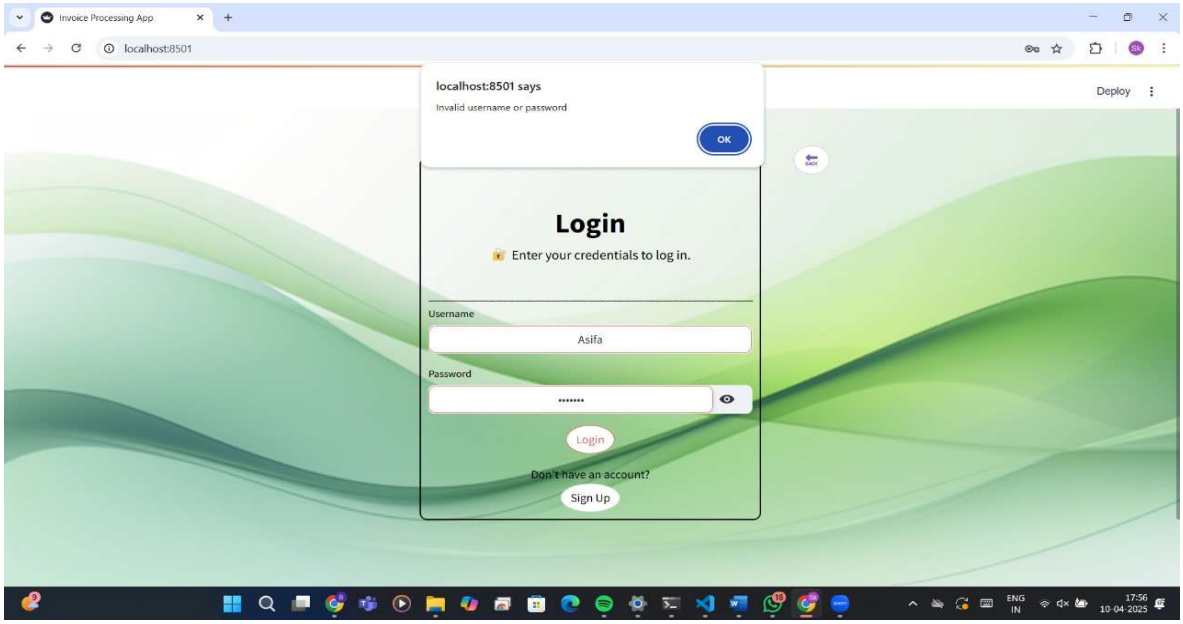


Figure 7.4.4 : Alert message showing invalid username or password

Test Case 3: Validation Testing on Invoice Upload Module

Input: User uploads a non invoice file

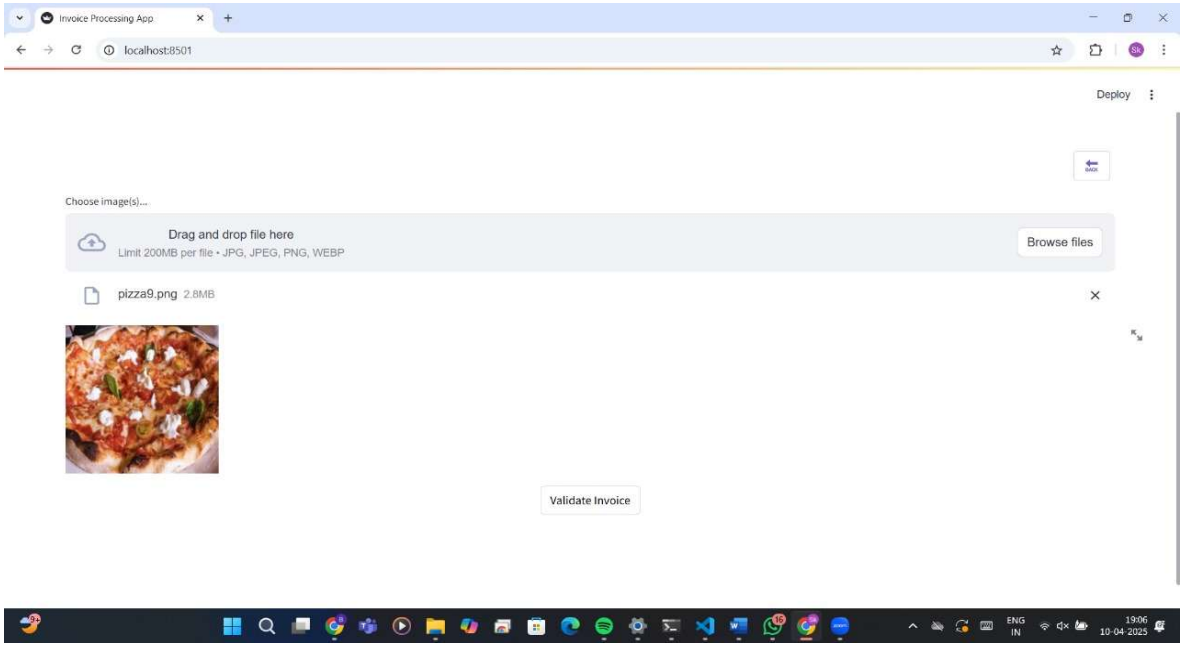


Figure 7.4.5 : Uploading an invalid invoice

Expected Output: System flags an error message such as please upload a valid Invoice

Observed Output : System Displays Error message please upload a valid invoice

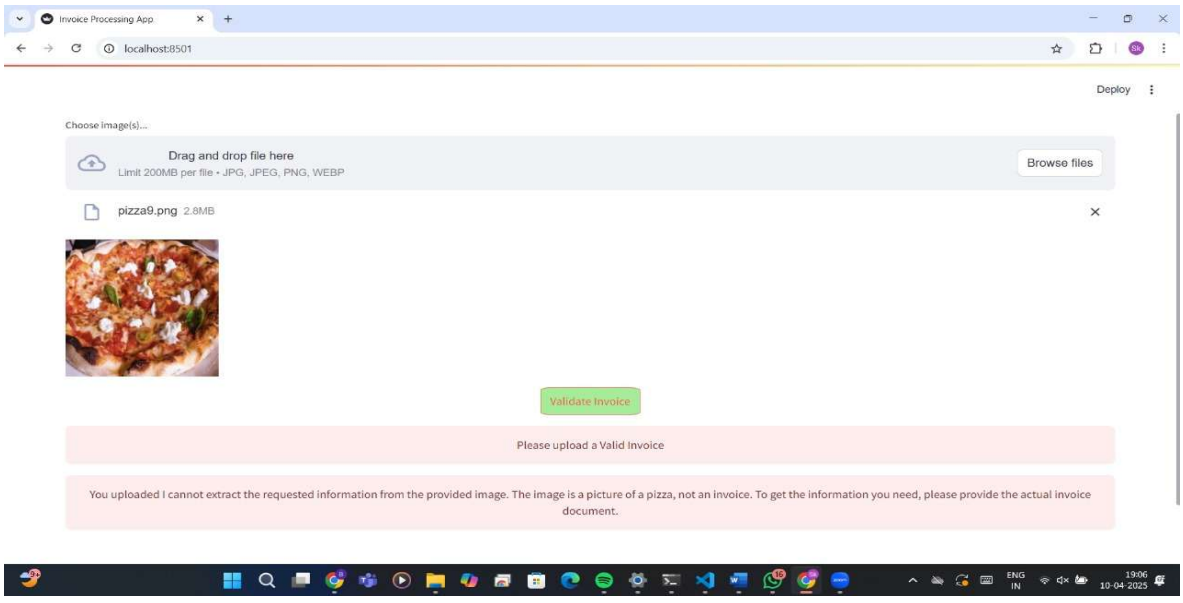


Figure 7.4.6 : System flags an error message

Test Case 4: Unit Testing in Fraud Detection Module

Input: Tampered invoice data is passed to the SVC model

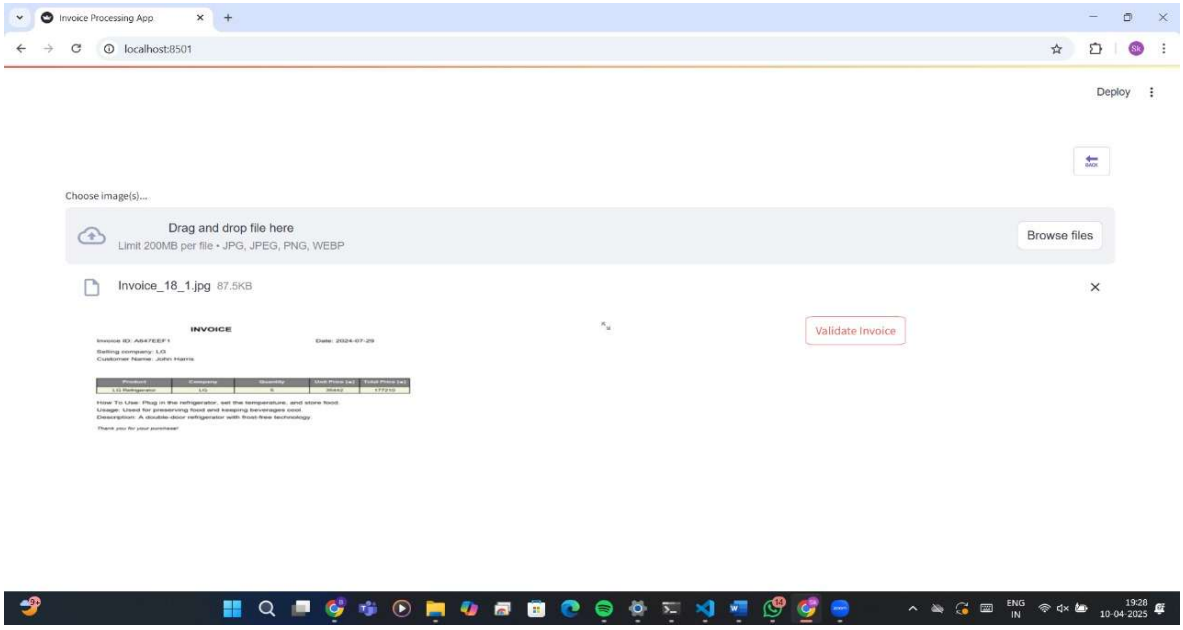


Figure 7.4.7 : Uploading an Tampered Invoice

Expected Output: Model classifies the invoice as Fraudlent

Observed Output : Model Classified the invoice as Fraudlent

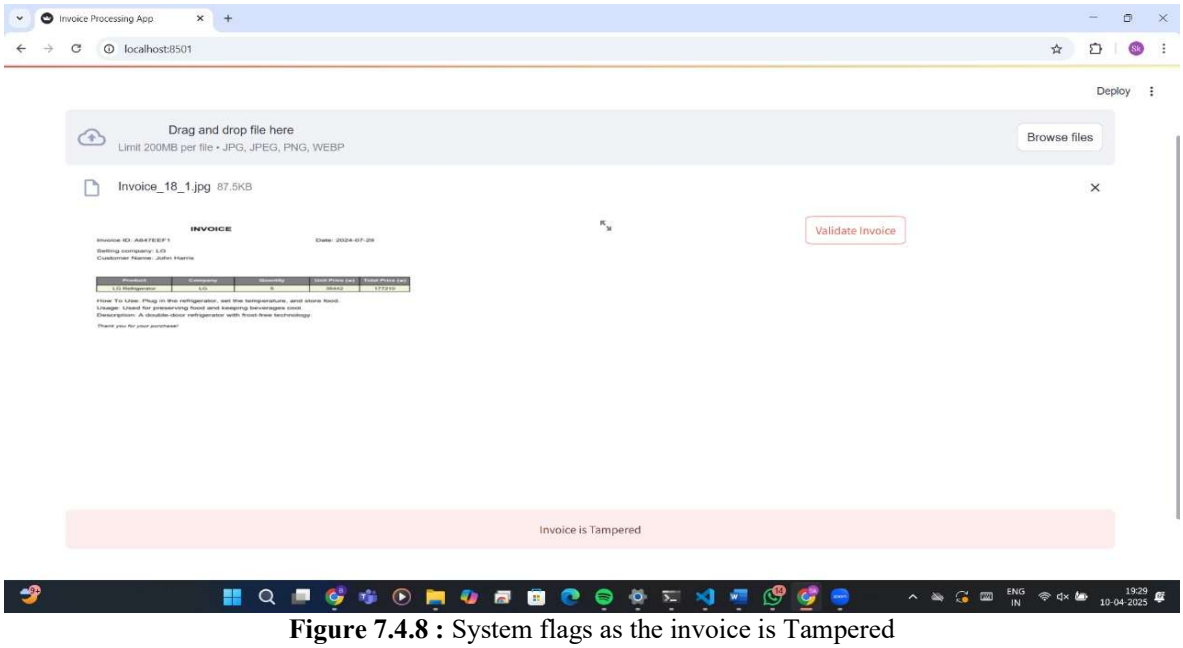


Figure 7.4.8 : System flags as the invoice is Tampered

Test Case 5: Integration Testing on Invoice Upload and Extraction Module, Fraud Detection Module

Input: User uploads a valid invoice (PDF/Image)

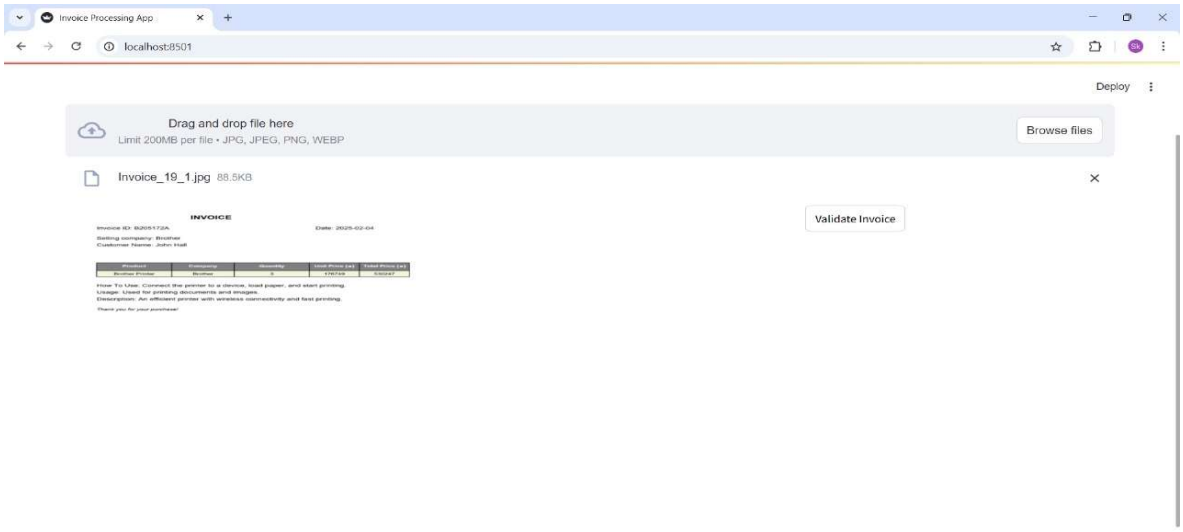
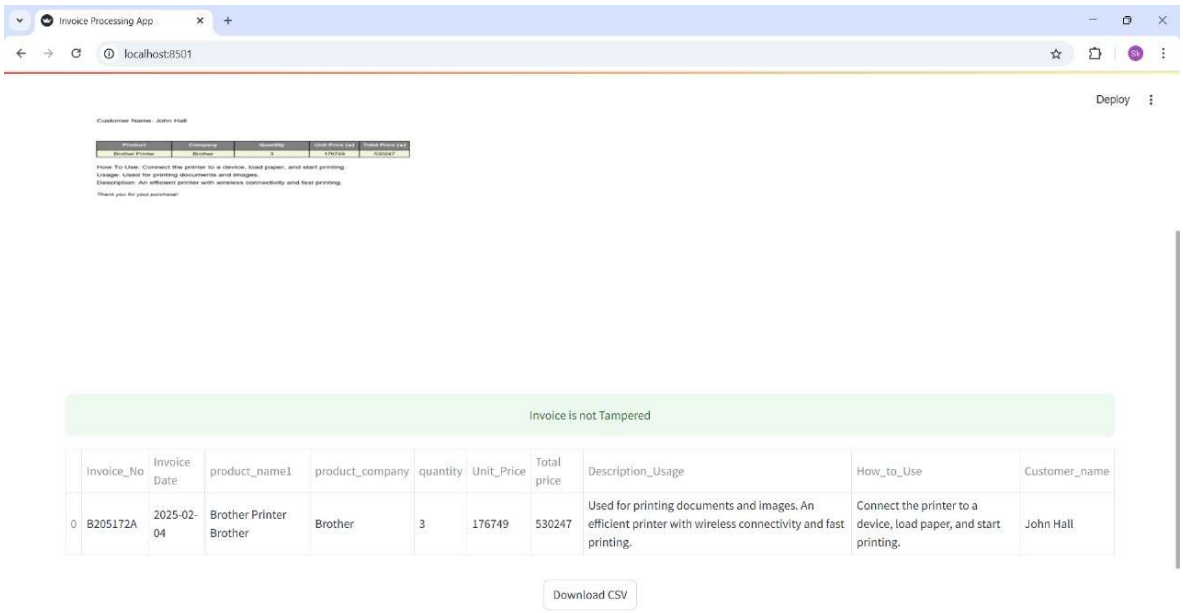


Figure 7.4.9 : Uploading an Valid Invoice

Expected Output: System displays the extracted invoice fileds and shows the fraud status as either tampered or not tampered

Observed Output : Extracted invoice Fields and shows the fraud status as not tampered



	Invoice_No	Invoice Date	product_name1	product_company	quantity	Unit_Price	Total price	Description_Usage	How_to_Use	Customer_name
0	B205172A	2025-02-04	Brother Printer Brother	Brother	3	176749	530247	Used for printing documents and images. An efficient printer with wireless connectivity and fast printing.	Connect the printer to a device, load paper, and start printing.	John Hall

Figure 7.4.10 : System extracts the data and flags as Invoice is not Tampered

Test Case 6: Integration Testing on Product Insights Module, Data Visualization Module

Input: User selects a product (e.g., "Laptop") from the available dropdown and chooses a time range (e.g., "past 30 days") to view frequency trend.

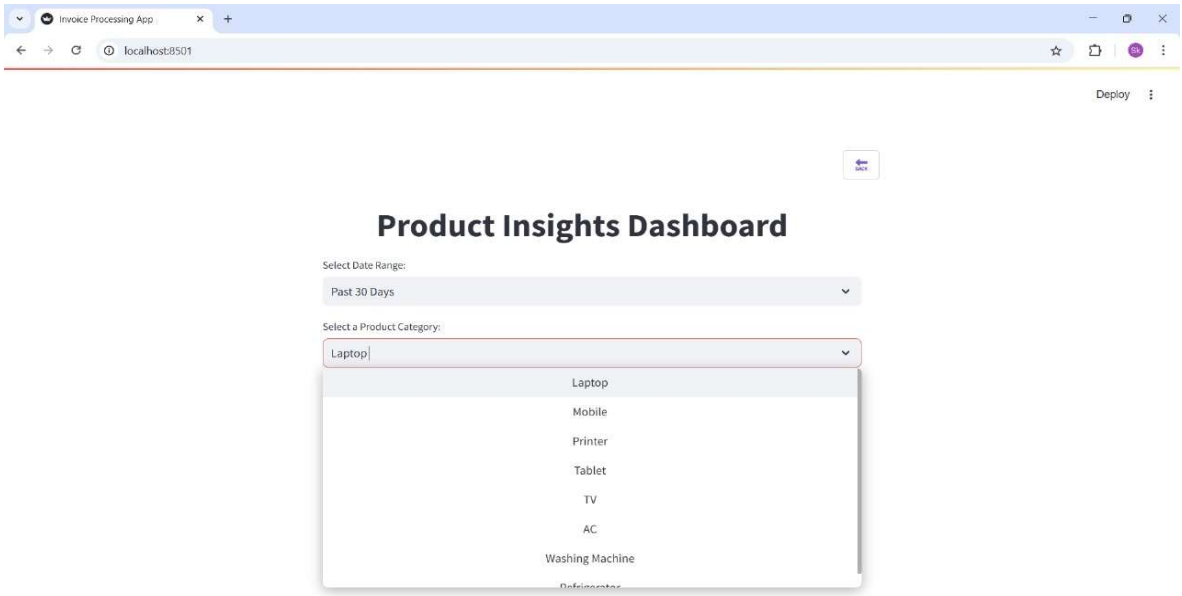


Figure 7.4.11 : Product and time range selected

Expected Output: The system displays a bar graph showing how frequency the selected product appeared in invoices during the chosen period.

Observed Output : System displayed a bar graph showing frequency of the selected product

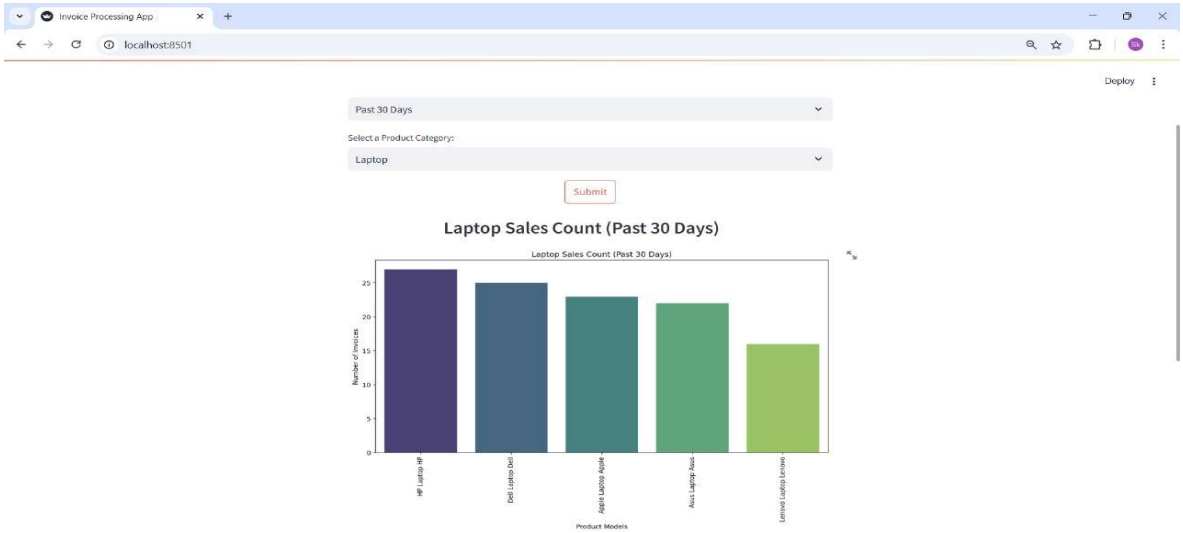


Figure 7.4.12 : Bar graph showing Laptop Sales Count

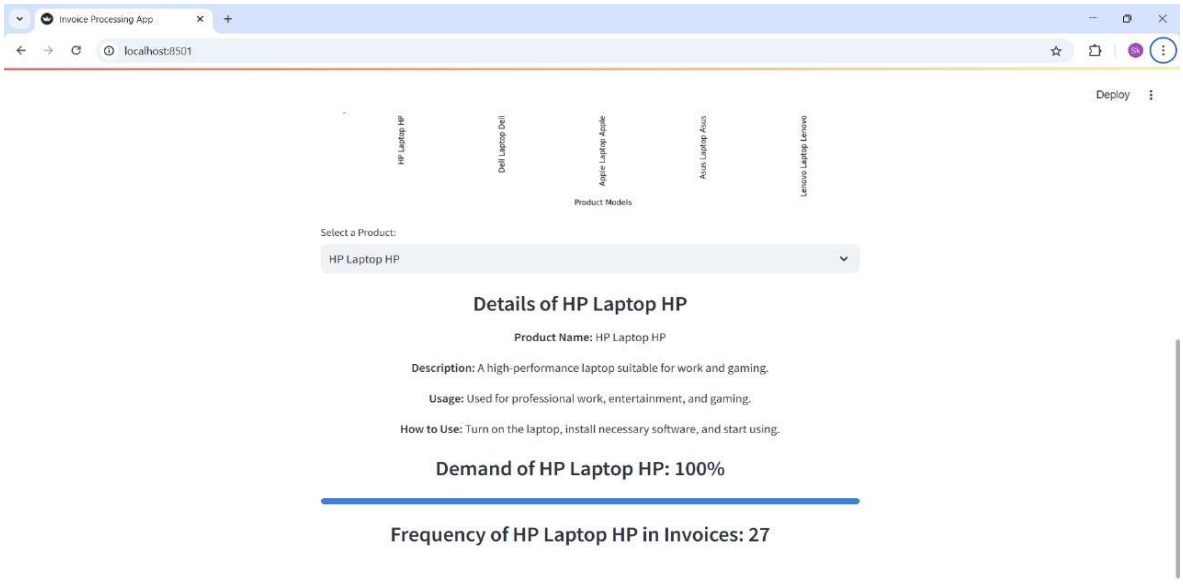


Figure 7.4.13 : Showing Details,Demand and Frequency of Selected product

7.5 SUMMARY

In this chapter, we have listed and executed various test cases for fraud detection, multi-language invoice extraction, product insights, and invoice generation in our Smart Invoice AnalyzerTool. The system was tested using a custom dataset and successfully extracted, analyzed, and categorized invoice data across different scenarios. All test cases were successfully passed, ensuring the system's accuracy and efficiency in handling multilingual invoices and detecting fraudulent ones.

CHAPTER 8
RESULTS

8. RESULTS

The Smart Invoice Analyzer Tool efficiently automates invoice processing by extracting details, detecting fraud, and generating summaries. It provides product insights, supports multilingual translation, and enables new invoice generation. AI-powered OCR ensures accuracy, while fraud detection enhances security Visualization features offer valuable business insights. Overall, the tool streamlines invoice management, improving efficiency and reliability.

8.1 USER INTERFACE

1. Initially the application will look like this with signin and signup option

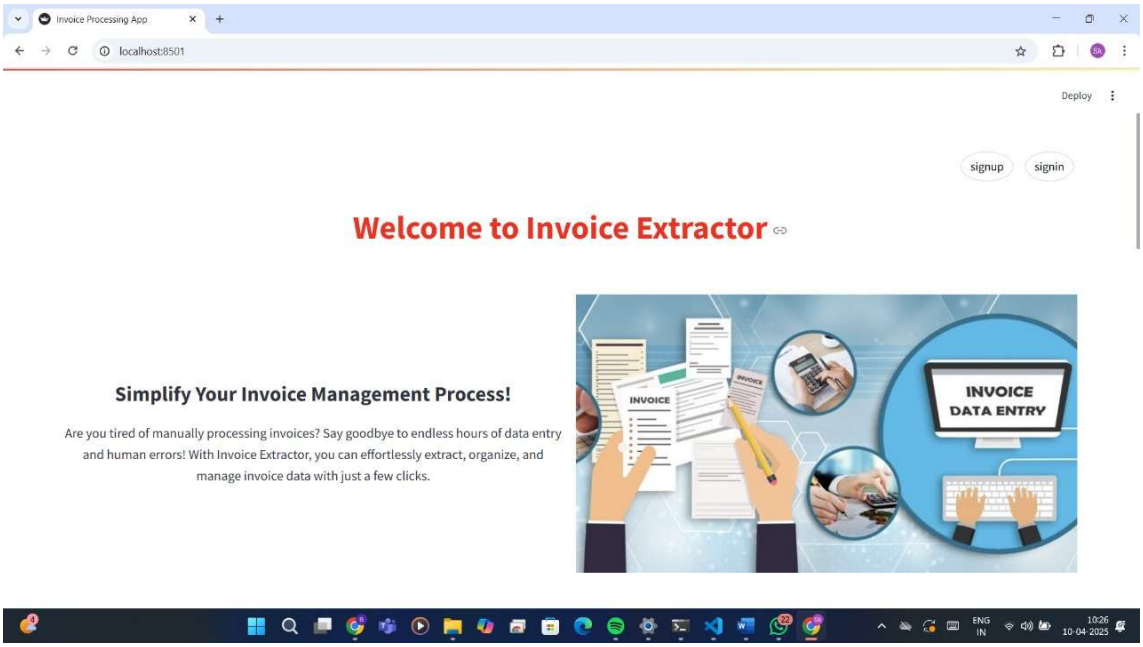


Figure 8.1.1 : Web Interface of Smart Invoice Analyzer

User can signup with their credentials or can create new account by clicking on signin button

2. After logging in the application will look like this where users can easily navigate through different functions of the Smart Invoice Analyzer.

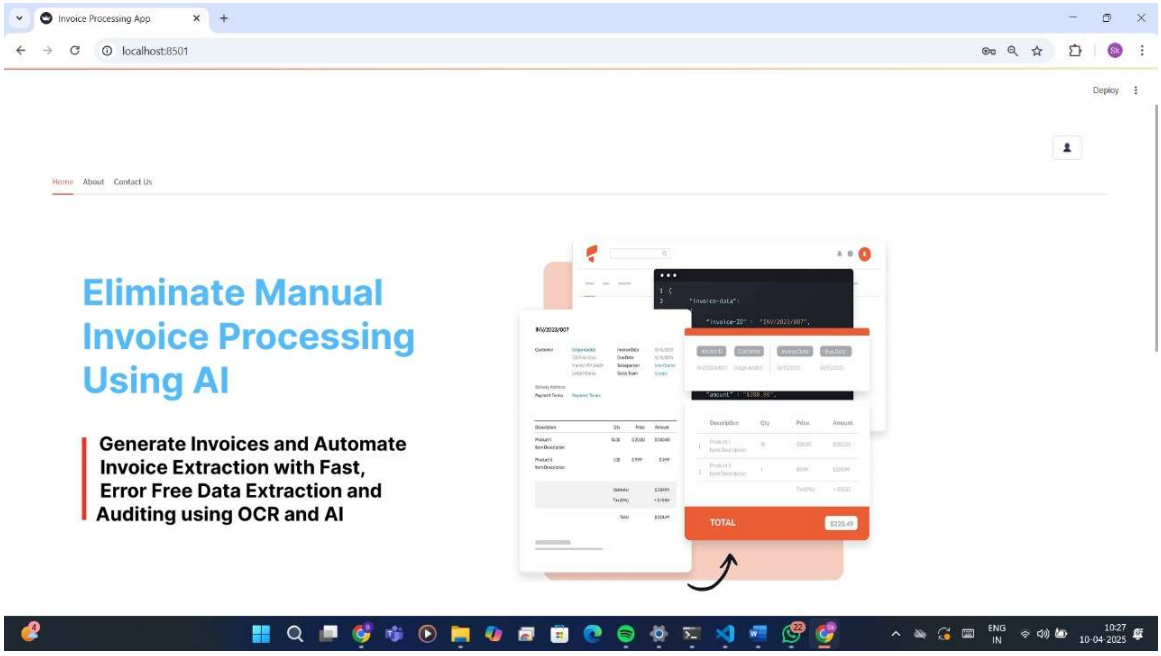


Figure 8.1.2: User Interface of Home page

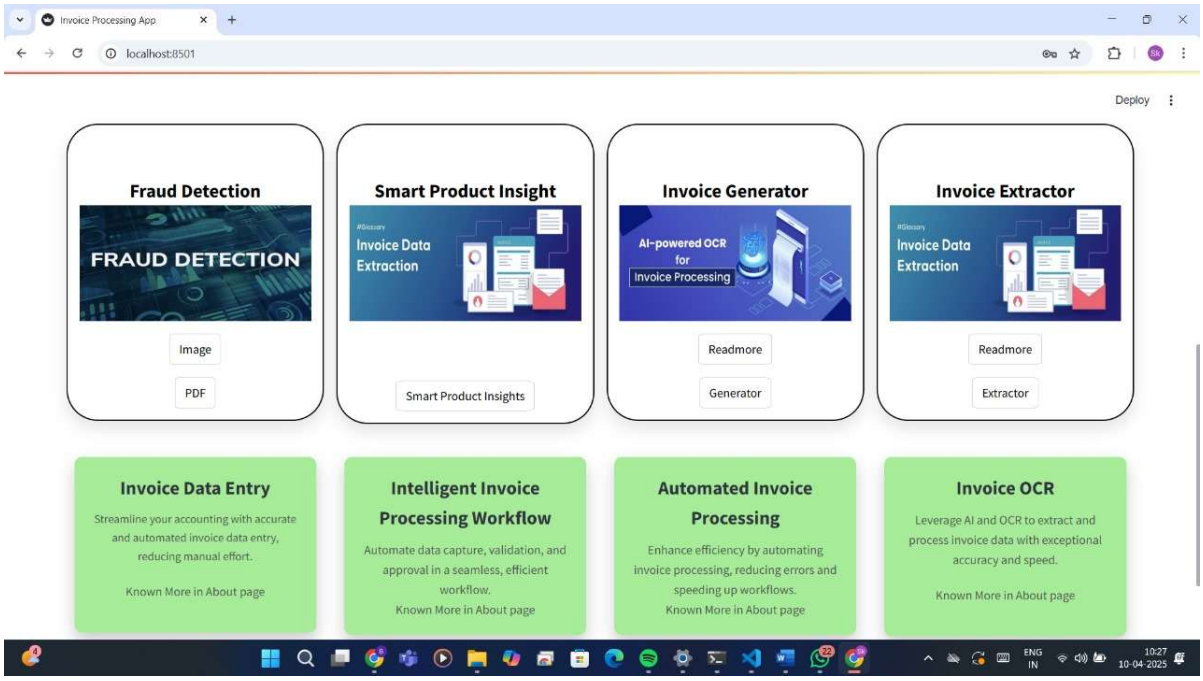


Figure 8.1.3 : Core Functionalities of Application

8.2 Fraudulent Invoice Detection

1)User can upload Image / PDF to detect Tampered Invoices

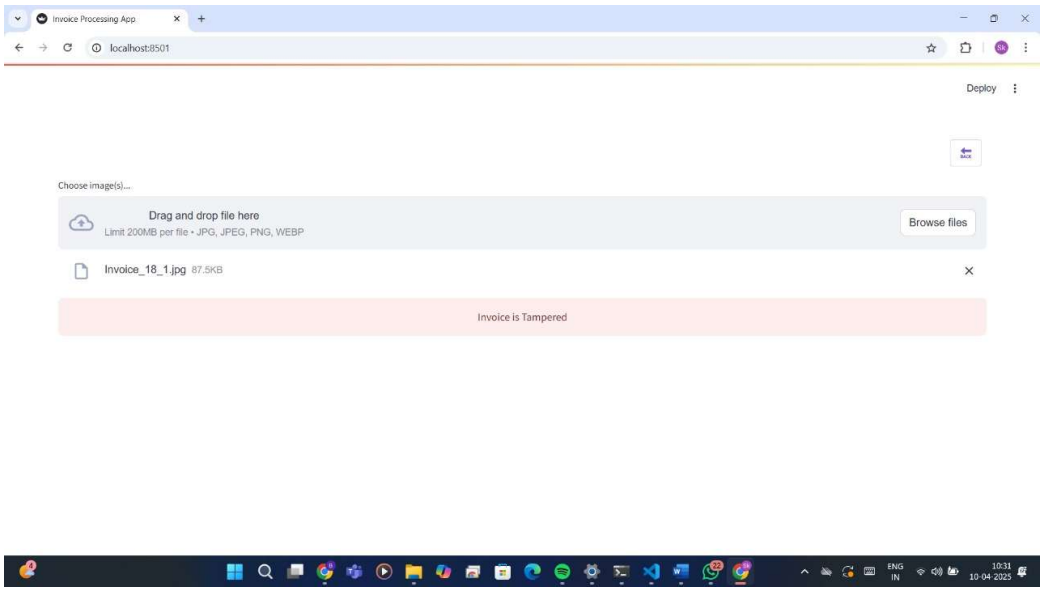


Figure 8.2.1 : Tampered Invoice Detection

2) User can upload untampered invoice to get the summary of the invoice.

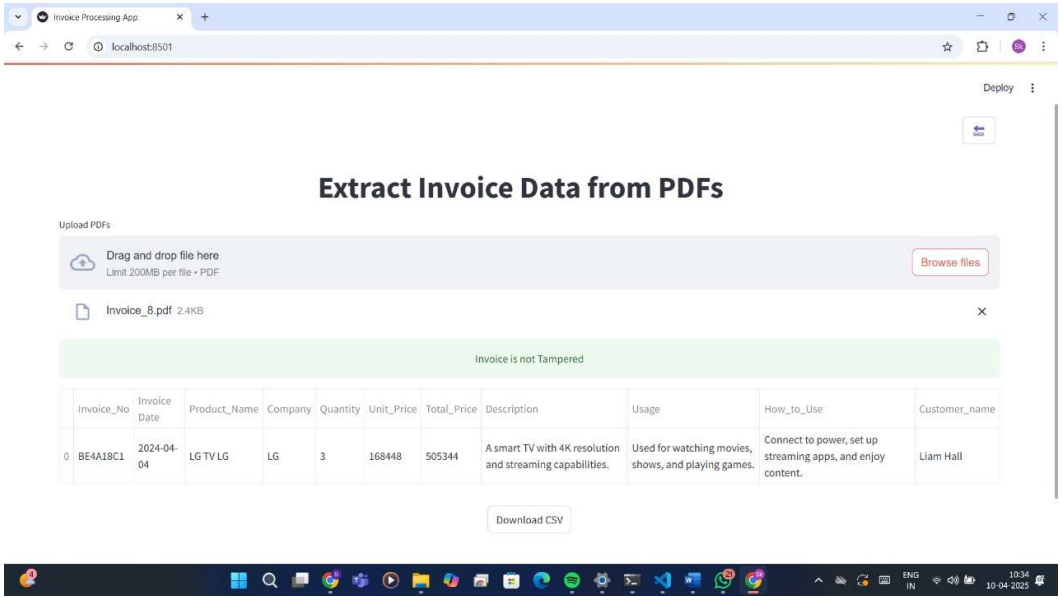


Figure 8.2.2: Untampered Invoice Summary

8.3 Product Insights

User can select a product from the available products and can able to know the demand and frequency of the product based on certain time period.

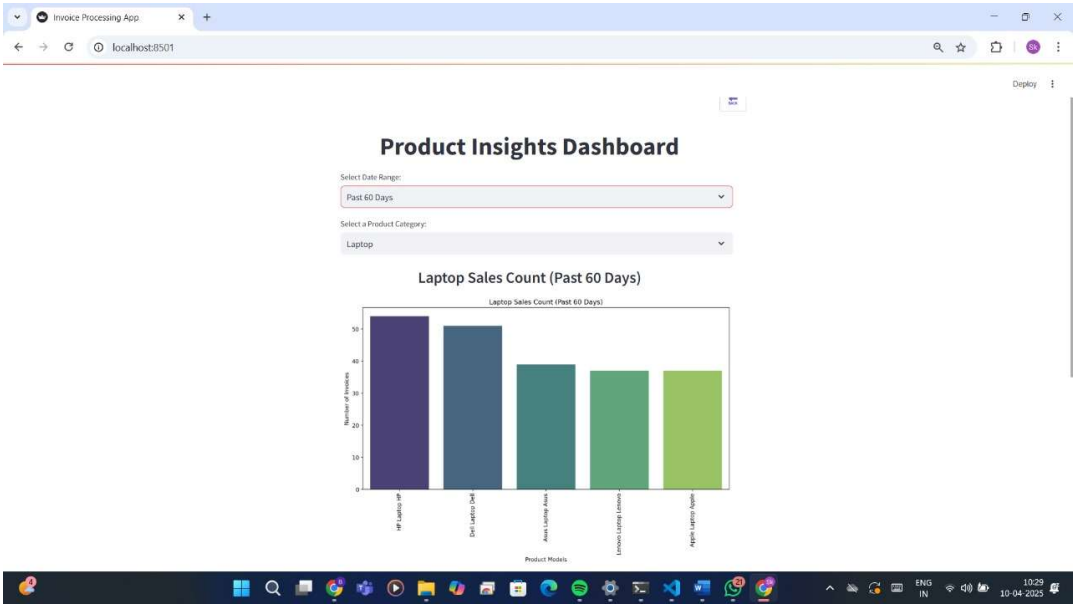


Figure 8.3.1 : Product Insights Dashboard with Sales Count Visualization of laptop

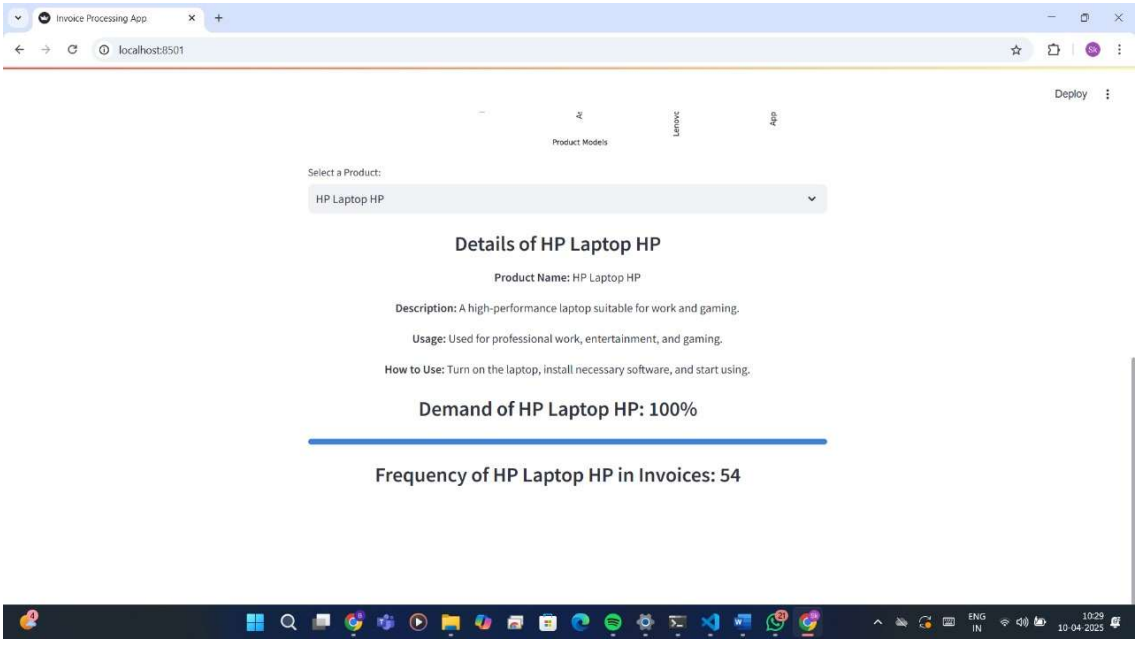


Figure 8.3.2 : Details, Demand and Frequency of Selected Product

8.4Multilanguage Invoice Extraction

User can upload invoices in multiple languages and extract essential information in English language from them

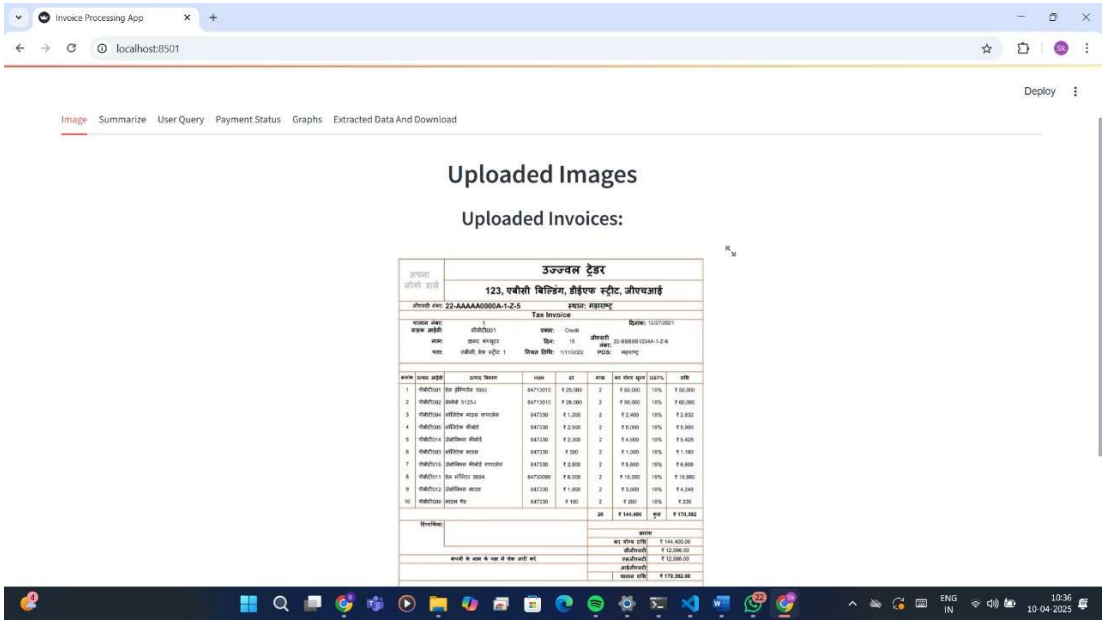


Figure 8.4.1 : Hindi Invoice Uploaded

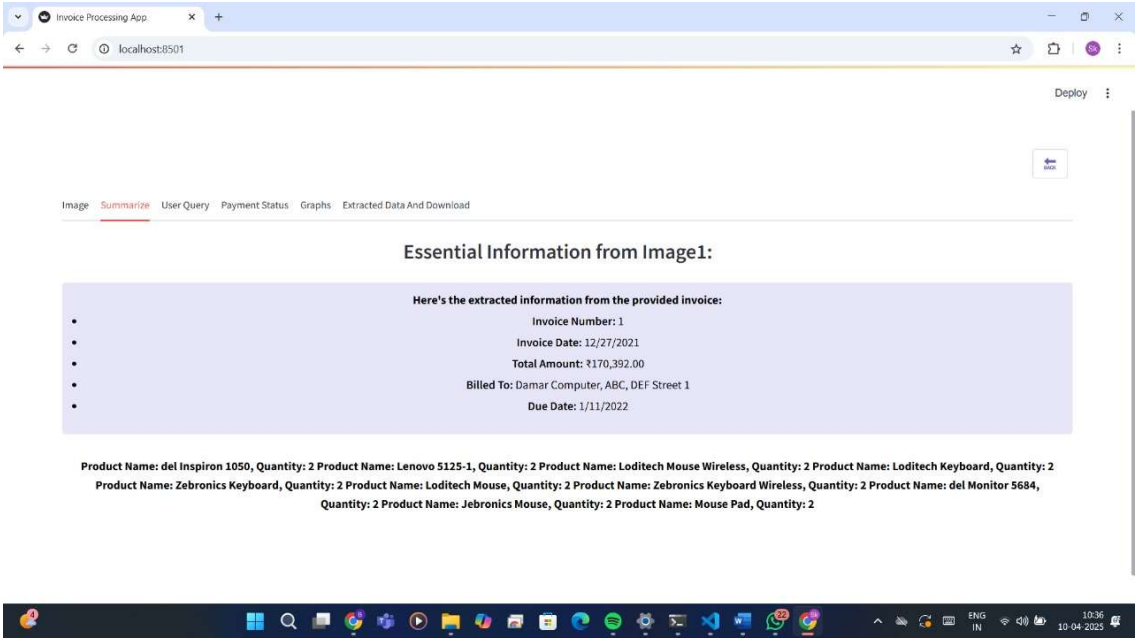


Figure 8.4.2 : Translated Text into English

8.5Dashboard Visualizations

User can able to view all invoice data at a glance and know the status of invoices

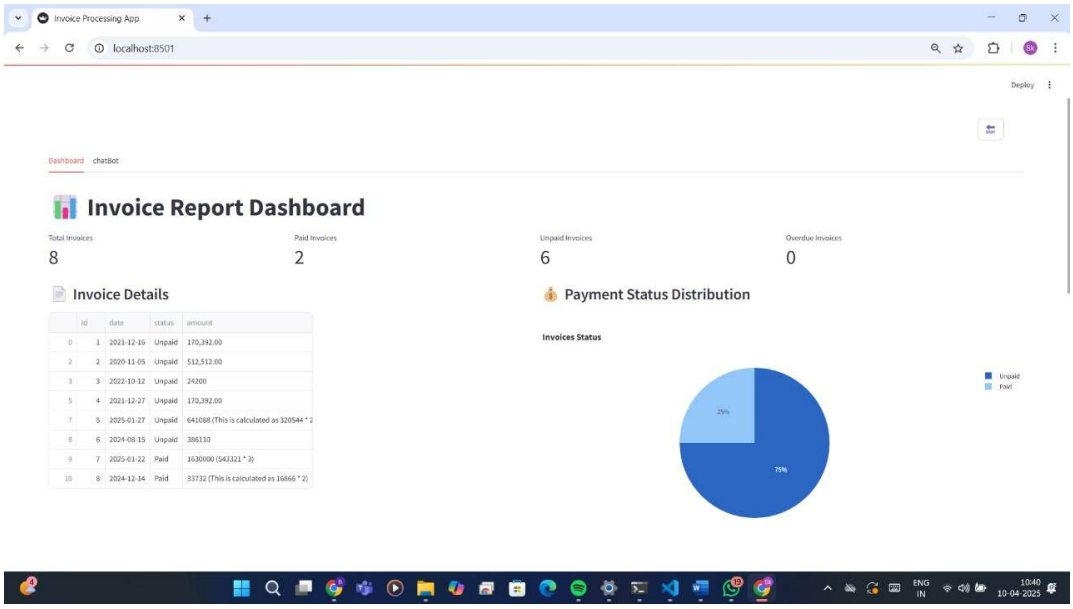


Figure 8.5.1 : Invoice Dashboard

Figure 8.5.2 : Paid and Unpaid invoice Visualization

8.6 InfoMate AI

Here user can give queires related to products.

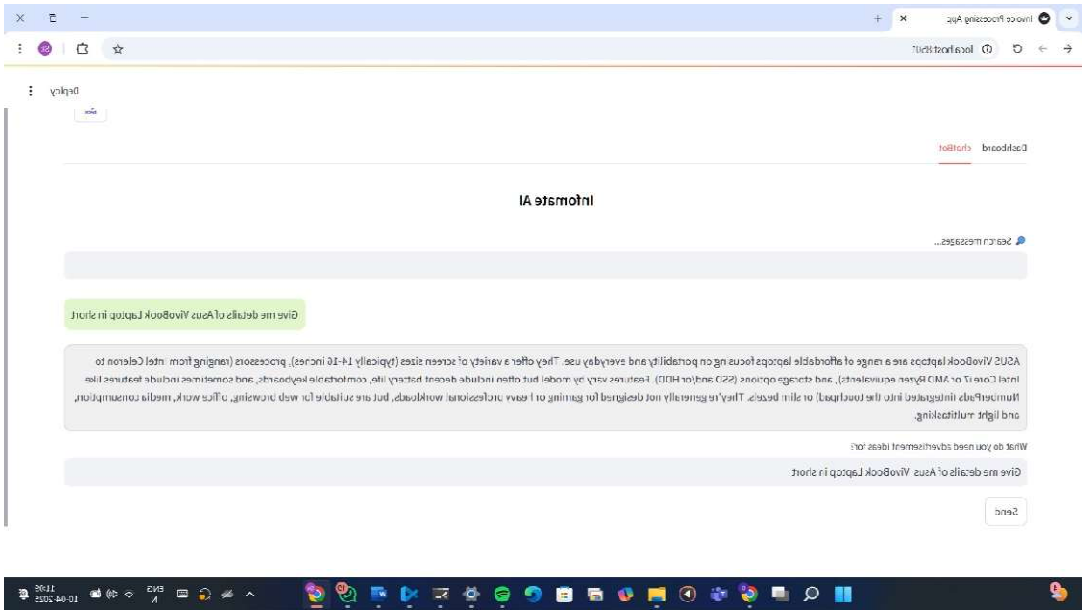


Figure 8.6 : Chatbot response to related query

8.7 SUMMARY

In this chapter, we have listed out different results of our application. We have listed all the possible results that are expected from our application.

CHAPTER 9
CONCLUSION AND FUTURE WORK

CONCLUSION

This project introduces an AI-powered Multi-Language Invoice Extraction, Fraud Detection, and Product Insights System, designed to automate and enhance invoice management with 90% accuracy and an MSE of 0.15. Leveraging OCR, Gemini API, and machine learning, the system extracts essential details from English and multilingual invoices, identifies fraudulent transactions using a custom-trained dataset, and classifies them into relevant categories. The product insights module provides a comprehensive analysis of product demand and sales frequency over different time periods, helping businesses track market trends. Additionally, the invoice generation module dynamically creates structured invoices using ReportLab, while the dashboard offers a visual summary of uploaded invoices, including the number of paid and unpaid invoices and their financial status. An interactive chatbot assists users by answering queries related to product details, invoice classification, and demand trends, improving overall user engagement. Developed using Streamlit, this system provides a scalable, efficient, and intelligent approach to invoice processing, making it a valuable asset for businesses looking to automate financial workflows and enhance decision-making.

FUTURE WORK

The Smart Invoice Analyzer Tool will expand its multi-language processing by integrating advanced NLP techniques, allowing support for additional languages. Efforts will also focus on improving OCR accuracy for handwritten and low-quality invoices. A language detection feature will be introduced to automatically identify and process invoices in different languages.To enhance fraud detection, deep learning models like LSTMs and autoencoders will be integrated to detect fraudulent patterns more effectively. Expanding the dataset with diverse fraudulent invoice samples will improve model accuracy. Additionally, real-time fraud detection will provide instant alerts upon invoice upload, enabling quick action.Predictive product demand analysis will utilize machine learning algorithms such as ARIMA, XGBoost, and Prophet to forecast trends. Seasonal analysis will help predict high- demand periods for specific product categories. A recommendation system will be developed to suggest inventory restocking based on predicted demand. The dashboard will offer interactive visualizations like heatmaps and line graphs for better data representation. Dynamic filters will allow users to analyze invoices based on parameters like time, category, or region. Automated reports and insights will help businesses make informed decisions efficiently. For enterprise integration, APIs will be developed to connect with ERP solutions like SAP, Oracle ERP, and Microsoft Dynamics. This will automate invoice reconciliation with purchase orders, streamlining financial workflows. Ensuring compatibility with ERP systems will make invoice processing more efficient.

CHAPTER 10
REFERENCES

REFERENCES

1. https://medium.com/%40prashantmalge181/building-an-multi-language-invoice-extractor-with-google-gemini-pro-free-llm-5749bab03592
2. https://netraneupane.medium.com/invoice-extraction-bot-using-streamlit-and- gemini-233fdf52f130
3. https://www.researchgate.net/publication/376371175_Data_extraction_from_scanned_invoice_documents_in_multiple_languages
4. https://ieeexplore.ieee.org/document/10807209
5. https://medium.com/analytics-vidhya/invoice-information-extraction-using-ocr- and-deep-learning-b79464f54d69
6. https://github.com/clovaai/cord
7. https://www.researchgate.net/publication/328313379_Baidu_Meizu_Deep_Learning_Competition_Arithmetic_Operation_Recognition_Using_End-to-End_Learning_OCR_Technologies
8. https://www.sciencedirect.com/science/article/abs/pii/S1568494622002897
9. https://www.researchgate.net/publication/353792830_Guided_Table_Structure_Recognition_Through_Anchor_Optimization
10. https://arxiv.org/abs/2208.11203
11. https://www.researchgate.net/publication/339028094_Table_Detection_in_Invoice_Documents_by_Graph_Neural_Networks
12. https://www.ijrte.org/portfolio-item/a0504032113/
13. https://dl.acm.org/doi/10.1145/3347320.3357694
14. https://www.researchgate.net/publication/342975723_Evaluation_of_Neural_Network_Classification_Systems_on_Document_Stream
15. https://ieeexplore.ieee.org/Xplore/login.jsp?url=%2Fielaam%2F34%2F7116666%2F6945320-aam.pdf&authDecision=-203
16. https://aclanthology.org/2020.lrec-1.569/

17. https://www.researchgate.net/publication/351406141_An_Integrated_Approach_of_Deep_Learning_and_Symbolic_Analysis_for_Digital_PDF_Table_Extraction

18. https://www.researchgate.net/publication/220887997_pdf2table_A_Method_to_Extract_Table_Information_from_PDF_Files

19. https://www.researchgate.net/publication/336288174_PubLayNet_Largest_Dataset_Ever_for_Document_Layout_Analysis

20. https://aclanthology.org/W03-0419/

21. https://link.springer.com/article/10.1007/BF02626999

22. https://www.researchgate.net/publication/320247182_Named_Entity_Recognition_with_Gated_Convolutional_Neural_Networks

23. https://www.researchgate.net/publication/258340860_Pattern-Based_Approach_to_Table_Extraction

24. https://www.researchgate.net/publication/322783139_DeepDeSRT_Deep_Learning_for_Detection_and_Structure_Recognition_of_Tables_in_Document_Images

25. https://dl.acm.org/doi/10.1007/s11042-021-11819-7

CHAPTER 11

BASE PAPER

Received 10 January 2024, accepted 27 January 2024, date of publication 31 January 2024, date of current version 9 February 2024. Digital Object Identifier 10.1109/ACCESS.2024.3360528

SURVEY

An Overview of Data Extraction From Invoices

THOMAS SAOUT, FRÉDÉRIC LARDEUX, AND FRÉDÉRIC SAUBION
LERIA, SFR MATHSTIC, University of Angers, 49000 Angers, France
Corresponding author: Thomas Saout (thomas.saout@etud.univ-angers.fr)

This work was supported by KS2 Company.

ABSTRACT This paper provides a comprehensive overview of the process for information retrieval from invoices. Invoices serve as proof of purchase and contain important information, including the date, description, quantity, and the price of goods or services, as well as the terms of payment. Companies must process invoices quickly and accurately to maintain proper financial records. To automate this workflow, commercial systems have been developed. Despite the complexity involved, realizing automated processing of invoices necessitates the harmonious integration of a wide range of techniques and methods. While several surveys have shed light on different aspects of this workflow, our objective in this paper is to present a synthetic view of the process and emphasize the most pertinent challenges. We discuss the digitalization of invoices and the use of natural language processing techniques to extract relevant information. We also review machine learning and deep learning techniques that are widely used to handle the variability of layouts, minimize end-user tasks, and train and adapt to new contexts. The purpose of this overview is not to evaluate various systems and algorithms, but rather to propose a survey that reviews a wide scope of techniques for different data extraction tasks, addressing both information extraction and structure recognition for invoice processing. Specifically, we focus on table processing, paying particular attention to graph-based approaches.

INDEX TERMS Invoice processing, table recognition, information extraction.

I. INTRODUCTION

Invoices are crucial documents for companies as they serve as proof of purchase and are necessary for accounting and tax purposes. They are created by the seller and sent to the buyer to request payment for goods or services. Invoices typically contain essential information such as the purchase date, the description of goods or services, the quantity and price, and the payment terms. Companies need to process invoices promptly and accurately to maintain proper financial records and avoid potential payment delays. Digitizing invoices can help streamline the process and reduce the risk of errors. Paper invoices can be converted into a digital format, and automated systems can extract critical information like invoice numbers, amounts, and dates. This approach can speed up processing time and improve accuracy. Furthermore, digital invoices can be easily stored and accessed through document management systems, making it simpler to keep track of them and retrieve them when needed.

The associate editor coordinating the review of this manuscript and approving it for publication was Gustavo Olague.

The process of automated invoice processing requires the handling of several document characteristics, such as varying formats and layouts of invoices, differences in language and terminology, and errors or inaccuracies in the data [31]. This can present challenges, but with advanced techniques such as machine learning and deep learning, the process can be automated and made more accurate to accomplish the following objectives:

- effectively handle the variability of layouts: due to the lack of a global standard, invoices often exhibit significantly different formatting. Naturally, the required legal information varies from country to country, and furthermore, it can be arranged in various ways within the document. Hence, it is crucial to have labeling and typing techniques in place to isolate the key elements of an invoice.
- train and rapidly adapt to new contexts: in a practical scenario, companies often lack a substantial corpus of invoices that are properly labeled for learning or testing purposes. However, for small companies, the invoices they handle are typically specific since they originate from a relatively limited number of customers

and suppliers. Consequently, it should be feasible to customize a system effortlessly for a particular situation,
- minimize the end-user task: while some systems rely on predefined invoice presentation styles, modifying these layouts typically requires extensive user interaction. Although it is important to engage the user in formulating their needs and specifying the desired information and management rules, it is essential to minimize the laborious manual tasks involved in system tuning,
- efficiently detect and extract tables from the invoices: tables play a crucial role in invoices, primarily used to present accounting information. However, their formatting can vary significantly, and in some cases, they may only be suggested without explicit graphic delimiters. Consequently, the detection of tables within invoices represents a significant challenge for automated systems, leveraging the distinctive characteristics of invoices compared to more generalized documents that rely on headings or predefined elements.

Automated processing of documents requires dedicated approaches based on the targeted domain. For instance, legal texts require specific techniques [17], [42]. The analysis of administrative documents, including invoices, has been an active area of research for many years [13]. The task is complex because invoices can come in various formats and contain a wide range of information such as invoice numbers, amounts, dates, and payment terms [31]. The lack of structure in documents poses a real challenge for companies [12]. To address this complexity, various techniques have been developed, such as Optical Character Recognition (OCR) [126] for digitizing paper invoices and natural language processing (NLP) techniques for extracting relevant information from the text. Neural networks are also frequently used for document classification tasks [137].

Commercial systems have been developed by companies like ITESOFT.[1] and ABBYY[2] [122] to automate the processing of invoices. These systems use a combination of OCR, NLP, and machine learning techniques to extract information from invoices and process them automatically. By integrating with the company's existing systems, such as accounting and enterprise resource planning (ERP) systems, these systems streamline the invoice processing workflow into a global electronic document management system (EDMS) [63]. Recent advances have led to the development of other end-to-end solutions for invoices [6].

Processing invoices requires complex administrative procedures and involves different departments such as accounting, logistics, and supply chain. To ensure efficiency and accuracy, specific workflows are often used [56]. These workflows typically involve multiple steps, such as document digitization, information extraction, and data validation, as well as security considerations [97]. Since invoices can take on various forms, statistical learning methods have been used to detect their possible classes [128].

The step of digitizing documents involves utilizing OCR technology to convert paper invoices into a digital format, allowing them to be processed and stored electronically with ease. Next comes the information extraction phase, which entails identifying the various identifiers such as types, amounts, dates, and other crucial details from the invoices. To achieve this, natural language processing (NLP) techniques, such as named entity recognition (NER), are typically employed, which aids in recognizing and extracting specific information from the text [50], [52].

Even if outside the scope of this overview, it is worth noting that classification techniques have been proposed for managing sets of invoices and categorizing financial transactions based on their economic nature [9], [131]. Machine learning can also be used to forecast financial data [55] related to invoicing, and time series tools such as [141], [142], and [140] are particularly useful for this purpose.

There have been many proposed solutions for managing information contained in scanned invoices, and most of these solutions are based on machine learning techniques, which have seen recent advances [50], [102]. In general, probabilistic and statistical approaches seem to be a natural way of understanding documents [88]. The first challenge in this field was identifying invoices from a set of documents [71], and models have been proposed to streamline this process [22].

Once invoices have been correctly scanned and identified, the next challenge is to extract relevant information from them. Labeling techniques can be applied using rules [33], but recent research has focused on using neural networks (NN) for named entity recognition (NER) tasks [73], [75]. This is because invoices often contain text sequences that are vastly different from natural language, and specific information extraction methods have been proposed to consider the specific structures in these documents. For example, [31] uses a star graph to consider the neighborhood of a text token, allowing for the context of a token to be taken into account when extracting information. This is a powerful method as it allows for meaningful information to be extracted from the document.

Several surveys provide an overview of general processing techniques for image documents, such as OCR techniques [59], text detection techniques [16], [146], NER approaches [75], [95], [144], and table processing [30], [38], [64]. However, few papers provide general considerations for invoice processing. One such paper is [52], which does not cover table extraction. In [6], a very interesting end-to-end system is proposed for processing invoices, including the different above-mentioned steps. Choices are made to select relevant techniques and the resulting system focuses on key fields extraction. From these considerations, our motivation is to offer a more comprehensive overview of available methods that practitioners can use to design end-to-end solutions for

[1]https://www.itesoft.com
[2]https://www.abbyy.com

invoice processing. Note that, we also pay particular attention to recent approaches based on graph representations. Please let us mention that our study is rooted in a practical experience, underpinned by the effective implementation of an electronic document management system in partnership with a company.

This overview aims to examine data extraction in the context of automated invoice processing. In Section II, we provide a comprehensive description of an invoice to highlight the critical data and structures that require attention. In our main section, Section III, we discuss the different components necessary for extracting this data. These include the digitization of the invoice using OCR (Section III-A), the development of a data extraction process (Section III-B), which involves recognizing specific entities (Section III-C) and identifying tables (Section III-D). Section III-E explores how geographical information can be utilized, with a particular focus on the use of graph-based representations.

Since such a survey involve numerous references, we propose an appendix with bibliographic tables that would help the reader to quickly identify the cited references according to the above-mentioned organization of the sections.

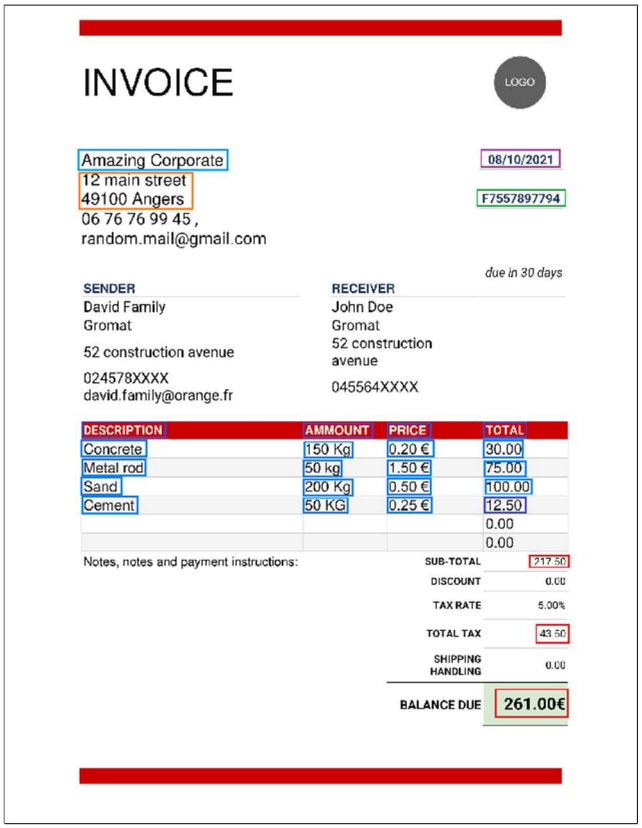


FIGURE 1. An invoice sample.

II. INVOICE MODELING

Defining a suitable representation of an invoice is an important step for clearly understanding its specifications. An invoice can include inputting data such as the invoice number, date, and amounts, as well as assigning it to a specific customer or project. In [22], a semantic network was used to describe the invoice domain by different levels of abstraction. Before going on through invoice processing techniques, we propose here a model that better focuses on relevant extraction tasks that are expected to be handled by an invoice processing application.

We chose to initially limit the scope of invoice extraction. Figure 1 illustrates a basic sample of an invoice, emphasizing key information sought by automated document processing tools. The extraction of specific fields, such as the invoice date (highlighted in the purple box), supplier address (in the orange box), and organizational providers (within the cyan box), is crucial. This survey places particular emphasis on table extraction, as indicated by data enclosed in blue and red boxes. Additionally, it is worth noting that the invoice contains other pertinent information that may be valuable for Named Entity Recognition (NER) processes, including the identification of both the sender and receiver. Figure 2 provides a comprehensive view of the typical content of a invoice by means of an UML class diagram.

Different types of information must be highlighted such as addresses, tables, dates, and actors (organizations or individuals identified on the invoice). This selected information seems coherent with the analysis of multiple invoice models and the usual requirements of the companies. One may identify 6 groups of data:

- **Actors**: individuals or companies involved in the invoice, such as a customer or a supplier.
- **Independent fields**: fields whose value is not linked to one of the other following fields and that often represent essential data for the invoice.
- **Information on the document**: information specific to the management of the document, such as its name or identifier in the file system, the dates of creation and processing of the document – all the data that are not extracted from the document but that come from its processing.
- **Addresses**: addresses contained in the document, with if possible precision on their types, billing address, delivery, or sender for example
- **Tables**: data tables are essential in invoices. They often include several lines of invoiced items, prices, quantities. . .
- **Date**: the set of dates, specific to the invoice processes such as the date of the edition of the invoice, the date of payment or of delivery.

Among these data, tables are considered complex to extract in this model because they often contain a large amount of structured data that needs to be parsed and understood. Companies need to perform verification operations on the table data, such as verifying VAT amounts and rates, or ensuring that the sum of the table lines matches the invoice amount. Efficient methods for extracting and analyzing table

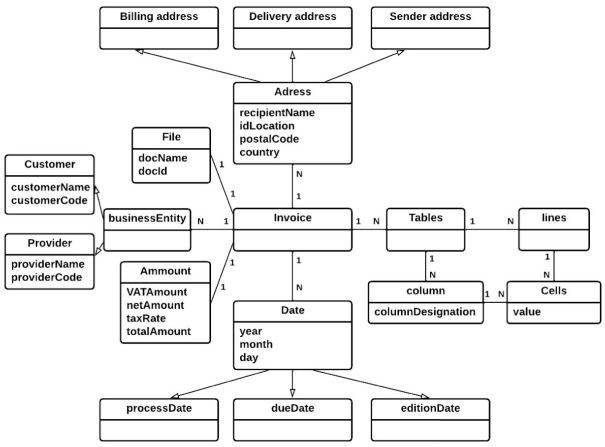


FIGURE 2. An UML model for invoices.

data are crucial due to the time-consuming and error-prone nature of the process.

III. INVOICE PROCESSING

As mentioned in the introduction, automated invoice processing requires a complete chain of software tools to automate the tasks involved in processing invoices. Hence, we could consider the following key features:

1) Optical Character Recognition (OCR): OCR is used to extract data from scanned or PDF invoices, making them searchable and easily readable by the system.
2) Machine Learning (ML): ML algorithms are widely used to classify and extract data from invoices, such as vendor information, invoice numbers, and amount. They are also intended to be able to extract structured information such as tables.
3) Workflow Automation: the system automatically route invoices for approval, flagging any discrepancies or errors for manual review.
4) Integration with ERP: automated invoice processing systems can integrate with enterprise resource planning (ERP) systems, allowing for seamless data transfer and real-time visibility into the invoice process.
5) Real-time Analytics: automated invoice processing systems can provide real-time analytics and reporting on invoice data, allowing businesses to track and analyze their spending. This is strongly related to business intelligence modules.
6) Compliance and Security: one may want to check compliance with tax regulations, and protect sensitive data through security measures such as encryption and secure data storage.

In this overview we restrict our scope to information extraction, considering raw scanned documents. Hence we restrict ourselves to the first two points of the above-mentioned features.

A. OPTICAL CHARACTER RECOGNITION

OCR systems have a long history, starting with early mechanical devices that were developed in the 1950s, such as GISMO (built by Sheppard in 1951). During the 1960s and 1970s, not much research was done on OCR due to the errors and slow recognition speed of the early systems [72]. However, during the past 40 years, there has been substantial research on OCR which has led to the development of document image analysis, multilingual, handwritten, and omni-font OCRs. Nevertheless, OCR technology is still far from matching human reading abilities and current research focuses on improving accuracy and speed of diverse document styles and languages, including complex languages.

Let us mention several state-of-the-art reviews [37], [93] that were already synthesizing the work in the early 90s. The seminal roots of OCR can be explored by reading the state-of-the-art of Mantas [83]. On the other hand, a good practical starting point for OCR can be accessed through the work of Breue [18], which presents an open-source OCR solution. A recent state of the art in OCR has been published in 2017 by Islam et al. [59].

Hence, OCR is a crucial discipline in image interpretation with highly important potential applications. A major problem was handwritten character recognition [89], including the need for a database. Note that important conferences were focusing on OCR since the 90s, e.g. ICDAR [1] with dedicated workshops [49]. Neural networks have then considered to overcome the previous limitations. In [28], the use of projection profile features coupled with a back-propagation neural network classifier has proven highly effective. Nowadays, neural networks are widely used in OCR technologies. Let us quote some recent works: in [96] the author consider a significantly extensive Urdu corpus ideally suited for applications involving deep learning techniques, [62] introduced end-to-end learning methods for recognizing arithmetic expressions combining deep a convolutional neural network and convolutional recurrent neural network, in [66] the authors propose an exploration of character recognition, encompassing both monolingual and multilingual contexts, utilizing both deep and shallow architectural approaches.

Among the impressive number of works related to OCR, let us mention the work of Mithe et al. [92] that presents a solution using an OCR solution to extract text and then send it to a voice synthesizer. The main objective behind this solution is to produce a solution that transforms an image into a speech on the contained text in the picture. This article proves that the processing of an image makes it possible to obtain fully structured information.

Of course, it is also very important to clearly assess the performance of OCR using suitable measures and available benchmark sets [98]. Let us note here that image processing techniques can be used to get better initial documents, even before applying OCR. Morphological operations, such as dilation, erosion, and opening, are commonly used in image processing to remove noise, blur,

and skewness from document images. These techniques have been applied to prepare images for OCR and to locate text-containing parts in an image [146], for instance using OpenCV [44].

Back to our structured information extraction concern, a dedicated challenge has been recently proposed by Huang et al. [58] at the ICDAR2019 conference. The prize for the best paper was awarded to Zhong et al. [156] which offers a solution based on neural networks for the recognition of certain entities related to the formatting of documents.

In recent times, there has been ongoing research in the field of OCR. Let us mention a first work [10] that specifically focuses on the application of OCR for the recognition of written texts within a medical context. A promising development in OCR techniques aligns with the progress in deep learning, as exemplified by the work of Li et al. [77]. In this work, the authors have adapted the transformer architecture to address OCR challenges and have presented a comprehensive benchmark featuring many contemporary techniques. This reflects the dynamic evolution of OCR methodologies, where advancements in deep learning play a pivotal role.

B. DATA EXTRACTION

Once the OCR has been applied, we are generally left with a set of PDF documents that are expected to be searchable and exploitable. Let us first begin with a general consideration of possible data extraction at this stage. At first glance, we may consider the visual aspect of the document and the relative positions of the information that it contains.

The work of Taylor et al. [132] presents an overview of the problem of document extraction from scanned documents. This article highlights the problems of alignment of the text. It also highlights that only part of the information is relevant to extract.

The global layout of the document has to be taken into account [7]. Ahmad and Man [2] use the concept of unstructured, semi-structured, or structured documents. The work of Yao et al. [145] on the relationships between entities, which is also unlabeled, also seems very relevant. Sun et al. [130] present a solution for orienting documents according to a specific entity (QR Code in the article). These methods address two common challenges in data extraction: document orientation and scale. The invoices, which are in the form of images, are first preprocessed to remove any unnecessary background and to correct the angle of the invoice. Then, the region containing the desired information on the invoice is identified using template matching. Another system (BINYAS) [16] performs document layout analysis for document image processing. This system uses connected components and pixel analysis for classifying elements such as paragraphs, graphics, images, and tables in the document. In [11] the authors propose a dataset for unstructured invoice documents that covers a wide range of layouts, which is designed to generalize key field extraction tasks for unstructured documents. The dataset is evaluated using various feature extraction techniques as well as Artificial Intelligence methods.

As already mentioned, tabular content extraction from PDF documents is of great importance, in particular for benefiting from available open-source document repositories [30]. The extraction and processing of data from PDF files have indeed always been studied [81]. Data in tables is often displayed in a tabular format. Although tables may appear simple, extracting and processing them from PDFs can be difficult and require complex computational methods [48]. The purpose is often to produce new formats from initial PDFs such as XML files [112]. Note that PDFs do not typically record the structure of their graphical objects in their description, although it could be done.

Of course, visual separators are important for identifying tables in documents as they reveal the table structures [41]. Actually, when tables include visible lines that can be extracted from the document, considering the maximum independent set of rectangles (MISR) problem seems relevant [24]. MISR consists of finding in a set of rectangles the smallest set of rectangles with no intersection. Unfortunately, many tables miss lines to separate some columns or rows and some techniques do not apply in these cases. Yildiz et al [148] present approaches based on line intervals and columns to identify the entities corresponding to tables' cells. Note that table extraction will be detailed in Section III-D.

Deep learning techniques are now widely used to identify and extract tables in PDF documents [46], [151]. This aspect will be detailed later. Note that some work uses APIs such as *PDFminer* to transform PDF into XML and perform supervised learning on XML [103].

C. ADDRESSING SPECIFIC INFORMATION EXTRACTION: NAMED ENTITY RECOGNITION

In the scope of this study, we are not concerned with general document processing but with invoices that are restricted to a specific domain, whose terms and concepts are known. Hence we are concerned by the semantics of the documents. The analysis of invoices is hence related to Natural Language Processing (NLP) and more specifically to Named Entity Recognition (NER) (see [95], [144] for dedicated surveys).

The problem of named entity recognition (NER) was presented by Marsh and Perzanowski at the MUC conference [85]. NER involves labeling a text by associating each character string with a specific category, such as a person, location, organization, temporality, amount, or percentage. This problem is also referred to as entity labeling or entity extraction. Research intensifies then on this purpose. During CoNLL-2003 [134] the focus was put on language-independent named entity recognition. The challenge concentrates on four types of named entities: persons, locations, organizations and names of miscellaneous entities that do not belong to the previous three groups. During same period, the ACE program's goal [35]) was to advance technology for automatically extracting information from human language data. This includes identifying mentioned

entities, determining the relationships between these entities as expressed in the text, and recognizing the events in which these entities are involved. The program encompasses various data sources.

At this time the NER was restricted to the names of people, locations, and organizations, and sometimes to some other proper names, which does not cover all the possible types expected in an invoice.

The specific set of labels used in NER depends on the data and the task at hand. NER is, of course, strongly dependent on the application domain (e.g., [106], [153]). Some researchers limit themselves to the 6 initial categories (person, location, organization, temporality, amount, and percentage) and believe that these labels are sufficient for all NER tasks. However, other researchers argue that specific labels may be necessary to effectively solve specific NER tasks [20]. The number of labels used can vary depending on the complexity of the data and the specific requirements of the task. Therefore, the choice of labels is often a trade-off between the need for specific information and the complexity of the model. Let us particularly mention the works of Alfonseca et al. [3] and R. Evans [40] that use the notion of ''open domain''. Recently, data sets have been made available for NER related to invoices [11]. From a practical point of view, Mikolov et al. [90] demonstrate the benefit of using vector representation of words and also that it is possible to train a model of neural networks on a large training set, including a large number of sentences with approximately one billion words and a vocabulary of more than one million different words. A month later, Mikolov et al. [91] considered a distributed representation of words and prove that, by adding certain vectors of words, the learning process allows one to learn the meaning of the words. The linguist Scharolta Katharina Siencnik [125] attempts then to demonstrate the possible application of these algorithms to named entity recognition.

While state-of-the-art named entity recognition systems relied heavily on hand-crafted features and domain-specific knowledge, new neural architectures for NER were proposed [27], [73]. These architectures aim to improve performance by leveraging the strengths of neural networks, such as their ability to learn useful features from data, while still addressing some of the limitations of previous methods. Convolutional neural networks (CNN) [79] have been then considered with NER problems [138], [150] as well as bidirectional networks [4]. Let us mention work on the identification of depression according to the answers of the patient in an interview [114] as well as the work of He et al. [54] to establish distant dependencies between the entity terms via the processing of CNN.

ELMo is a language model that was developed by Matthew E. Petters and his team [104]. Unlike traditional word embeddings that represent words as fixed vectors, ELMo utilizes the context in which words appear to generate more dynamic and informative word embeddings. The model is semi-bidirectional, which means it takes into account both the preceding and succeeding words in a sentence to better understand the meaning of the word it is trying to represent.

ELMo's innovative approach to word embeddings quickly gained attention from researchers in the natural language processing (NLP) community. Dogan et al. [36] applied ELMo's neural network architecture to tackle Named Entity Recognition (NER) problems, which involve identifying and classifying entities in text such as names, dates, and locations.

While ELMo showed promising results, it had a limitation that it could not be effectively fine-tuned with other models using a ''masked language model''.

To address this shortcoming, Devlin et al. proposed BERT [34], a bidirectional model also based on ELMo, which has since become one of the most widely used pre-training models for NLP tasks. BERT uses a ''masked language model'' that allows it to refine its representations using an unsupervised pre-training method. This makes it possible for BERT to generate high-quality word embeddings that can be fine-tuned with other NLP models to achieve state-of-the-art performance on a wide range of language tasks.

Ali Safaya et al. [115] demonstrate the possible association between CNN and BERT and study its efficiency. This work focuses on BERT associated with Arabic, Turkish, and Greek languages, which presents a more structured construction than some languages. This study achieves better efficiency in the recognition of hateful content for these languages.

GPT models have become essential in natural language processing (NLP) due to their ability to be fine-tuned for specific NLP tasks. Radford's work on language model transformers, particularly the GPT model [110], has revolutionized the field of NLP. Unlike bi-directional models like BERT and ELMo, GPT is a unidirectional model where word embeddings are only enhanced in one direction, typically from left to right. This unidirectional architecture makes GPT particularly useful in language prediction tasks, where the model predicts the next word in a sentence based on the preceding words.

Getting back closer to our main concern, Francis et al. [43] present a solution for extracting data from financial or medical documents using a neural network trained for named entity recognition, which evaluates the efficiency of a character-based model or on a word. One also has to consider general language processing. For instance, the work of Suárez et al. [129] on the state of the art of named entity recognition for the French language can be useful for dealing with French invoices. Hamdi et al. [52] present tools to improve the learning of invoice-specific labeling by reducing the cost of time and human intervention.

To ensure better explainability, rule-based approaches are useful alternative techniques for achieving NER [26]. Shreeshiv et al. [102] address the extraction of key parameters of the invoice (KPIE), by proposing a rule-based approach and an approach based on neural networks to recognize these parameters of the invoice. Declarative approaches based on

constraint solving should also be considered as promising research direction [5].

Practical solutions are available for NER, such as that of Nanonets.[3]com and ABBYY. A well-documented example explains the use of BERT in the case of a NER [34].

In summary, there are two main approaches. The historical rules-based approach tends to be inspired by the rules of traditional grammar for labeling words in the context of the text. This approach is very efficient on specific domains because the writing of the rules is often very oriented towards the desired domain to avoid ambiguity. Nevertheless, this specialization leads to processing difficulties for the new context, not defined during the implementations. It is also necessary to rework the model to extend its capacity. This step often requires the intervention of an expert.

The neural network approach to label the entities of our document seems interesting to avoid spending too much time defining the labeling rules. This method better manages the new domains and we can more easily set up automation of the relearning for the new concepts treated. Nevertheless, NN requires huge computational resources and training corpora to be efficient.

In Figure 3 we propose an empirical evaluation of NER systems according to the state of the art, the statements of the various specialists in this field, and the needs encountered in companies. This evaluation is therefore subjective.

FIGURE 3. Advantages and disadvantages of NER methods according to the state of the art.

D. FOCUS ON TABLE EXTRACTION

Examining more precisely invoices leads to consider that most of them include tables as a main structural character. Hence, table detection within invoices appears as an important processing task [121]. Table processing is indeed an old challenge (the 2004 survey [149] propose already an overview of the field) but these challenges are still active [45].

Understanding information embedded into tables involves three steps as quoted in [61]: detecting the table boundaries, identifying the structure of the table including rows, columns, and cell positions, and recognizing the contents of the table (tokens of information that are expected to be presented in a more readable format). The layout is an important aspect [69]. Techniques used for detection include object detection models [23] like Faster-RCNN (Region Based Convolutional Neural Networks) and Mask-RCNN [107] and NLP-based methods that incorporate both textual and visual features [57].

Note that TableBank [76] includes a new image-based table detection and recognition dataset. PubLayNet [156] can accurately recognize the layout of scientific articles after training on over one million PDF articles. LayoutLMv3 [57] is pre-trained with a word-patch alignment objective to improve cross-modal alignment. This allows the model to predict whether the image patch associated with a text word has been masked.

Deep learning techniques are now widely used for achieving table structure recognition. Recently, Kavasidis et al [67] introduce a fully-convolutional neural network that utilizes saliency-based techniques for multi-scale reasoning with visual cues. They also incorporate a fully-connected conditional random field to precisely locate tables and charts within digital or digitized documents. A common approach consists of using a bi-directional RNN with Gated Recurrent Units (GRUs) to process image data [68]. The pre-processing step is used to form the image data so that it can be fed into the network. The bi-directional RNN with GRUs is then used to analyze the image data and extract features. Finally, a fully connected layer with a softmax activation function is used to classify the image based on the features extracted by the RNN. Gilani et al. [47] introduced an approach based on deep learning to detect tables. Our method begins by pre-processing document images, which are then input into a Region Proposal Network (RPN), followed by a fully connected neural network to identify tables. Their method demonstrates remarkable precision when applied to document images with diverse layouts, encompassing documents, research papers, and magazines. Vine et al. [136] introduce a two-step approach including a generative adversarial network (GAN) and a genetic algorithm to optimize a distance measure between candidate table structures. Another two-step process that uses cell detection and interaction modules to recognize the structure of a table is proposed in [111]. The cell detection module is used to locate and identify individual cells in the table image. The interaction module then predicts the associations between the detected cells, such as their row and column associations. This approach can be useful for determining the overall structure of a table, including the number of rows and columns, as well as the relationships between cells within the table. Convolutional networks have, of course, been explored [67], [124], with Split and Merge models [133]. In [99], the authors consider also explainability as an issue in an NN. Global end-to-end solutions are now available TableNet [100], DeepDeSRT [119] PubTabNet [155] or GTE [154]. Dedicated benchmarks repository have been proposed to evaluate these methods: Tablebank [76] (417K high quality

[3]https://www.nanonets

labeled tables) and even a novel dataset derived from the RVL-CDIP invoice data [113].

Table detection may also rely on more specific knowledge. In [139], the authors propose a system for automatically generating ground truth data for training table detection algorithms. We found in the literature important works on layouts, for example in [105], David P. al use ''Conditional Random Fields'' (CRF) to compose different layouts of a table that can sometimes overlap and may be misinterpreted by other modeling languages. Tools such as *TableSeer* [80] searches for forms that can correspond to tables to extract them and be able to execute queries on their contents.

The specific structures of invoices lead to considering the geographical organization of the document and graph-based models are thus relevant [121]. Recent work [65] proposes an approach to detect the general frame of a table and extract its content. Focusing on more specific tables, their characteristics are also intended to help these tasks, such as headers [120]. Rule-based systems, which were seminal table extraction techniques, may also be relevant [123].

Graph-based approaches also seem to be a natural way to handle tables. In [116] the authors use graph mining for extracting tables using key fields. Hence, Graph Neural Networks (GNN) [118] appears as natural to handle graph-based knowledge [147]. Graph Neural Networks (GNNs) can indeed capture the local repeating structural information in invoice document tables [113]. In [78], the authors propose a method based on GNN to mix position and text. Their algorithm also uses visual recognition to predict the right numbers of columns and lines. In [108] architecture that combines the benefits of convolutional neural networks for visual feature extraction and graph networks is introduced for dealing with the problem structure. Cell detection and cell logic are used to predict the location of the cells in [143]. [152] presents a unified framework that utilizes a combination of vision, semantics, and relations for analyzing document layouts, supporting natural language processing and computer vision-based methods. Slightly different, LGPMA [109] employs a soft pyramid mask learning approach to recover table structure by analyzing both local and global feature maps. Additionally, it considers the location of empty cells during this process.

E. HANDLING GEOGRAPHIC INFORMATION IN THE INVOICES: POSSIBLE PERSPECTIVES FOR GRAPH-BASED REPRESENTATIONS

Since the layout of invoices is particularly relevant as described above, let us explore the modeling and the processing of geometric or geographic information, to discover links that cannot be handled by a purely semantic analysis of the document. For example, an invoice may contain a keyword and its expected associated value close to it. Let us review some methods for representing and exploring this structured data. For instance, Esser et al [39] try to extract templates from scanned documents.

This section is devoted to methods that would not consider image processing or NN to handle the global layout of the document using a training process. We are merely interested in techniques based on representation models and associated solving techniques to process geometric data in a more frugal (without the need for a huge and costly training) and more declarative way.

A long time ago, Cesarini et al. [21] were already interested in the structural analysis of a document by trying to label areas. They consider that an invoice is a set of regions that can be identified using their relative geometrical position.

As mentioned in Section III-D, graph-based representation has been explored for handling the structures of the tables in documents. Therefore, we focus here on such representations and how they can be exploited to efficiently retrieve table structures and their content. Since the structure of a table may contain different levels, we argue that several levels of abstraction are needed to represent the geometrical structure of a table. Using models with geometric constraints and enabling their declarative handling has been explored in [19]. An abstract model is linked to a graphic model and a refinement process is proposed. Geometric constraints [94] require dedicated constraint solvers according to targeted domains. In [117], we propose an approach based on hypergraph to handle table extraction. Hypergraphs [15] are classic extensions of graphs and enable more powerful models. Hence, after suitable modeling, one may consider table extraction in a document as an isomorphism problem in hypergraphs [14]. The sub-isomorphism problem is NP-Complete [29] and its complexity has been refined according to parameters [86]. Solvers, such as the Glasgow solver [87] are available to solve this problem as well as efficient algorithms [127] including recent quantum search algorithms [84]. In a recent work [74], the author proposes to represent tables as planar graphs with cell regions as their faces. They generate junction confidence maps and line fields using heatmap regression networks. Their approach mixes deep NN and constrained optimization problems.

F. TURNING TO EFFICIENT SOLUTIONS FOR INDUSTRY

As a starting point, it might be worthwhile to delve into the intricacies of Extraction, Transformation, and Loading (ETL) processes [135], which form the backbone of operations within a data warehouse architecture, with the aim of acquiring data from diverse document sources, each characterized by its potential multimodal attributes. A critical dimension in this context is the recognition that data assimilation stems from a variety of document origins. The multifaceted nature of these documents underscores the complexity of the task at hand. Furthermore, automated document processing systems must exhibit the capability to update data at regular intervals, emphasizing the need for real-time adaptability. Following these lines, Figure 4 encapsulates the multifunctional essence of information extraction from invoices. It provides a visual representation of the

intricate multitasking inherent in the information extraction workflow.

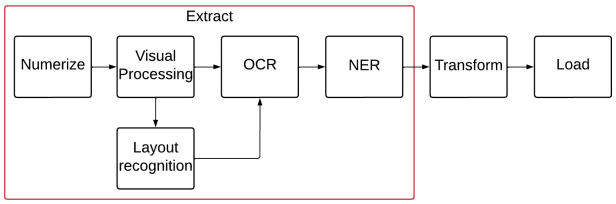


FIGURE 4. Different steps for in an ETL process.

Some industry solutions offer a partial solution to the total ETL process. They are based on plugins designed for each information retrieval task. For instance, the Azure[4] solution developed by Microsoft offers numerous APIs for processing documents including OCR and NER. The ABBY solution is split into different programs : Flexicapture for OCR and FlexiLayout for extracting data from a document using templates.

Transformers maybe now used to provide end to end solutions and address various modalities related to document processing tasks, such as classification, question answering or NER [32], [70]. The diverse nature of documents necessitates multimodal reasoning that encompasses various types of inputs [8]. These inputs, including visual, textual, and layout elements, are found in a variety of document sources. These aspects may be considered for developing efficient invoices processing tools.

IV. CONCLUSION

In conclusion, invoices are crucial documents for companies as they serve as proof of purchase and are necessary for accounting and tax purposes. The processing of invoices can be time-consuming and prone to errors, but recent advances in technology have led to the development of systems that automate the process. These systems use a combination of OCR, NLP, and machine learning techniques to digitize paper invoices and extract relevant information. The processing of invoices involves different steps such as document digitization, information extraction, and data validation, and specific workflows are often used to ensure efficiency and accuracy. The challenge of processing invoices lies in handling the variability of layouts, language, and terminology, and the presence of errors or inaccuracies in the data.

In this survey, we have reviewed the essential components that must be taken into account when developing an automated invoice processing system. Our goal is to provide valuable insights to researchers and engineers striving to create end-to-end solutions, and in this pursuit, several critical factors demand careful consideration:

- Document Quality: The quality of the documents input for processing plays a crucial role. Standard digitized invoices can often be handled with relatively basic OCR systems. However, when dealing with documents

[4]https://azure.microsoft.com/en-us

TABLE 1. Summary of cited surveys.

Main topic	References
OCR techniques	[37], [59], [66], [83], [93], [98] [10], [77]
Text detection techniques	[16], [147]
NER approaches	[75], [95], [107], [145], [154]
Table processing	[30], [38], [64], [150]
Convolutionnal networks	[79]
Invoice processing	[52]
Graph neural networks	[148]
Information retrieval	[51]

TABLE 2. Summary of available datasets for document analysis and recognition.

Name	Desc.	Ref.
CORD	Receipt Dataset for Post-OCR Parsing	[101]
rvl-cdip-invoice	set of invoices extracted from RVL-CDIP	[53]
GHEGA-DATASET	labeled dataset for document understanding research experiments	[88]
ICDAR2019	competition on scanned receipt ocr and information extraction.	[58]
FUNSD	Form Understanding in Noisy Scanned Documents challenge	[60]
PubLayNet	dataset for document layout analysis	[158]

TABLE 3. Summary of available datasets for table analysis.

Name	Desc.	Ref.
cTDar	annotated documents with table entities	[45]
SciTSR	large-scale table structure recognition dataset	[25]
PubTabNet	image-based table recognition	[157]
WikiTableSet	publicly available image-based table recognition dataset in three languages built from Wikipedia	[82]

TABLE 4. Summary of main cited works on OCR.

Reference	Topic
[18]	open source OCR solution
[89]	handwritten character recognition
[28]	handwritten OCR
[96]	text recognition using deep learning
[62]	deep learning based OCR
[92]	OCR solution including image to speech transformation
[98]	benchmark sets for OCR
[44]	OpenCV system
[158]	neural network based OCR
[58]	description of Icdar2019 competition on scanned receipt
[10]	A survey into OCR specialized for medical reports.
[77]	A technique based on transformer architecture for OCR and a benchmark with modern solutions

TABLE 5. Summary of main cited works on data extraction.

Reference	Topic
[133]	seminal work on data extraction
[7]	computational-geometry algorithms for analyzing document structures
[2]	handling multiple types of data structures
[146]	considering relations between data
[131]	orientation of documents
[16]	document layout analysis
[11]	data sets for evaluation
[81]	seminal work on pdf documents management
[48]	data extraction from tables
[113]	table extraction for pdf documents
[41]	table detection for multipage pdf documents
[24]	solving of the maximum independent set of rectangles problem
[149]	pdf2table : method for extracting table
[46]	graph neural network for extracting tables from pdf documents
[152]	deep learning for pdf table extraction
[104]	presentation of TAO for table detection and extraction

TABLE 6. Summary of main cited works on NER.

Reference	Topic
[85]	seminal work on NER
[135]	NER Challenge at CoNLL
[35]	ACE program : challenge for NER systems
[20]	empirical study of NER
[3]	procedure to automatically extend an ontology with domain specific knowledge
[40]	system for NER in the open domain
[90]	model architectures for computing continuous vector representations of words (word2vec)
[91]	distributed architectures for word2vec
[126]	adaptation of word2vec to NER
[73]	neural networks for NER
[27]	neural networks for NER
[4]	bidirectional recurrent neural network for NER
[34]	presentation of BERT
[54]	combination of convolutional neural network with BERT
[115]	application of bert-cnn for an application in health care
[105]	presentation of ELMO, a model language word representation
[36]	use of ELMO for NER
[116]	Bert-cnn for speech identification
[111]	enhancing language comprehension through pre-training
[43]	data extraction from financial documents
[130]	state of the NER for French language
[52]	specific work on invoices
[26]	rule-based information extraction systems
[103]	information extraction from scanned invoices
[5]	constraint satisfaction for invoice processing
ABBY	a commercial system for NER

exhibiting orientation issues or containing handwritten sections, a more sophisticated image processing pipeline and highly efficient text recognition are imperative. Real-world financial documents, for instance, may feature handwritten notes from employees seeking reimbursements, making document quality a critical determinant.

TABLE 7. Summary of main cited works on table Extraction.

Ref.	Topic
[122]	reference work on table extraction
[45]	ICDAR 2019 Competition on Table Detection and Recognition
[69]	the T-Recs system for table recognition
[23]	algorithm for searching parallel lines in documents to extract tables
[61]	proposal for the representation of tables (Wang Notation Tool)
[158]	Publaynet, a data bank for table extraction
[76]	TableBank, a data bank for table extraction
[108]	presentation of CascadeTabNet an end to end system using Convolutional Neural Networks
[57]	LayoutLMv3: a general-purpose pre-trained model for documents
[67]	Convolutional Neural Network for table detection
[68]	approach based on bi-directional gated recurrent unit networks
[47]	deep learning for table detection
[137]	approach based on a generative adversarial network
[112]	two step approach that combines cell detection and interaction module
[125]	DeepTabStR : a deep learning based system for table recognition
[134]	use of a novel deep learning models (Split and Merge models)
[99]	explainability to get the semantic structures of tables
[100]	Tablenet : end to end solution for table extraction
[120]	DeepDeSRT : end to end solution for table extraction
[156]	PubTabNet : end to end solution for table extraction
[155]	GTE : end to end solution for table extraction
[114]	use of Graph Neural Network for table extraction
[140]	system for automatically generating ground truth data for training table detection algorithms
[106]	introduction of conditional random fields to manage layouts of a table
[80]	presentation of TableSeer, a search engine for tables
[65]	an end-to-end table structure recognition system using a Yolo-based object detector
[121]	segmentation techniques for tables
[124]	presentation of TabbyPDF: heuristic-based approach to table detection and structure recognition
[117]	approach that uses a graph-based representation of documents
[109]	architecture that combines convolutional neural networks and graph networks
[144]	presentation of TGRNet an end-to-end trainable table graph reconstruction network
[153]	presentation of VSR a combination of computer vision and NLP techniques
[110]	LGPMA a system that uses the concept of Local and Global Pyramid Mask Alignment

- Invoice Content: The nature of the invoice content is another crucial consideration. In cases where invoices consist of limited and concise information, without extensive descriptions or intricate commercial terms, employing simple Named Entity Recognition (NER) techniques based on a compact model, as exemplified in Figure 2, suffices. Conversely, for more complex scenarios, the integration of Natural Language Processing (NLP) techniques becomes essential to delve into the semantic nuances of scanned texts.
- Layout Diversity: The diversity of invoice layouts cannot be underestimated. When documents are associated with a finite number of suppliers or clients, rule-based techniques designed to match predefined layouts can be

harnessed. Moreover, these techniques may offer flexibility, allowing end-users to fine-tune the system to visually locate and extract key information from invoices.

- Annotated Data Sets: Machine learning techniques, while powerful, rely heavily on sizable and representative training datasets for optimal performance. As mentioned in this survey, rule-based approaches can often be generic enough to process invoices effectively without necessitating extensive supervised learning processes.
- Table Diversity and Quality: Tables within invoices represent a pivotal aspect of the processing pipeline. While basic tables can be detected using image processing and neural network-based algorithms, more complex scenarios emerge when tables are incomplete and exhibit considerable diversity, often due to variations in invoice layouts. In such cases, recent graph-based algorithms present a compelling and efficient alternative.

By taking these facets into account, engineers can embark on the development of robust, efficient, and adaptable automated invoice processing systems that cater to a wide spectrum of real-world invoice scenarios. In this context, hybrid methods, combining both rule-based and neural network approaches.

In recent times, there has been a notable emergence of large language models (LLM). These models present promising prospects for document processing by integrating structural and semantic recognition to achieve effective extraction of information from both structured and semi-structured documents.

APPENDIX
BIBLIOGRAPHIC TABLES

See Tables 1–7.

REFERENCES

[1] *ICDAR 2nd International Conference Document Analysis*, Comput. Soc., Washington, DC, USA, 1993.

[2] I. A. S. Ahmad and M. Man, ''Multiple types of semi-structured data extraction using wrapper for extraction of image using DOM (WEID),'' in *Proc. Regional Conf. Sci., Technol. Social Sci.* Singapore: Springer, 2016, pp. 67–76.

[3] E. Alfonseca and S. Manandhar, ''An unsupervised method for general named entity recognition and automated concept discovery,'' in *Proc. 1st Int. Conf. Gen. WordNet*, 2002.

[4] M. Ali, G. Tan, and A. Hussain, ''Bidirectional recurrent neural network approach for Arabic named entity recognition,'' *Future Internet*, vol. 10, no. 12, p. 123, Dec. 2018.

[5] J. Andersson, ''Automatic invoice data extraction as a constraint satisfaction problem,'' Uppsala Univ., Uppsala, Sweden, Tech. Rep., 2020.

[6] H. Arslan, ''End to end invoice processing application based on key fields extraction,'' *IEEE Access*, vol. 10, pp. 78398–78413, 2022.

[7] H. S. Baird, ''Background structure in document images,'' *Int. J. Pattern Recognit. Artif. Intell.*, vol. 8, no. 5, pp. 1013–1030, Oct. 1994.

[8] S. Bakkali, Z. Ming, M. Coustaty, M. Rusiñol, and O. R. Terrades, ''VLCDoC: Vision-language contrastive pre-training model for cross-modal document classification,'' *Pattern Recognit.*, vol. 139, Jul. 2023, Art. no. 109419.

[9] C. Bardelli, A. Rondinelli, R. Vecchio, and S. Figini, ''Automatic electronic invoice classification using machine learning models,'' *Mach. Learn. Knowl. Extraction*, vol. 2, no. 4, pp. 617–629, Nov. 2020.

[10] P. Batra, N. Phalnikar, D. Kurmi, J. Tembhurne, P. Sahare, and T. Diwan, ''OCR-MRD: Performance analysis of different optical character recognition engines for medical report digitization,'' *Int. J. Inf. Technol.*, vol. 16, no. 1, pp. 447–455, Jan. 2024.

[11] D. Baviskar, S. Ahirrao, and K. Kotecha, ''Multi-layout invoice document dataset (MIDD): A dataset for named entity recognition,'' *Data*, vol. 6, no. 7, p. 78, Jul. 2021.

[12] D. Baviskar, S. Ahirrao, V. Potdar, and K. Kotecha, ''Efficient automated processing of the unstructured documents using artificial intelligence: A systematic literature review and future directions,'' *IEEE Access*, vol. 9, pp. 72894–72936, 2021.

[13] A. Belaïd, V. P. D'Andecy, H. Hamza, and Y. Belaïd, ''Administrative document analysis and structure,'' in *Learning Structure and Schemas From Documents* (Studies in Computational Intelligence), B. Marenglen and X. Fatos, Eds. Berlin, Germany: Springer, 2011.

[14] C. Berge, ''Isomorphism problems for hypergraphs,'' in *Hypergraph Seminar*. Dordrecht, The Netherlands: Springer, 1974, pp. 1–12.

[15] C. Berge, *Graphs and Hypergraphs*. Amsterdam, The Netherlands: Elsevier, 1985.

[16] S. Bhowmik, R. Sarkar, M. Nasipuri, and D. Doermann, ''Text and non-text separation in offline document images: A survey,'' *Int. J. Document Anal. Recognit.*, vol. 21, nos. 1–2, pp. 1–20, Jun. 2018.

[17] C. Biagioli, E. Francesconi, A. Passerini, S. Montemagni, and C. Soria, ''Automatic semantics extraction in law documents,'' in *Proc. 10th Int. Conf. Artif. Intell. law*, Jun. 2005, pp. 133–140.

[18] T. M. Breuel, ''The OCRopus open source OCR system,'' *Proc. SPIE*, vol. 6815, Jan. 2008, Art. no. 68150F.

[19] T. L. Calvar, F. Chhel, F. Jouault, and F. Saubion, ''Toward a declarative language to generate explorable sets of models,'' in *Proc. 34th ACM/SIGAPP Symp. Appl. Comput.*, Apr. 2019, pp. 1837–1844.

[20] H. Ceovic, A. S. Kurdija, G. Delac, and M. Šilic, ''Named entity recognition for addresses: An empirical study,'' *IEEE Access*, vol. 10, pp. 42108–42120, 2022.

[21] F. Cesarini, E. Francesconi, M. Gori, S. Marinai, J. Q. Sheng, and G. Soda, ''Rectangle labelling for an invoice understanding system,'' in *Proc. 4th ICDAR*, Aug. 1997, pp. 324–330.

[22] F. Cesarini, E. Francesconi, M. Gori, S. Marinai, J. Q. Sheng, and G. Soda, ''Conceptual modelling for invoice document processing,'' in *Proc. 8th Int. Workshop Database Expert Syst. Appl.*, R. R. Wagner, Ed., Sep. 1997, pp. 596–603.

[23] F. Cesarini, S. Marinai, L. Sarti, and G. Soda, ''Trainable table location in document images,'' in *Proc. 16th Int. Conf. Pattern Recognit.*, Quebec, QC, Canada, Aug. 2002, pp. 236–240.

[24] P. Chalermsook and J. Chuzhoy, ''Maximum independent set of rectangles,'' in *Proc. 20th Annu. Symp. Discrete Algorithms*. Philadelphia, PA, USA: SIAM, Jan. 2009, pp. 892–901.

[25] Z. Chi, H. Huang, H.-D. Xu, H. Yu, W. Yin, and X.-L. Mao, ''Complicated table structure recognition,'' 2019, *arXiv:1908.04729*.

[26] L. Chiticariu, Y. Li, and F. Reiss, ''Rule-based information extraction is dead! long live rule-based information extraction systems!'' in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2013, pp. 827–832.

[27] J. P. C. Chiu and E. Nichols, ''Named entity recognition with bidirectional LSTM-CNNs,'' *Trans. Assoc. Comput. Linguistics*, vol. 4, pp. 357–370, Dec. 2016.

[28] A. Choudhary, R. Rishi, and S. Ahlawat, ''Unconstrained handwritten digit OCR using projection profile and neural network approach,'' in *Proc. Int. Conf. Inf. Syst. Des. Intell. Appl.*, Visakhapatnam, India. Berlin, Germany: Springer, 2012, pp. 119–126.

[29] S. A. Cook, ''The complexity of theorem-proving procedures,'' in *Proc. 3rd Annu. ACM Symp. Theory Comput. (STOC)*, 1971, pp. 151–158.

[30] A. S. Corrc^a and P.-O. Zander, ''Unleashing tabular content to open data: A survey on PDF table extraction methods and tools,'' in *Proc. 18th Annu. Int. Conf. Digit. Government Res.*, C. C. Hinnant and A. Ojo, Eds., Jun. 2017, pp. 54–63.

[31] V. P. d'Andecy, E. Hartmann, and M. Rusiñol, ''Field extraction by hybrid incremental and a-priori structural templates,'' in *Proc. 13th IAPR Int. Workshop Document Anal. Syst. (DAS)*, Apr. 2018, pp. 251–256.

[32] B. Davis, B. Morse, B. Price, C. Tensmeyer, C. Wigington, and V. Morariu, ''End-to-end document recognition and understanding with dessurt,'' in *Proc. Eur. Conf. Comput. Vis.*, in Lecture Notes in Computer Science, vol. 13804, Tel Aviv, Israel, L. Karlinsky, T. Michaeli, and K. Nishino, Eds. Cham, Switzerland: Springer, 2022, pp. 280–296.

[33] A. Dengel and B. Klein, ''*smartFIX* : A requirements-driven system for document analysis and understanding,'' in *Proc. 5th Int. Workshop*, in Lecture Notes in Computer Science, vol. 2423, D. P. Lopresti, J. Hu, and R. S. Kashi, Eds. Berlin, Germany: Springer, 2002, pp. 433–444.

[34] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, ''BERT: Pre-training of deep bidirectional transformers for language understanding,'' in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, vol. 1, Minneapolis, MN, USA, J. Burstein, C. Doran, and T. Solorio, Eds. Minneapolis, MN, USA: Association for Computational Linguistics, Jun. 2019, pp. 4171–4186.

[35] G. R. Doddington, A. Mitchell, M. A. Przybocki, L. A. Ramshaw, S. M. Strassel, and R. M. Weischedel, ''The automatic content extraction (ACE) program—Tasks, data, and evaluation,'' in *Proc. 4th Int. Conf. Lang. Resour. Eval.*, Lisbon, Portugal, European Language Resources Association, 2004, pp. 1–4.

[36] C. Dogan, A. Dutra, A. Gara, A. Gemma, L. Shi, M. Sigamani, and E. Walters, ''Fine-grained named entity recognition using ELMo and Wikidata,'' 2019, *arXiv:1904.10503*.

[37] L. Eikvil. (1993). *Optical Character Recognition*. [Online]. Available: https://citeseer.ist.psu.edu/142042.html

[38] D. W. Embley, M. Hurst, D. Lopresti, and G. Nagy, ''Table-processing paradigms: A research survey,'' *Int. J. Document Anal. Recognit.*, vol. 8, nos. 2–3, pp. 66–86, Jun. 2006.

[39] D. Esser, D. Schuster, K. Muthmann, M. Berger, and A. Schill, ''Automatic indexing of scanned documents: A layout-based approach,'' *Proc. SPIE*, vol. 8297, Jan.'2012, Art. no. 82970H.

[40] R. Evans and S. Street, ''A framework for named entity recognition in the open domain,'' *Recent Adv. Natural Lang. Process.*, vol. 260, nos. 267–274, p. 110, 2003.

[41] J. Fang, L. Gao, K. Bai, R. Qiu, X. Tao, and Z. Tang, ''A table detection method for multipage PDF documents via visual seperators and tabular structures,'' in *Proc. Int. Conf. Document Anal. Recognit.*, Beijing, China, Sep. 2011, pp. 779–783.

[42] E. Francesconi, S. Montemagni, W. Peters, and D. Tiscornia, *Semantic Processing of Legal Texts: Where the Language of Law Meets the Law of Language*, vol. 6036, Springer, 2010.

[43] S. Francis, J. V. Landeghem, and M.-F. Moens, ''Transfer learning for named entity recognition in financial and biomedical documents,'' *Information*, vol. 10, no. 8, p. 248, Jul. 2019.

[44] A. Gangal, P. Kumar, and S. Kumari, ''Complete scanning application using OpenCv,'' 2021, *arXiv:2107.03700*.

[45] L. Gao, Y. Huang, H. Déjean, J.-L. Meunier, Q. Yan, Y. Fang, F. Kleber, and E. Lang, ''ICDAR 2019 competition on table detection and recognition (cTDaR),'' in *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Sep. 2019, pp. 1510–1515.

[46] A. Gemelli, E. Vivoli, and S. Marinai, ''Graph neural networks and representation embedding for table extraction in PDF documents,'' in *Proc. 26th Int. Conf. Pattern Recognit. (ICPR)*, Montreal, QC, Canada, Aug. 2022, pp. 1719–1726.

[47] A. Gilani, S. R. Qasim, I. Malik, and F. Shafait, ''Table detection using deep learning,'' in *Proc. 14th IAPR Int. Conf. Document Anal. Recognit. (ICDAR)*, vol. 1, Nov. 2017, pp. 771–776.

[48] M. Göbel, T. Hassan, E. Oro, G. Orsi, and R. Rastan, ''Table modelling, extraction and processing,'' in *Proc. ACM Symp. Document Eng.*, R. Sablatnig and T. Hassan, Eds., Sep. 2016, pp. 1–2.

[49] V. Govindaraju, P. Natarajan, S. Chaudhury, and D. P. Lopresti, *Proceedings of the International Workshop on Multilingual OCR*. Barcelona, Spain: ACM, Jul. 2009.

[50] H. T. Ha and A. Horák, ''Information extraction from scanned invoice images using text analysis and layout features,'' *Signal Process., Image Commun.*, vol. 102, Mar. 2022, Art. no. 116601.

[51] K. A. Hambarde and H. Proenca, ''Information retrieval: Recent advances and beyond,'' 2023, *arXiv:2301.08801*.

[52] A. Hamdi, E. Carel, A. Joseph, M. Coustaty, and A. Doucet, ''Information extraction from invoices,'' in *Proc. Int. Conf. Document Anal. Recognit.*, in Lecture Notes in Computer Science, vol. 12822, J. Lladós, D. Lopresti, and S. Uchida, Eds., Springer, 2021, pp. 699–714.

[53] A. W. Harley, A. Ufkes, and K. G. Derpanis, ''Evaluation of deep convolutional nets for document image classification and retrieval,'' in *Proc. 13th Int. Conf. Document Anal. Recognit. (ICDAR)*, Nancy, France, Aug. 2015, pp. 991–995.

[54] C. He, S. Chen, S. Huang, J. Zhang, and X. Song, ''Using convolutional neural network with BERT for intent determination,'' in *Proc. Int. Conf. Asian Lang. Process. (IALP)*, Nov. 2019, pp. 65–70.

[55] K. He, Q. Yang, L. Ji, J. Pan, and Y. Zou, ''Financial time series forecasting with the deep learning ensemble model,'' *Mathematics*, vol. 11, no. 4, p. 1054, Feb. 2023.

[56] D. Hollingsworth, ''The workflow reference model,'' Workflow Manag. Coalition, Tech. Rep., 1994.

[57] Y. Huang, T. Lv, L. Cui, Y. Lu, and F. Wei, ''LayoutLMv3: Pre-training for document AI with unified text and image masking,'' in *Proc. 30th ACM Int. Conf. Multimedia*, Lisboa, Portugal, J. MagalhÃes, A. D. Bimbo, S. Satoh, N. Sebe, X. Alameda-Pineda, Q. Jin, V. Oria, and L. Toni, Eds., Oct. 2022, pp. 4083–4091.

[58] Z. Huang, K. Chen, J. He, X. Bai, D. Karatzas, S. Lu, and C. V. Jawahar, ''ICDAR2019 competition on scanned receipt OCR and information extraction,'' in *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Sep. 2019, pp. 1516–1520.

[59] N. Islam, Z. Islam, and N. Noor, ''A survey on optical character recognition system,'' *J. Inf. Commun. Technol.*, 2017.

[60] G. Jaume, H. K. Ekenel, and J.-P. Thiran, ''FUNSD: A dataset for form understanding in noisy scanned documents,'' in *Proc. 2nd Int. Conf. Document Anal. Recognit. Workshops (ICDARW)*, vol. 2, Sydney, NSW, Australia, Sep. 2019, pp. 1–6.

[61] P. Jha and G. Nagy, ''Wang notation tool: Layout independent representation of tables,'' in *Proc. 19th Int. Conf. Pattern Recognit.*, Dec. 2008, pp. 1–4.

[62] Y. Jiang, H. Dong, and A. El Saddik, ''Baidu meizu deep learning competition: Arithmetic operation recognition using end-to-end learning OCR technologies,'' *IEEE Access*, vol. 6, pp. 60128–60136, 2018.

[63] R. H. Sprague, ''Electronic document management: Challenges and opportunities for information systems managers,'' *MIS Quart.*, vol. 19, no. 1, pp. 29–49, Mar. 1995.

[64] M. Kasem, A. Abdallah, A. Berendeyev, E. Elkady, M. Abdalla, M. Mahmoud, M. Hamada, D. Nurseitov, and I. Taj-Eddin, ''Deep learning for table detection and structure recognition: A survey,'' 2022, *arXiv:2211.08469*.

[65] T. Kashinath, T. Jain, Y. Agrawal, T. Anand, and S. Singh, ''End-to-end table structure recognition and extraction in heterogeneous documents,'' *Appl. Soft Comput.*, vol. 123, Jul. 2022, Art. no. 108942.

[66] S. Kaur, S. Bawa, and R. Kumar, ''A survey of mono- and multi-lingual character recognition using deep and shallow architectures: Indic and non-indic scripts,'' *Artif. Intell. Rev.*, vol. 53, no. 3, pp. 1813–1872, Mar. 2020.

[67] I. Kavasidis, C. Pino, S. Palazzo, F. Rundo, D. Giordano, P. Messina, and C. Spampinato, ''A saliency-based convolutional neural network for table and chart detection in digitized documents,'' in *Image Analysis and Processing—ICIAP*, E. Ricci, S. R. Bulò, C. Snoek, O. Lanz, S. Messelodi, and N. Sebe, Eds. Cham, Switzerland: Springer, 2019, pp. 292–302.

[68] S. A. Khan, S. M. D. Khalid, M. A. Shahzad, and F. Shafait, ''Table structure extraction with bi-directional gated recurrent unit networks,'' in *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Sep. 2019, pp. 1366–1371.

[69] T. Kieninger and A. Dengel, ''The T-RECS table recognition and analysis system,'' in *Proc. 3rd Workshop Document Anal., Theory Pract.*, in Lecture Notes in Computer Science, vol. 1655, Nagano, Japan, S.-W. Lee and Y. Nakano, Eds. Berlin, Germany: Springer, 1998, pp. 255–269.

[70] G. Kim, T. Hong, M. Yim, J. Nam, J. Park, J. Yim, W. Hwang, S. Yun, D. Han, and S. Park, ''OCR-free document understanding transformer,'' in *Proc. 17th Eur. Conf. Comput. Vis.*, in Lecture Notes in Computer Science, vol. 13688, Tel Aviv, Israel, S. Avidan, G. J. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, Eds., Springer, 2022, pp. 498–517.

[71] M. Köppen, D. Waldöstl, and B. Nickolay, ''A system for the automated evaluation of invoices,'' in *Document Analysis Systems* (Series in Machine Perception and Artificial Intelligence), vol. 29, J. J. Hull and S. L. Taylor, Eds. Singapore: World Scientific, 1996, pp. 223–241.

[72] S. N. Srihari and S. W. Lam, ''Character recognition,'' *IETE J. Educ.*, vol. 17, no. 3, pp. 154–156, 1976.

[73] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, ''Neural architectures for named entity recognition,'' in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, San Diego, CA, USA, K. Knight, A. Nenkova, and O. Rambow, Eds., Association for Computational Linguistics, 2016, pp. 260–270.

[74] E. Lee, J. Park, H. I. Koo, and N. I. Cho, ''Deep-learning and graph-based approach to table structure recognition,'' *Multimedia Tools Appl.*, vol. 81, no. 4, pp. 5827–5848, Feb. 2022.

[75] J. Li, A. Sun, J. Han, and C. Li, ''A survey on deep learning for named entity recognition,'' *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 1, pp. 50–70, Jan. 2022.

[76] M. Li, L. Cui, S. Huang, F. Wei, M. Zhou, and Z. Li, ''TableBank: Table benchmark for image-based table detection and recognition,'' in *Proc. 12th Lang. Resour. Eval. Conf.*, Marseille, France, European Language Resources Association, May 2020, pp. 1918–1925.

[77] M. Li, T. Lv, J. Chen, L. Cui, Y. Lu, D. Florencio, C. Zhang, Z. Li, and F. Wei, ''TrOCR: Transformer-based optical character recognition with pre-trained models,'' in *Proc. AAAI Conf. Artif. Intell.*, vol. 37, 2023, pp. 13094–13102.

[78] Y. Li, Z. Huang, J. Yan, Y. Zhou, F. Ye, and X. Liu, ''GFTE: Graph-based financial table extraction,'' in *Proc. Int. Conf. Pattern Recognit.* Cham, Switzerland: Springer, 2021, pp. 644–658.

[79] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, ''A survey of convolutional neural networks: Analysis, applications, and prospects,'' *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 12, pp. 6999–7019, Dec. 2022.

[80] Y. Liu, K. Bai, P. Mitra, and C. L. Giles, ''TableSeer: Automatic table metadata extraction and searching in digital libraries,'' in *Proc. 7th ACM/IEEE-CS joint Conf. Digit. Libraries*, Jun. 2007, pp. 91–100.

[81] W. Lovegrove, ''Advanced document analysis and automatic classification of PDF documents,'' M.S. thesis, Univ. Nottingham, Nottingham, U.K., 1996.

[82] N. Tuan Ly, A. Takasu, P. Nguyen, and H. Takeda, ''Rethinking image-based table recognition using weakly supervised methods,'' 2023, *arXiv:2303.07641*.

[83] J. Mantas, ''An overview of character recognition methodologies,'' *Pattern Recognit.*, vol. 19, no. 6, pp. 425–430, Jan. 1986.

[84] N. Mariella and A. Simonetto, ''A quantum algorithm for the sub-graph isomorphism problem,'' *ACM Trans. Quantum Comput.*, vol. 4, no. 2, pp. 1–34, Jun. 2023.

[85] E. Marsh and D. Perzanowski, ''MUC-7 evaluation of IE technology: Overview of results,'' in *Proc. Conf. 7th Message Understand.*, Fairfax, Virginia, May 1998.

[86] D. Marx and M. Pilipczuk, ''Everything you always wanted to know about the parameterized complexity of subgraph isomorphism (but were afraid to ask),'' in *Proc. 31st Int. Symp. Theor. Aspects Comput. Sci.*, 2014, p. 542.

[87] C. McCreesh, P. Prosser, and J. Trimble, ''The Glasgow Subgraph solver: Using constraint programming to tackle hard subgraph isomorphism problem variants,'' in *Proc. Int. Conf. Graph Transformation*. Cham, Switzerland: Springer, 2020, pp. 316–324.

[88] E. Medvet, A. Bartoli, and G. Davanzo, ''A probabilistic approach to printed document understanding,'' *Int. J. Document Anal. Recognit.*, vol. 14, no. 4, pp. 335–347, Dec. 2011.

[89] J. Memon, M. Sami, R. A. Khan, and M. Uddin, ''Handwritten optical character recognition (OCR): A comprehensive systematic literature review (SLR),'' *IEEE Access*, vol. 8, pp. 142642–142668, 2020.

[90] T. Mikolov, K. Chen, G. Corrado, and J. Dean, ''Efficient estimation of word representations in vector space,'' in *Proc. 1st Int. Conf. Learn. Represent.*, Scottsdale, AZ, USA, Y. Bengio and Y. LeCun, Eds., 2013.

[91] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, ''Distributed representations of words and phrases and their compositionality,'' in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 26, 2013, pp. 3111–3119.

[92] R. Mithe, S. Indalkar, and N. Divekar, ''Optical character recognition,'' *Int. J. Recent Technol. Eng.*, vol. 2, no. 1, pp. 72–75, 2013.

[93] S. Mori, C. Y. Suen, and K. Yamamoto, ''Historical review of OCR research and development,'' *Proc. IEEE*, vol. 80, no. 7, pp. 1029–1058, Jul. 1992.

[94] A. Moussaoui, ''Geometric constraint solver,'' M.S. thesis, Ecole Nationale Supérieure d'Informatique, 2016.

[95] D. Nadeau and S. Sekine, ''A survey of named entity recognition and classification,'' *Lingvisticæ Investigationes*, vol. 30, no. 1, pp. 3–26, Jan. 2007.

[96] T. Nasir, M. K. Malik, and K. Shahzad, ''MMU-OCR-21: Towards end-to-end Urdu text recognition using deep learning,'' *IEEE Access*, vol. 9, pp. 124945–124962, 2021.

[97] M. Netter, E. B. Fernandez, and G. Pernul, ''Refining the pattern-based reference model for electronic invoices by incorporating threats,'' in *Proc. Int. Conf. Availability, Rel. Secur.*, Krakow, Poland, Feb. 2010, pp. 560–564.

[98] C. Neudecker, K. Baierer, M. Gerber, C. Clausner, A. Antonacopoulos, and S. Pletschacher, ''A survey of OCR evaluation tools and metrics,'' in *Proc. 6th Int. Workshop Historical Document Imag. Process.*, Lausanne, Switzerland, A. Antonacopoulos, C. Clausner, M. Ehrmann, C. Neudecker, and S. Pletschacher, Eds., 2021, pp. 13–18.

[99] K. Nishida, K. Sadamitsu, R. Higashinaka, and Y. Matsuo, ''Understanding the semantic structures of tables with a hybrid deep neural network architecture,'' in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017.

[100] S. S. Paliwal, D. Vishwanath, R. Rahul, M. Sharma, and L. Vig, ''TableNet: Deep learning model for end-to-end table detection and tabular data extraction from scanned document images,'' in *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Sydney, NSW, Australia, Sep. 2019, pp. 128–133.

[101] S. Park, S. Shin, B. Lee, J. Lee, J. Surh, M. Seo, and H. Lee, ''CORD: A consolidated receipt dataset for post-OCR parsing,'' in *Proc. Workshop Document Intell. (NeurIPS)*, 2019.

[102] S. Patel and D. Bhatt, ''Abstractive information extraction from scanned invoices (AIESI) using end-to-end sequential approach,'' 2020, *arXiv:2009.05728*.

[103] M. O. Perez-Arriaga, T. Estrada, and S. Abad-Mota, ''TAO: System for table detection and extraction from PDF documents,'' in *Proc. 29th Int. Flairs Conf.*, 2016.

[104] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, ''Deep contextualized word representations,'' in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Human Lang. Technol.*, New Orleans, LA, USA, M. A. Walker, H. Ji, and A. Stent, Eds., Association for Computational Linguistics, 2018, pp. 2227–2237.

[105] D. Pinto, A. McCallum, X. Wei, and W. B. Croft, ''Table extraction using conditional random fields,'' in *Proc. 26th Annu. Int. ACM SIGIR Conf. Res. Develop. informaion Retr.*, Jul. 2003, pp. 235–242.

[106] G. Popovski, B. K. Seljak, and T. Eftimov, ''A survey of named-entity recognition methods for food information extraction,'' *IEEE Access*, vol. 8, pp. 31586–31594, 2020.

[107] D. Prasad, A. Gadpal, K. Kapadni, M. Visave, and K. Sultanpure, ''CascadeTabNet: An approach for end to end table detection and structure recognition from image-based documents,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Seattle, WA, USA, Jun. 2020, pp. 2439–2447.

[108] S. R. Qasim, H. Mahmood, and F. Shafait, ''Rethinking table recognition using graph neural networks,'' in *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Sydney, NSW, Australia, Sep. 2019, pp. 142–147.

[109] L. Qiao, Z. Li, Z. Cheng, P. Zhang, S. Pu, Y. Niu, W. Ren, W. Tan, and F. Wu, ''LGPMA: Complicated table structure recognition with local and global pyramid mask alignment,'' in *Proc. 16th Int. Conf. Document Anal. Recognit.*, in Lecture Notes in Computer Science, vol. 12821, Lausanne, Switzerland, J. Lladós, D. Lopresti, and S. Uchida, Eds. Cham, Switzerland: Springer, 2021, pp. 99–114.

[110] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, ''Improving language understanding by generative pre-training,'' OpenAI, Tech. Rep., 2018.

[111] S. Raja, A. Mondal, and C. V. Jawahar, ''Table structure recognition using top-down and bottom-up cues,'' in *Proc. 16th Eur. Conf. Comput. Vis.*, in Lecture Notes in Computer Science, vol. 12373, Glasgow, U.K., A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds., Springer, 2020, pp. 70–86.

[112] R. Rastan, H.-Y. Paik, J. Shepherd, S. H. Ryu, and A. Beheshti, ''TEXUS: Table extraction system for PDF documents,'' in *Proc. 29th Australas. Database Conf. Databases Theory Appl.*, in Lecture Notes in Computer Science, vol. 10837, Gold Coast, QLD, Australia, J. Wang, G. Cong, J. Chen, and J. Qi, Eds. Cham, Switzerland: Springer, 2018, pp. 345–349.

[113] P. Riba, A. Dutta, L. Goldmann, A. Fornés, O. Ramos, and J. Lladós, ''Table detection in invoice documents by graph neural networks,'' in *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Sep. 2019, pp. 122–127.

[114] M. R. Makiuchi, T. Warnita, K. Uto, and K. Shinoda, ''Multimodal fusion of BERT-CNN and gated CNN representations for depression detection,'' in *Proc. 9th Int. Audio/Vis. Emotion Challenge Workshop*, 2019, pp. 55–63.

[115] A. Safaya, M. Abdullatif, and D. Yuret, ''KUISAIL at SemEval-2020 task 12: BERT-CNN for offensive speech identification in social media,'' in *Proc. 14th Workshop Semantic Eval.*, 2020, pp. 2054–2059.

[116] K. C. Santosh and A. Belaïd, ''Pattern-based approach to table extraction,'' in *Proc. Iberian Conf. Pattern Recognit. Image Anal.* Berlin, Germany: Springer, 2013, pp. 766–773.

[117] T. Saout, F. Lardeux, and F. Saubion, ''A two-stage approach for table extraction in invoices,'' 2022, *arXiv:2210.04716*.

[118] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, ''The graph neural network model,'' *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Jan. 2009.

[119] S. Schreiber, S. Agne, I. Wolf, A. Dengel, and S. Ahmed, ''DeepDeSRT: Deep learning for detection and structure recognition of tables in document images,'' in *Proc. 14th IAPR Int. Conf. Document Anal. Recognit. (ICDAR)*, vol. 1, Kyoto, Japan, Nov. 2017, pp. 1162–1167.

[120] S. Seth and G. Nagy, ''Segmenting tables via indexing of value cells by table headers,'' in *Proc. 12th Int. Conf. Document Anal. Recognit.*, Aug. 2013, pp. 887–891.

[121] F. Shafait and R. Smith, ''Table detection in heterogeneous documents,'' in *Proc. 9th Int. Workshop Document Anal. Syst.*, S. D. David, G. Venu, P. L. aniel, and N. Premkumar, Eds., Jun. 2010.

[122] A. Shapenko, V. Korovkin, and B. Leleux, ''ABBYY: The digitization of language and text,'' *Emerald Emerg. Markets Case Stud.*, vol. 8, no. 2, pp. 1–26, Jun. 2018.

[123] A. Shigarov, A. Altaev, A. Mikhailov, V. Paramonov, and E. Cherkashin, ''TabbyPDF: Web-based system for PDF table extraction,'' in *Proc. ICIST*, in Communications in Computer and Information Science, D. Robertas and G. Vasiljeviene, Eds., 2018.

[124] S. A. Siddiqui, I. A. Fateh, S. T. R. Rizvi, A. Dengel, and S. Ahmed, ''DeepTabStR: Deep learning based table structure recognition,'' in *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Sydney, NSW, Australia, Sep. 2019, pp. 1403–1409.

[125] S. K. Sienčnik, ''Adapting word2vec to named entity recognition,'' in *Proc. 20th Nordic Conf. Comput. Linguistics*, 2015, pp. 239–243.

[126] R. Smith, ''An overview of the tesseract OCR engine,'' in *Proc. 9th ICDAR*, Sep. 2007, pp. 629–633.

[127] C. Solnon, ''Experimental evaluation of subgraph isomorphism solvers,'' in *Proc. Int. Workshop Graph-Based Represent. Pattern Recognit.* Berlin, Germany: Springer, 2019, pp. 1–13.

[128] E. Sorio, A. Bartoli, G. Davanzo, and E. Medvet, ''Open world classification of printed invoices,'' in *Proc. 10th ACM Symp. Document Eng.*, Manchester, U.K., Sep. 2010, pp. 187–190.

[129] P. J. O. Suárez, Y. Dupont, B. Müller, L. Romary, and B. Sagot, ''Establishing a new state-of-the-art for French named entity recognition,'' in *Proc. 12th Lang. Resour. Eval. Conf.*, 2020.

[130] Y. Sun, X. Mao, S. Hong, W. Xu, and G. Gui, ''Template matching-based method for intelligent invoice information identification,'' *IEEE Access*, vol. 7, pp. 28392–28401, 2019.

[131] A. S. Tarawneh, A. B. Hassanat, D. Chetverikov, I. Lendak, and C. Verma, ''Invoice classification using deep features and machine learning techniques,'' in *Proc. IEEE Jordan Int. Joint Conf. Electr. Eng. Inf. Technol. (JEEIT)*, Apr. 2019, pp. 855–859.

[132] S. L. Taylor, R. Fritzson, and J. A. Pastor, ''Extraction of data from preprinted forms,'' *Mach. Vis. Appl.*, vol. 5, no. 3, pp. 211–222, Jun. 1992.

[133] C. Tensmeyer, V. I. Morariu, B. Price, S. Cohen, and T. Martinez, ''Deep splitting and merging for table structure decomposition,'' in *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Sydney, NSW, Australia, Sep. 2019, pp. 114–121.

[134] E. F. T. K. Sang and F. De Meulder, ''Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition,'' in *Proc. 7th Conf. Natural Lang. Learn. (HLT-NAACL)*, 2003, pp. 142–147.

[135] P. Vassiliadis and A. Simitsis, ''Extraction, transformation, and loading,'' *Encyclopedia Database Syst.*, Oct. 2009.

[136] N. L. Vine, M. Zeigenfuse, and M. Rowan, ''Extracting tables from documents using conditional generative adversarial networks and genetic algorithms,'' in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Budapest, Hungary, Jul. 2019, pp. 1–8.

[137] J. Voerman, A. Joseph, M. Coustaty, V. P. d'Andecy, and J. M. Ogier, ''Evaluation of neural network classification systems on document stream,'' in *Proc. 14th Int. Workshop Document Anal. Syst.*, in Lecture Notes in Computer Science, vol. 12116, Wuhan, China, X. Bai, D. Karatzas, and D. Lopresti, Eds. Cham, Switzerland: Springer, 2020, pp. 262–276.

[138] Q. Wang and M. Iwaihara, ''Deep neural architectures for joint named entity recognition and disambiguation,'' in *Proc. IEEE Int. Conf. Big Data Smart Comput. (BigComp)*, Kyoto, Japan, Feb. 2019, pp. 1–4.

[139] Y. Wangt, I. T. Phillipst, and R. Haralick, ''Automatic table ground truth generation and a background-analysis-based table structure extraction method,'' in *Proc. 6th Int. Conf. Document Anal. Recognit.*, Sep. 2001, pp. 528–532.

[140] Z. Xiao, H. Zhang, H. Tong, and X. Xu, ''An efficient temporal network with dual self-distillation for electroencephalography signal classification,'' in *Proc. IEEE Int. Conf. Bioinf. Biomed. (BIBM)*, Dec. 2022, pp. 1759–1762.

[141] H. Xing, Z. Xiao, R. Qu, Z. Zhu, and B. Zhao, ''An efficient federated distillation learning system for multitask time series classification,'' *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–12, 2022.

[142] H. Xing, Z. Xiao, D. Zhan, S. Luo, P. Dai, and K. Li, ''SelfMatch: Robust semisupervised time-series classification with self-distillation,'' *Int. J. Intell. Syst.*, vol. 37, no. 11, pp. 8583–8610, Nov. 2022.

[143] W. Xue, B. Yu, W. Wang, D. Tao, and Q. Li, ''TGRNet: A table graph reconstruction network for table structure recognition,'' in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Montreal, QC, Canada, Oct. 2021, pp. 1275–1284.

[144] V. Yadav and S. Bethard, ''A survey on recent advances in named entity recognition from deep learning models,'' in *Proc. 27th Int. Conf. Comput. Linguistics*, 2018, pp. 2145–2158.

[145] L. Yao, S. Riedel, and A. McCallum, ''Collective cross-document relation extraction without labelled data,'' in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2010, pp. 1013–1023.

[146] Q. Ye and D. Doermann, ''Text detection and recognition in imagery: A survey,'' *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 7, pp. 1480–1500, Jul. 2015.

[147] Z. Ye, Y. J. Kumar, G. O. Sing, F. Song, and J. Wang, ''A comprehensive survey of graph neural networks for knowledge graphs,'' *IEEE Access*, vol. 10, pp. 75729–75741, 2022.

[148] B. Yildiz, K. Kaiser, and S. Miksch, ''pdf2table: A method to extract table information from PDF files,'' in *Proc. IICAI*, 2005, pp. 1773–1785.

[149] R. Zanibbi, D. Blostein, and J. Cordy, ''A survey of table recognition,'' *Int. J. Document Anal. Recognit.*, vol. 7, no. 1, Mar. 2004.

[150] L. Zhang and H. Zhao, ''Named entity recognition for Chinese microblog with convolutional neural network,'' in *Proc. 13th Int. Conf. Natural Comput., Fuzzy Syst. Knowl. Discovery (ICNC-FSKD)*, Guilin, China, Jul. 2017, pp. 87–92.

[151] M. Zhang, D. Perelman, V. Le, and S. Gulwani, ''An integrated approach of deep learning and symbolic analysis for digital PDF table extraction,'' in *Proc. 25th Int. Conf. Pattern Recognit. (ICPR)*, Milan, Italy, Jan. 2021, pp. 4062–4069.

[152] P. Zhang, C. Li, L. Qiao, Z. Cheng, S. Pu, Y. Niu, and F. Wu, ''VSR: A unified framework for document layout analysis combining vision, semantics and relations,'' in *Proc. 16th Int. Conf. Document Anal. Recognit.*, in Lecture Notes in Computer Science, vol. 12821, Lausanne, Switzerland, J. Lladós, D. Lopresti, and S. Uchida, Eds., Springer, 2021, pp. 115–130.

[153] K. Zheng, L. Sun, X. Wang, S. Zhou, H. Li, S. Li, L. Zeng, and Q. Gong, ''Named entity recognition in electric power metering domain based on attention mechanism,'' *IEEE Access*, vol. 9, pp. 152564–152573, 2021.

[154] X. Zheng, D. Burdick, L. Popa, X. Zhong, and N. X. R. Wang, ''Global table extractor (GTE): A framework for joint table identification and cell structure recognition using visual context,'' in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Waikoloa, HI, USA, Jan. 2021, pp. 697–706.

[155] X. Zhong, E. ShafieiBavani, and A. J. Yepes, ''Image-based table recognition: Data, model, and evaluation,'' in *Proc. 16th Eur. Conf. Comput. Vis.*, in Lecture Notes in Computer Science, vol. 12366, Glasgow, U.K., A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds. Cham, Switzerland: Springer, 2020, pp. 564–580.

[156] X. Zhong, J. Tang, and A. Jimeno Yepes, ''PubLayNet: Largest dataset ever for document layout analysis,'' in *Proc. Int. Conf. Document Anal. Recognit. (ICDAR)*, Sep. 2019, pp. 1015–1022.

THOMAS SAOUT was born in Brest, France, in 1992. He received the M.S. degree in computer science, specializing in decision intelligence from the University of Angers, in 2020, where he is currently pursuing the Ph.D. degree with LERIA. He was a JAVA Developer with KS2, a French company that produces ERP solutions, for six months. His research interests include evolutionary algorithms, information retrieval, natural language processing, and graph pattern recognition.

FRÉDÉRIC LARDEUX was born in France, in 1979. He received the M.S. and Ph.D. degrees in computer science from the University of Angers, France, in 2002 and 2005, respectively.

Since 2006, he has been a Professor with the LERIA, University of Angers. His research interests include constraints (CSP and SAT), model transformations, combinatorial optimization, metaheuristics, evolutionary computation, learning (reinforcement learning and machine learning), and logical analysis of data.

FRÉDÉRIC SAUBION received the M.S. and Ph.D. degrees in computer science from the University of Orléans, France, in 1996.

From 1997 to 2003, he was an Assistant Professor with the University of Angers, France. Since 2004, he has been a Full Professor with the Faculty of Science, University of Angers. He has supervised a dozen of Ph.D. students. He has contributed to the autonomous search paradigm that consists in improving the automated setting and control of solving algorithms, in particular thanks to machine learning techniques. He has also investigated different application domains (biology and information retrieval). His research interests include metaheuristics, evolutionary computation, and machine learning.