

## ****RECEIPT & INVOICE DIGITIZER – WHAT IT ACTUALLY MEAN :****

The project title **“Receipt & Invoice Digitizer”** clearly represents the main goal of the system.

* **Receipt** refers to a proof of payment document usually generated after a purchase (billing slips, store receipts, etc.).
* **Invoice** refers to a formal billing document generated by sellers/service providers for transactions.
* **Digitizer** means converting physical or scanned documents (paper/image/PDF) into **digital structured data**.

So, **Receipt & Invoice Digitizer** is a system that takes receipt/invoice images or PDFs as input and automatically extracts key information (like date, amount, invoice number, vendor, items, tax, etc.), then converts it into a **machine-readable digital format** such as JSON/CSV/database records.

**INTRODUCTION :**

In today’s digital era, organizations and individuals handle a large number of receipts and invoices for expenses, billing, taxation, and financial record maintenance. However, most receipts and invoices are still generated in paper format or as image/PDF files, making manual data entry time-consuming, error-prone, and inefficient.

The **Receipt & Invoice Digitizer** project is designed to solve this problem by automatically extracting important information from receipts and invoices such as **invoice number, date, vendor name, item details, quantity, total amount, and tax information**.

This system converts unstructured receipt/invoice images into structured digital data that can be stored, searched, and analyzed easily.

By using modern technologies like **OCR (Optical Character Recognition)** and **AI-based text processing**, the proposed solution aims to improve speed, accuracy, and efficiency in document handling. It reduces human effort, enables faster financial processing, and supports better decision-making by maintaining clean and digitized records.

**PROJECT STATEMENT :**

Businesses and individuals handle numerous paper receipts and invoices, which are prone to loss, errors, and manual entry delays. This project builds a system that automatically scans, extracts, and digitizes information from receipts and invoices using OCR (Optical Character Recognition) and NLP-based field extraction. The digitized data is stored in a structured format, making it easy to search, analyze, and integrate with accounting or ERP systems.

**OBJECTIVES :**

1. **File Upload Module**

Enable users to upload receipts/invoices in formats like **JPG, PNG, JPEG, and PDF**.

1. **Document Ingestion Pipeline**

Store uploaded files safely for processing (local folder / cloud storage).

1. **Image Preprocessing for OCR**

Improve input quality using preprocessing techniques such as:

* + - resizing,
    - grayscale conversion,
    - noise removal,
    - thresholding / binarization,
    - sharpening / contrast enhancement.

1. **OCR Text Extraction**

Extract **raw text** from uploaded sample receipts/invoices using OCR engine.

1. **Initial Output Display**

Show extracted raw text to verify correctness and readability.

## ****SCOPE :****

* Upload receipt/invoice images and PDFs.
* Basic preprocessing to improve OCR accuracy.
* OCR extraction of raw text.
* Display extracted text result on UI (web app)

## ****SYSTEM ARCHITECHTURE :****

## 

## ****1) File Upload Module****

The File Upload Module is the first component of the Receipt & Invoice Digitizer system and is responsible for collecting input documents from the user. In this module, the user uploads receipts or invoices in supported formats such as JPG, PNG, JPEG (and PDF if applicable). The system validates the uploaded file to ensure it matches the accepted format and avoids corrupted or unsupported inputs.

After validation, the uploaded document is stored in a temporary/local storage location so it can be accessed by the preprocessing and OCR stages. This module ensures smooth document ingestion and acts as the foundation for the complete OCR pipeline.

## ****2) Image Preprocessing Module****

The Image Preprocessing Module improves the quality of the uploaded receipt or invoice image to achieve better OCR accuracy. Since receipts are often captured using mobile cameras or scanners, the images may contain noise, blur, low brightness, shadows, skew, or uneven contrast. This module applies preprocessing operations such as resizing, grayscale conversion, denoising, thresholding (binarization), and contrast enhancement to make the printed text clearer and more distinguishable from the background.

In some cases, additional processing like deskewing can also be used to correct tilted documents. As a result, the preprocessing module generates an OCR-ready image that significantly increases the reliability of text extraction.

## ****3) OCR Extraction Module****

The OCR Extraction Module is the core part of milestone-1, where the system converts the preprocessed receipt/invoice image into machine-readable text. After receiving the enhanced image from the preprocessing stage, an OCR engine such as Tesseract OCR or EasyOCR detects characters and words from the document and extracts them as raw text output.

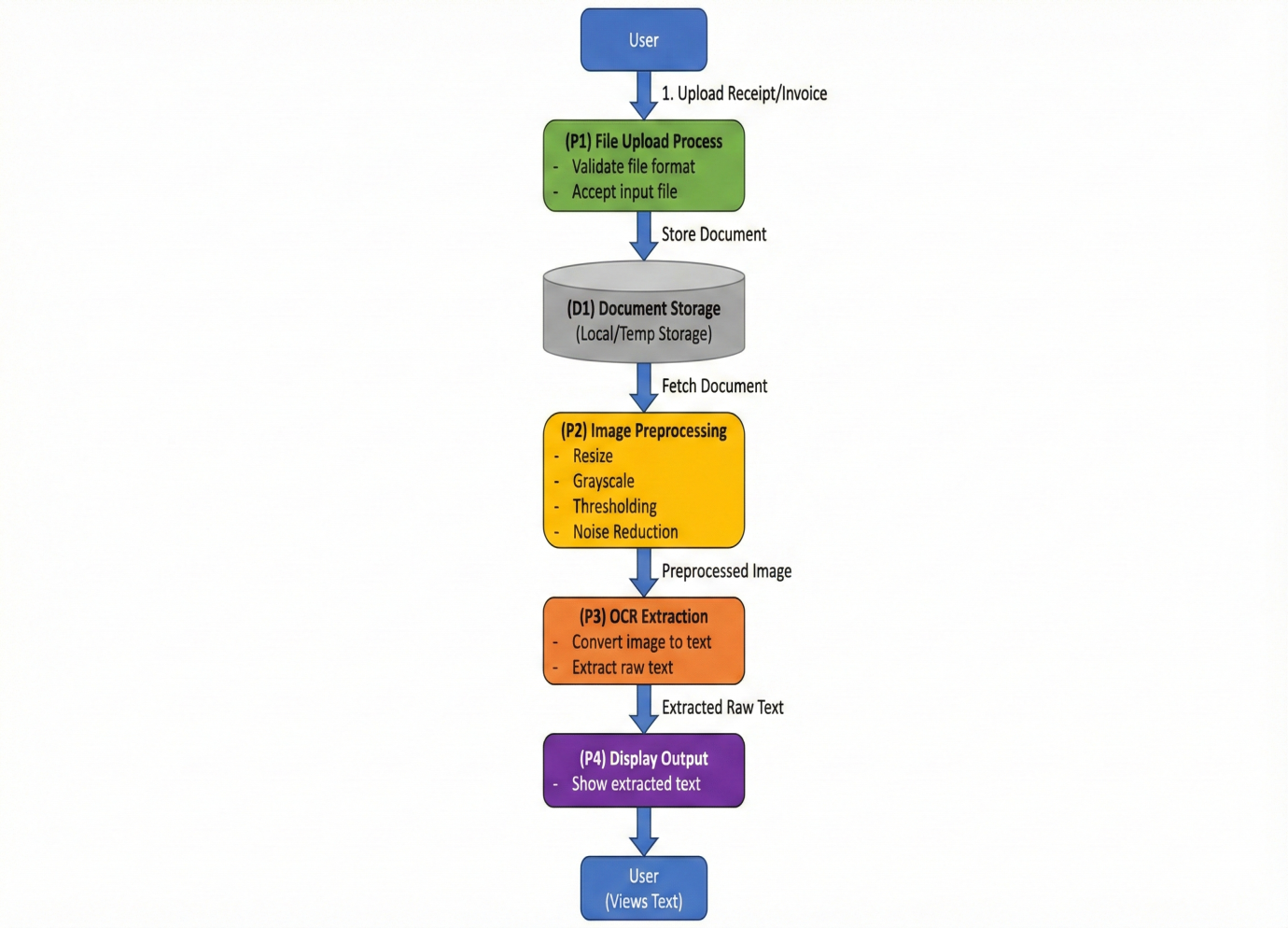
The extracted content may include vendor name, transaction details, item list, quantities, tax values, total amount, and other printed information present on the receipt/invoice. At this milestone stage, the extracted text is not yet structured into fields, but it provides the required raw text foundation for later milestones like key-value extraction and data digitization.

## ****4) Output Display Module****

The Output Display Module presents the extracted OCR text to the user for verification and evaluation. Once the OCR module generates raw text, this module displays the output in a readable format on the user interface, allowing the user to check the extracted content and confirm its accuracy.

This is useful in milestone-1 because it helps in testing the OCR pipeline with different sample receipts and invoices and ensures that preprocessing is improving recognition quality. By showing the extracted results clearly, the module supports performance validation and forms the basis for future steps such as structured data extraction and storage.

**DATA FLOW DIAGRAM :**

****

The **DFD diagram** explains the complete working flow of Milestone–1 in the Receipt & Invoice Digitizer system. It starts with the **User**, who uploads a receipt or invoice document into the application.

This input is first handled by **Process P1 (File Upload Process)**, where the system accepts the file and validates whether the format is correct and supported. After successful validation, the uploaded file is saved in **Data Store D1 (Document Storage)**, which represents local or temporary storage used during processing.

Once stored, the system retrieves the document from storage and sends it to **Process P2 (Image Preprocessing)**, where the receipt image is enhanced through steps like resizing, grayscale conversion, thresholding, and noise reduction to improve text clarity. The preprocessed output is then passed to **Process P3 (OCR Extraction)**, where an OCR engine converts the enhanced image into machine-readable **raw text**.

Finally, this extracted text is forwarded to **Process P4 (Display Output)**, which shows the OCR result on the interface so the user can view and verify the extracted content. This diagram clearly represents the milestone-1 pipeline from document upload to OCR text output.

**TECHNOLOGY STACK :**

| **Technology / Tool** | **Purpose in Milestone–1** | **Why Used** |
| --- | --- | --- |
| **Python** | Core programming language for the entire pipeline. | Easy integration for OCR and image processing libraries. |
| **Streamlit (or Flask UI)** | Frontend interface for uploading receipts and showing OCR output. | Simple, fast, and interactive web UI development. |
| **OpenCV (cv2)** | Image preprocessing (grayscale, resizing, noise removal, thresholding). | Improves OCR accuracy by enhancing image quality. |
| **Tesseract OCR (pytesseract)** | Extracts raw text from preprocessed images. | Converts scanned images into machine-readable text. |
| **NumPy** | Handles image array operations during preprocessing. | Efficient numerical processing for OpenCV image data. |
| **Local Storage** | Stores uploaded files temporarily before processing. | Easy and lightweight storage for initial milestone testing. |
| **VS Code / Jupyter** | Development and testing environment. | Helps in rapid debugging and experimenting with OCR outputs. |

## ****ERROR HANDLING AND VALIDATION :****

In Milestone–1, proper error handling and validation mechanisms are implemented to ensure smooth document ingestion and reliable OCR execution. During the file upload stage, the system validates the uploaded document by checking the file type and allowing only supported formats such as JPG, PNG, JPEG (and PDF if enabled).

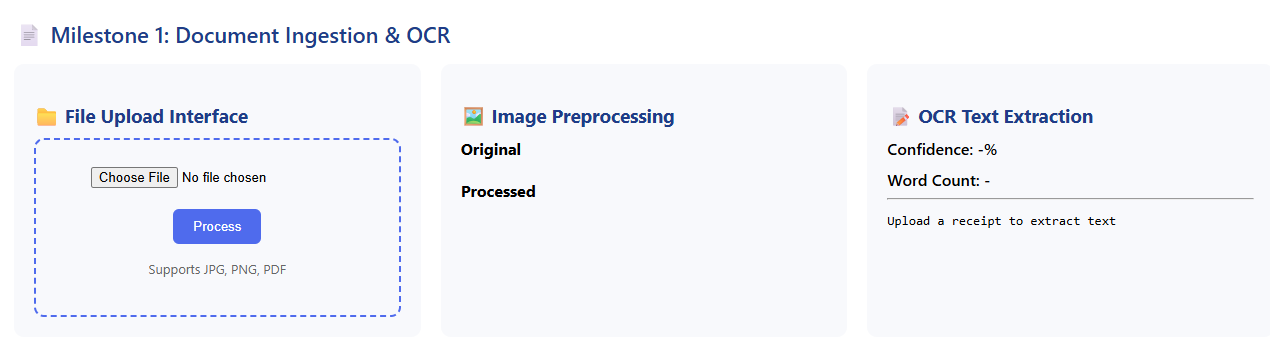
If an unsupported file type is uploaded, the application immediately rejects it and displays an appropriate error message to the user. File validation also includes checking whether a file is empty, corrupted, or unreadable, preventing the pipeline from crashing due to invalid inputs.

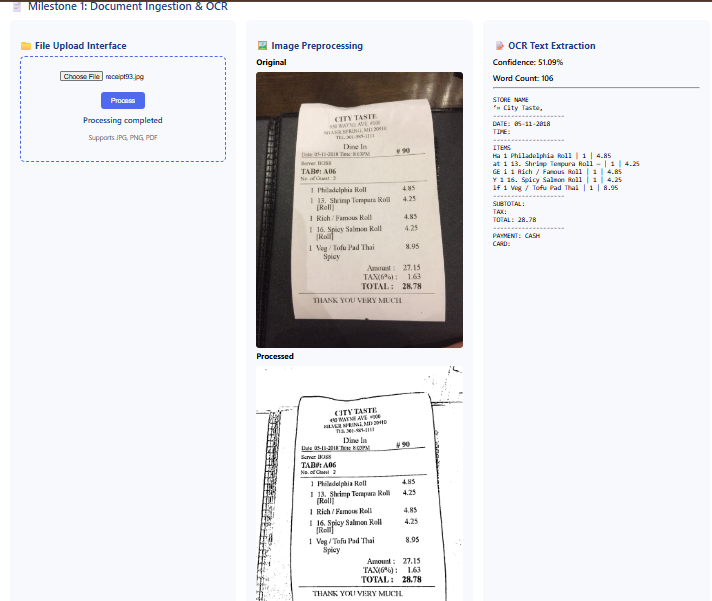
After upload, the system verifies successful storage of the document before forwarding it for preprocessing. In the preprocessing stage, validations are performed to confirm that the image is correctly loaded into memory and has valid dimensions; if image reading fails (for example, due to corrupted images or incomplete downloads), the system stops processing and notifies the user.

During OCR extraction, the system handles OCR failures such as unclear images, low resolution, heavy noise, and poor lighting by displaying a warning message when extracted text is insufficient or blank.

Additionally, exception handling is used throughout the pipeline to catch runtime errors, ensuring the system does not terminate unexpectedly and instead provides user-friendly feedback. These validation and error-handling practices improve system reliability and help ensure accurate OCR output during milestone testing.

**OUTPUT SCREENS :**

****

****

**CONCLUSION :**

In this milestone, the foundational stage of the **Receipt & Invoice Digitizer** project was successfully completed by implementing the end-to-end pipeline for **document ingestion and OCR processing**.

The system was able to accept receipt and invoice documents through a file upload interface, perform essential image preprocessing to enhance document quality, and extract raw text using an OCR engine.

This milestone validates that the application can reliably convert receipt/invoice images into machine-readable text, which serves as the base for future development. The outcomes of this milestone provide a strong foundation for upcoming phases such as structured key-value extraction, data validation, and storing digitized invoice/receipt details into databases for search and analytics.



## ****RECEIPT & INVOICE DIGITIZER – WHAT IT ACTUALLY MEAN :****

The project title **“Receipt & Invoice Digitizer”** clearly represents the main goal of the system.

* **Receipt** refers to a proof of payment document usually generated after a purchase (billing slips, store receipts, etc.).
* **Invoice** refers to a formal billing document generated by sellers/service providers for transactions.
* **Digitizer** means converting physical or scanned documents (paper/image/PDF) into **digital structured data**.

So, **Receipt & Invoice Digitizer** is a system that takes receipt/invoice images or PDFs as input and automatically extracts key information (like date, amount, invoice number, vendor, items, tax, etc.), then converts it into a **machine-readable digital format** such as JSON/CSV/database records.

**INTRODUCTION :**

In today’s digital era, organizations and individuals handle a large number of receipts and invoices for expenses, billing, taxation, and financial record maintenance. However, most receipts and invoices are still generated in paper format or as image/PDF files, making manual data entry time-consuming, error-prone, and inefficient.

The **Receipt & Invoice Digitizer** project is designed to solve this problem by automatically extracting important information from receipts and invoices such as **invoice number, date, vendor name, item details, quantity, total amount, and tax information**.

This system converts unstructured receipt/invoice images into structured digital data that can be stored, searched, and analyzed easily.

By using modern technologies like **OCR (Optical Character Recognition)** and **AI-based text processing**, the proposed solution aims to improve speed, accuracy, and efficiency in document handling. It reduces human effort, enables faster financial processing, and supports better decision-making by maintaining clean and digitized records.

**PROJECT STATEMENT :**

Businesses and individuals handle numerous paper receipts and invoices, which are prone to loss, errors, and manual entry delays. This project builds a system that automatically scans, extracts, and digitizes information from receipts and invoices using OCR (Optical Character Recognition) and NLP-based field extraction. The digitized data is stored in a structured format, making it easy to search, analyze, and integrate with accounting or ERP systems.

**OBJECTIVES :**

The primary objective of **Milestone–2** is to transform unstructured OCR text extracted from receipts and invoices into meaningful, structured, and analyzable data. In this phase, the system leverages a **Large Language Model (Groq LLM)** to intelligently parse raw OCR output and extract structured fields such as merchant name, transaction date, payment method, subtotal, tax, total amount, and item-level details. This approach enables accurate data extraction even when receipts have varying layouts, fonts, or formats.

Another key objective of this milestone is to ensure **data accuracy and consistency** through a comprehensive validation mechanism. The system applies multiple business rule validations, including subtotal and tax consistency checks, duplicate receipt detection, tax range verification, date format recognition, and mandatory field validation. These checks help identify incorrect or incomplete extractions and improve the reliability of stored data.

Milestone–2 also focuses on **persistent data storage** by saving validated receipt information into a **SQLite database**. This allows the system to maintain a permanent record of all processed receipts, supporting features such as receipt history, filtering, deletion, and retrieval for future reference.

Finally, this milestone aims to enable **history tracking and analytics visualization**. The stored receipt data is used to generate meaningful insights such as merchant-wise spending distribution and time-based expense trends through visual charts. This objective enhances the usability of the system by converting digitized receipts into actionable financial insights, laying the groundwork for advanced analytics and future system enhancements.

## ****SYSTEM ARCHITECHTURE :****

## 

The system architecture for **Milestone–2** extends the basic OCR pipeline developed in Milestone–1 by adding intelligent data extraction, validation, persistent storage, and analytics. The process begins when the **user uploads a receipt or invoice** through the web application built using **Streamlit or Flask UI**. The uploaded file is first handled by the **File Upload Module**, which accepts supported formats such as JPG, PNG, and PDF and validates the file type to prevent invalid inputs.

After validation, the uploaded document is saved in **local or temporary document storage**, ensuring the file is available for further processing. The stored document is then passed to the **Image Preprocessing Module**, where techniques such as resizing, grayscale conversion, noise removal, and thresholding are applied using **OpenCV**. These steps enhance image quality and improve the accuracy of text extraction.

Once preprocessing is completed, the enhanced image is sent to the **OCR Engine** (Tesseract or EasyOCR), which extracts the visible text from the receipt or invoice and converts it into raw OCR text. In Milestone–2, this raw OCR output is further processed by an **LLM-based Structured Extraction Module** powered by the **Groq LLM**. This module intelligently analyzes the OCR text and converts it into structured JSON data containing fields such as merchant name, date, payment method, subtotal, tax, total amount, and line-item details.

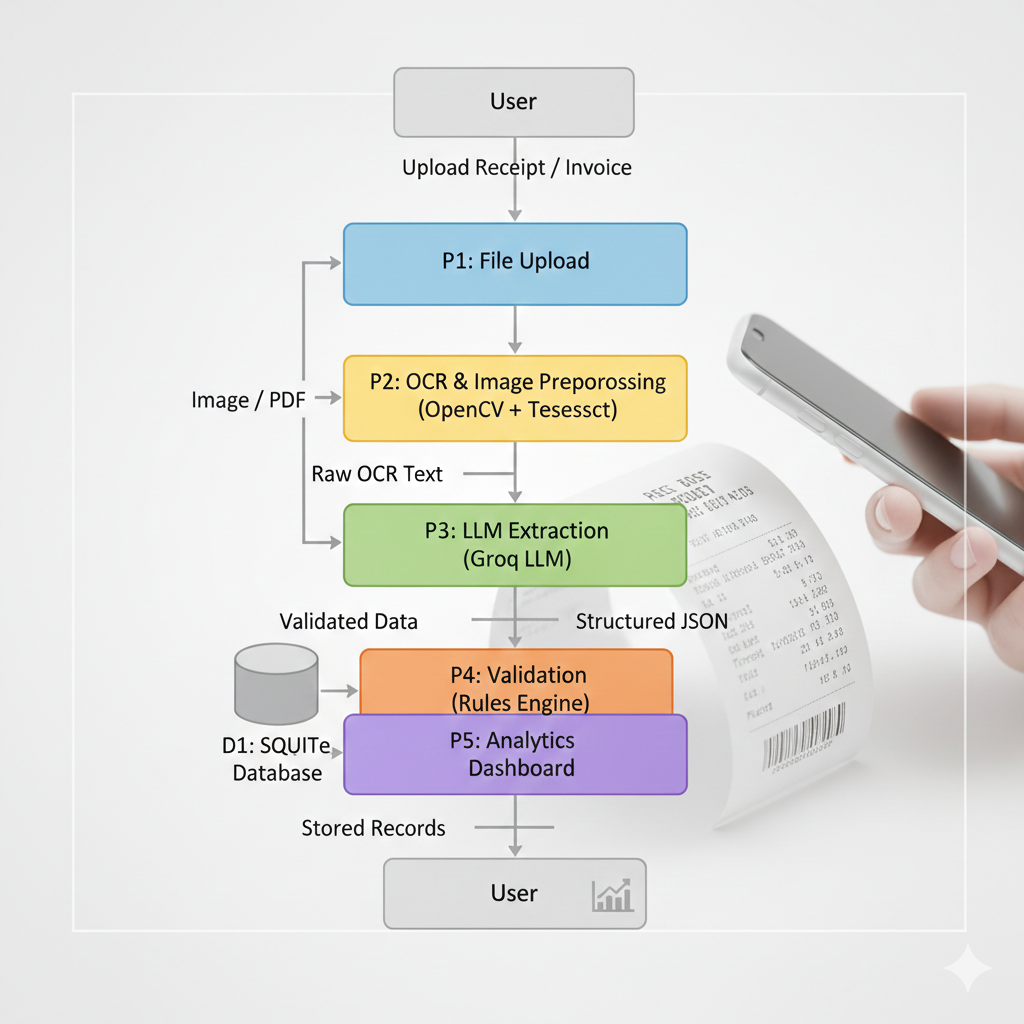
The structured data then passes through the **Validation Engine**, where multiple business-rule checks are performed. These include verifying that the subtotal and tax approximately match the total amount, detecting duplicate receipts, validating tax ranges, and checking the presence and format of essential fields such as date and merchant name. This step ensures data accuracy, consistency, and reliability.

After successful validation, the cleaned and verified receipt data is stored in a **SQLite database**, providing persistent storage and enabling receipt history management. The stored data can be retrieved later for review, filtering, deletion, and analysis. Finally, the database feeds into the **Analytics Dashboard**, which generates visual insights such as merchant-wise spending distribution and time-based expense trends using charts and graphs. These insights are displayed back to the user, transforming raw receipt images into meaningful and actionable financial information.

**DATA FLOW DIAGRAM :**

## ****External Entity: User****

The **User** is the external entity who interacts with the system. The user uploads receipt or invoice documents and later views extracted data, stored records, and analytics results generated by the system.

****

## ****Process P1: File Upload****

The data flow begins when the user uploads a receipt or invoice image/PDF into the system.  
**Process P1 (File Upload)** is responsible for accepting the input document and validating basic constraints such as file type (JPG, PNG, PDF) and file readability. Once the file is successfully validated, it is forwarded for storage and further processing. This process ensures that only valid input data enters the system.

**Data Flow:**  
User → Receipt/Invoice File → P1 (File Upload)

## ****Data Store D1: Document Storage****

After file upload, the receipt or invoice is stored temporarily in **Document Storage (D1)**. This storage acts as a buffer to hold the uploaded file so that it can be accessed by subsequent processing stages. The system retrieves the document from this storage whenever OCR or preprocessing is required.

**Data Flow:**  
P1 → Stored Document → D1  
D1 → Image/PDF → P2

## ****Process P2: OCR & Image Preprocessing****

**Process P2** performs both image enhancement and OCR operations. The stored receipt image is first preprocessed using **OpenCV**, applying techniques such as resizing, grayscale conversion, noise reduction, and thresholding to improve text clarity.  
After preprocessing, **Tesseract OCR** extracts raw text from the enhanced image. This raw OCR text represents unstructured textual data obtained directly from the receipt.

**Data Flow:**  
D1 → Image/PDF → P2  
P2 → Raw OCR Text → P3

## ****Process P3: LLM-Based Extraction****

In **Process P3**, the system uses a **Groq Large Language Model (LLM)** to convert raw OCR text into meaningful structured information. The LLM analyzes the unstructured text and extracts important fields such as merchant name, date, payment mode, subtotal, tax, total amount, and item-level details.  
The output of this process is **structured JSON data**, which is easier to validate, store, and analyze.

**Data Flow:**  
P2 → Raw OCR Text → P3  
P3 → Structured JSON → P4

## ****Process P4: Validation (Rules Engine)****

**Process P4** validates the structured data using predefined business rules. This step ensures data correctness and reliability. Validation checks include subtotal and tax consistency with total amount, duplicate receipt detection, acceptable tax range verification, date format recognition, and mandatory field validation.  
Only validated and consistent data is allowed to proceed to permanent storage.

**Data Flow:**  
P3 → Structured JSON → P4  
P4 → Validated Data → D2

## ****Data Store D2: SQLite Database****

The **SQLite Database (D2)** stores validated receipt records permanently. This data store maintains a history of all processed receipts and supports operations such as retrieval, filtering, deletion, and reuse for analytics. Persistent storage ensures that receipt data remains available even after the session ends.

**Data Flow:**  
P4 → Validated Receipt Data → D2  
D2 → Stored Records → P5

## ****Process P5: Analytics Dashboard****

**Process P5** uses stored receipt data from the database to generate analytical insights. The analytics dashboard creates visual representations such as merchant-wise spending charts, time-based expense trends, and summary statistics. These insights help users understand their spending behavior in a meaningful way.

**Data Flow:**  
D2 → Stored Records → P5  
P5 → Charts & Insights → User

## ****Final Output to User****

The user receives structured receipt information, validated records, and analytical insights through the dashboard interface. This completes the data flow cycle from raw document upload to actionable financial insights.

**TECHNOLOGY STACK :**

| **S.No** | **Technology** | **Purpose / Usage** |
| --- | --- | --- |
| **1** | **Python** | **Core backend processing and application logic** |
| **2** | **Streamlit** | **Interactive web-based user interface and dashboard** |
| **3** | **OpenCV** | **Image preprocessing and enhancement** |
| **4** | **Tesseract OCR** | **Extraction of raw text from receipt/invoice images** |
| **5** | **Groq LLM** | **Structured field extraction from OCR text** |
| **6** | **SQLite** | **Persistent storage of validated receipt data** |
| **7** | **Pandas** | **Data manipulation, validation, and filtering** |
| **8** | **Matplotlib** | **Analytics visualization and graphical insights** |

## ****OCR & IMAGE PREPROCESSING :****

The OCR and image preprocessing module plays a crucial role in converting physical or scanned receipts into machine-readable text. In this system, **OpenCV** is used to enhance the quality of receipt images before text extraction. Since receipts may contain noise, shadows, skew, or low contrast, preprocessing techniques such as resizing, grayscale conversion, noise reduction, and thresholding are applied to improve text visibility. Once the image quality is enhanced, **Tesseract OCR** is used to extract raw textual content from the preprocessed image. This combination significantly improves OCR accuracy and ensures that the extracted text is clear and reliable, forming a strong foundation for further structured data extraction.

## ****LLM-BASED STRUCTURED EXTRACTION :****

After obtaining raw OCR text, the system uses a **Large Language Model (Groq LLM)** to convert unstructured text into structured data. Receipts often vary in layout, font style, and formatting, making rule-based extraction unreliable. The Groq LLM intelligently analyzes the OCR text and extracts meaningful fields such as merchant name, transaction date, payment method, subtotal, tax amount, total amount, and line-item details. The extracted information is returned in a **structured JSON format**, which standardizes data representation and makes it easier to validate, store, and analyze. This LLM-based approach enables the system to handle diverse receipt formats efficiently.

## ****VALIDATION ENGINE :****

The validation engine ensures the correctness and consistency of extracted receipt data before it is stored permanently. The system applies multiple **business rule validations** to detect errors and inconsistencies. These include verifying whether the subtotal and tax values approximately match the total amount, detecting duplicate receipts based on merchant, date, and total, validating whether tax values fall within acceptable ranges, and checking the presence and format of mandatory fields such as date and merchant name. By filtering out invalid or inconsistent data, the validation engine improves system reliability and prevents incorrect financial records from entering the database.

## ****DATABASE STORAGE :****

Validated receipt data is stored in a **SQLite database**, which provides persistent and lightweight storage. The database maintains a structured record of all processed receipts, enabling long-term storage and retrieval. It supports essential operations such as inserting new receipt records, deleting incorrect or duplicate entries, and filtering receipts based on merchant or date. This persistent storage ensures that extracted data is not lost between sessions and allows the system to maintain a complete receipt history for future analysis and reporting.

## ****ANALYTICS DASHBOARD :****

The analytics dashboard transforms stored receipt data into meaningful insights. Using data retrieved from the SQLite database, the system generates visual analytics such as **merchant-wise spending distribution** and **time-based expense trends**. Charts like pie charts and line graphs are created using **Matplotlib** to provide a clear and intuitive representation of spending patterns. This dashboard helps users understand where and how their money is spent, turning digitized receipts into actionable financial insights rather than just stored data.

## ****OUTPUT SCREENS :****

## 

## 

## 

## ****CONCLUSION :****

The successful completion of **Milestone–1 and Milestone–2** demonstrates the gradual and systematic development of a complete **Receipt & Invoice Digitizer system**. In **Milestone–1**, the primary focus was on building a strong technical foundation for receipt digitization. This milestone effectively addressed the challenge of handling physical and scanned receipts by implementing a reliable document ingestion pipeline. Users were able to upload receipt and invoice documents in multiple formats, which were then enhanced using image preprocessing techniques such as resizing, grayscale conversion, noise reduction, and thresholding. By integrating OCR technology, the system successfully converted visual receipt data into machine-readable raw text. This milestone validated that the system could accurately extract textual information from real-world receipts, forming the essential base for further automation.

Building upon this foundation, **Milestone–2** significantly enhanced the system’s capabilities by transforming raw OCR output into meaningful, structured, and usable data. Instead of relying solely on unstructured OCR text, the system integrated an **LLM-based structured extraction approach** using Groq LLM, enabling intelligent identification of key receipt fields such as merchant name, transaction date, payment method, subtotal, tax, total amount, and item-level details. This approach proved effective in handling variations in receipt layouts, fonts, and formats that are difficult to manage using traditional rule-based methods.

Furthermore, Milestone–2 introduced a robust **validation mechanism** to ensure data accuracy and consistency. By applying business rules such as subtotal–tax–total verification, duplicate receipt detection, tax range validation, and mandatory field checks, the system significantly reduced the possibility of incorrect or inconsistent data being stored. The inclusion of a **persistent database (SQLite)** allowed validated receipt data to be stored reliably, supporting long-term record maintenance, retrieval, filtering, and deletion. This persistent storage transformed the system from a temporary OCR tool into a reusable and dependable receipt management solution.

In addition, the implementation of an **analytics dashboard** converted stored receipt data into valuable insights. By visualizing merchant-wise spending patterns and time-based expense trends, the system enabled users to better understand and analyze their financial behavior. This marked a critical transition from simple data extraction to intelligent financial analysis.

Overall, the combined implementation of Milestone–1 and Milestone–2 results in a **complete, scalable, and intelligent receipt digitization platform**. The system successfully progresses from raw document input to validated, structured data and finally to actionable insights. These milestones not only meet the project objectives but also lay a strong foundation for future enhancements such as expense categorization, multi-language support, cloud integration, and advanced financial reporting. The project demonstrates practical application of OCR, AI, and data analytics in solving real-world problems related to document digitization and financial data management.

**Milestone -3**

**Introduction**

Milestone 3 represents a crucial phase in the project lifecycle where the focus shifts from data extraction and validation to data presentation, visualization, and reporting. While previous milestones concentrated on acquiring accurate receipt data through OCR, APIs, NLP, and validation mechanisms, this milestone emphasizes making the data usable and understandable for end users.

The Dashboard & Reporting module acts as the user-facing layer of the system. It enables users to interact with the application, upload receipts, review extracted information, analyse spending behaviour, and export reports for external use. This milestone significantly enhances the practical utility of the system.

**Purpose of Milestone 3**

The primary purpose of Milestone 3 is to convert previously extracted and validated receipt data into meaningful insights through a visual, interactive, and user-friendly dashboard. While earlier milestones focused on data extraction, validation, and storage, this milestone emphasizes data utilization and presentation.

Milestone 3 bridges the gap between raw structured data and end-user understanding by enabling users to interact with their receipt data, analyse spending patterns, and generate reports. The goal is to make the system practically usable for decision-making rather than just data collection.

## Scope of Milestone 3

## Design and implementation of an improved and responsive user interface.

## Development of an interactive analytics dashboard with multiple visualizations.

## Category-wise, merchant-wise, time-based, and payment method analysis.

## Integration of multi-currency analysis features.

## Implementation of trend analysis and spending pattern detection.

## Addition of export and reporting functionality for downloading insights.

## Enhancement of overall system usability and analytical capability.

## Preparation of the system for better scalability and future feature expansion.

## Objectives of Milestone 3: Dashboard & Reporting

The primary objective of Milestone 3 is to transform the validated receipt data generated in earlier milestones into a user-friendly, interactive, and analytical system. This milestone focuses on presenting data in a meaningful way so that users can easily understand, review, and analyse their spending behaviour.

1. **To Develop an Interactive Streamlit Dashboard**

The first objective is to design and implement an interactive dashboard using Streamlit. The dashboard serves as the main interface through which users interact with the system. It enables seamless navigation between different features such as uploading receipts, reviewing stored data, viewing analytics, and exporting reports. The use of Streamlit ensures rapid development and real-time updates based on user input.

**2. To Enable Receipt Upload and Review**

Another key objective is to allow users to upload receipt files and review previously stored receipts within the dashboard. This feature ensures transparency by letting users verify extracted details such as merchant name, date, and total amount. It also improves trust in the system by allowing users to cross-check stored information.

**3. To Provide Simple yet Meaningful Analytics**

Milestone 3 aims to generate simple analytics, particularly monthly spending totals, to help users understand their financial patterns. By grouping receipts based on time periods, users can identify trends, compare expenses across months, and gain insights into their spending habits without requiring complex financial knowledge.

**4. To Visualize Data for Better Understanding**

An important objective is to convert numerical data into visual representations such as tables and charts. Visual analytics makes it easier for users to interpret spending patterns, recognize trends, and quickly grasp insights that would otherwise be difficult to understand from raw data.

**5. To Support Report Export in CSV and Excel Formats**

This milestone also focuses on enabling users to export analysed data in CSV and Excel formats. Export functionality allows users to store records externally, perform offline analysis, share reports, and maintain financial documentation for personal or academic use.

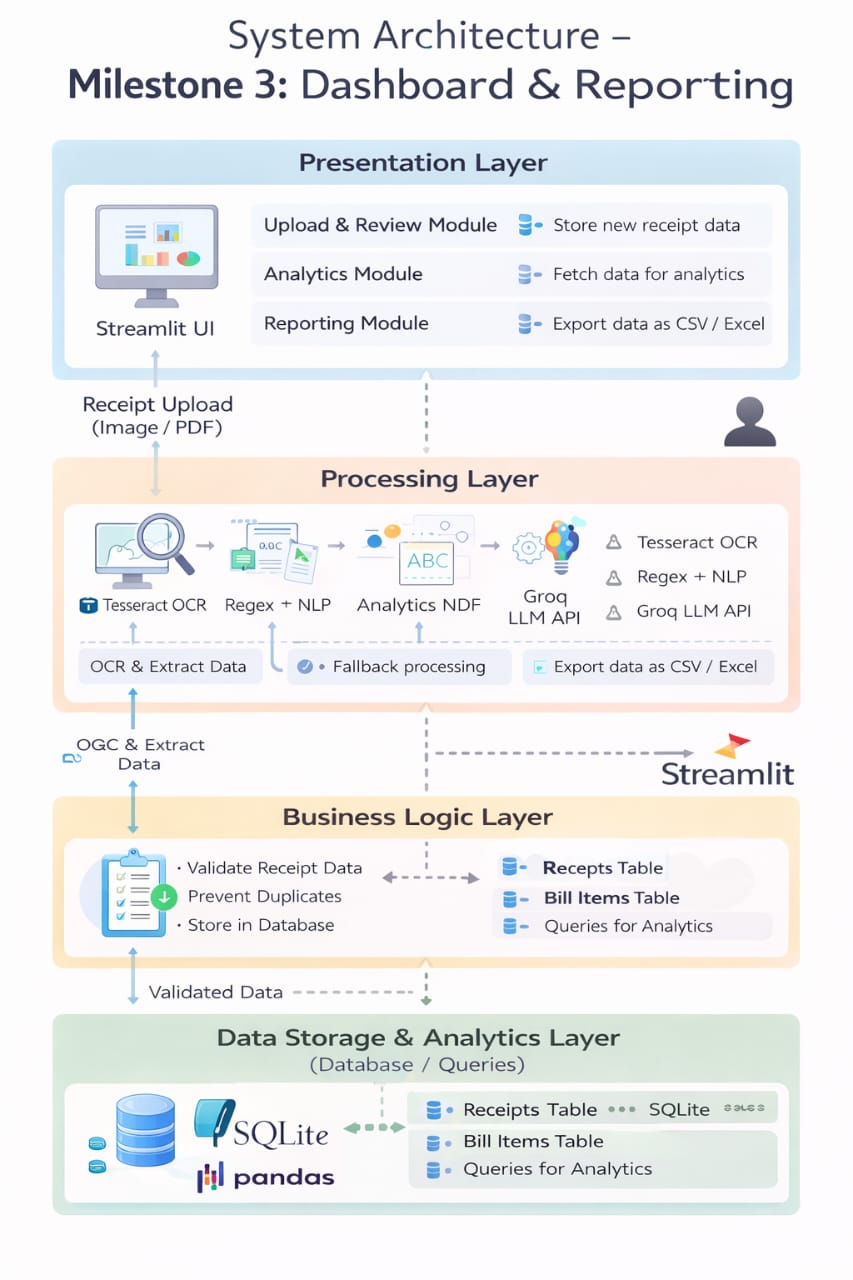
**6. To Improve Overall User Experience**

Improving usability is another objective of Milestone 3. The dashboard is designed to be intuitive, responsive, and easy to use, even for non-technical users. Clear labels, structured layouts, and interactive components enhance the overall user experience.

**7. To Prepare the System for Advanced Analytics**

Finally, Milestone 3 lays the foundation for future enhancements such as category-wise analysis, predictive analytics, and budget forecasting. By organizing data properly and implementing basic analytics, the system becomes ready for more advanced features in later milestones.

**System Architecture**

****

**System Architecture – Milestone 3: Dashboard & Reporting**

The system architecture for Milestone 3 is designed to convert processed receipt data into visual insights and downloadable reports through a Streamlit-based dashboard. The architecture follows a layered and modular approach, which improves scalability, readability, and maintenance.

**1. Presentation Layer (User Interface)**

The Presentation Layer is the topmost layer and serves as the interaction point between the user and the system.

Components:

* Streamlit UI
* Upload & Review Module
* Analytics Module
* Reporting (Export) Module

Functionality:

* Allows users to upload receipts (Image or PDF format)
* Displays extracted receipt details for review
* Shows analytics such as monthly spending summaries
* Enables users to export data as CSV or Excel files

This layer focuses entirely on user experience and visualization and does not contain heavy processing logic.

**2. Receipt Upload Flow**

When a user uploads a receipt:

* The file is sent from the Streamlit UI to the backend
* Supported formats include images and PDFs
* The uploaded receipt triggers the processing workflow
* This ensures smooth integration between the UI and backend processing.

**3. Processing Layer**

The Processing Layer is responsible for extracting and interpreting text data from uploaded receipts.

Key Components:

* Tesseract OCR: Converts receipt images/PDFs into raw text
* Regex + NLP: Identifies structured fields such as:
* Date
* Merchant name
* Total amount
* Line items
* **Groq LLM API**: Handles complex or unclear receipt formats and improves extraction accuracy
* **Fallback Processing**: Ensures robustness when OCR or rule-based extraction fails

This layer prepares clean, semi-structured data for validation.

**4. Business Logic Layer**

The Business Logic Layer acts as the control layer that ensures data quality and consistency.

Responsibilities:

* Validates extracted receipt data
* Prevents duplicate receipts from being stored
* Applies business rules (mandatory fields, correct formats)
* Organizes data into logical entities

Database Tables:

* Receipts Table: Stores receipt-level details
* Bill Items Table: Stores item-wise purchase details
* Supports optimized queries for analytics

Only validated and structured data is passed to the storage layer.

**5. Data Storage & Analytics Layer**

This layer handles persistent data storage and analytics queries.

Technologies Used:

* SQLite: Lightweight relational database
* Pandas: Used for data analysis and aggregation

Functionality:

* Stores validated receipt and item data

Executes queries for:

* Monthly totals
* Category-wise spending
* Trends for dashboard visualization

Supplies data to analytics and reporting modules

**6. Analytics & Reporting Flow**

Using the stored data:

* Pandas processes data to compute totals and summaries
* Results are visualized in Streamlit using charts and tables

Users can download:

* CSV reports
* Excel files

This completes the data-to-insight pipeline.

**Dashboard Features – Detailed Explanation (Milestone 3)**

The dashboard is the core component of Milestone 3. It transforms structured receipt data stored in the database into visual, interactive, and actionable insights. The dashboard is developed using Streamlit, which enables rapid UI development and real-time data interaction.

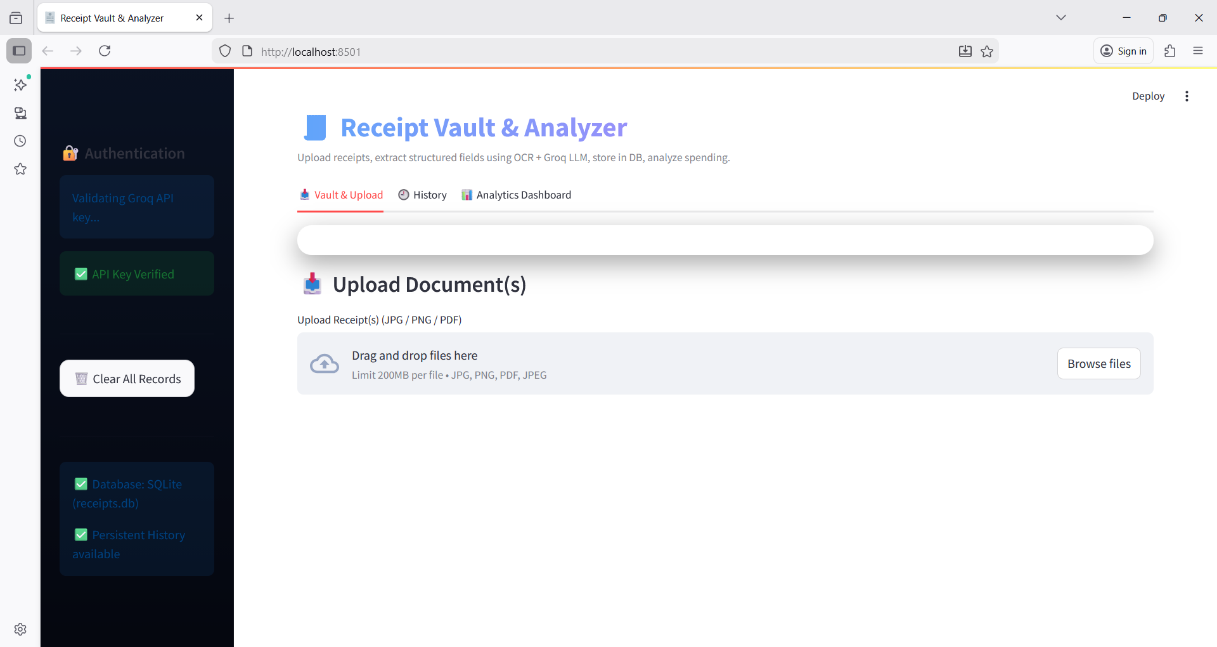
**1. User-Friendly Dashboard Interface**

**Description:**

The dashboard provides a clean and intuitive layout where all analytical components are organized into sections such as filters, metrics, charts, and reports.

**Importance:**

A simple interface ensures that users without technical knowledge can easily navigate the system and understand their spending behaviour.



**2. Receipt Upload and Review Panel**

**Description:**

Users can upload receipt images or PDF files. Extracted data such as merchant name, date, subtotal, tax, and total amount are displayed for review.

**Importance:**

This feature ensures data transparency and accuracy, allowing users to confirm the correctness of extracted values before analysis.

**3. Interactive Filtering Mechanism**

**Description:**

* The dashboard supports dynamic filters including:
* Date range (monthly or custom period)
* Merchant selection
* Category-based filtering

**Importance:**

Filters allow users to focus on specific datasets, making analysis more meaningful and personalized.

**4. Key Performance Metrics Display**

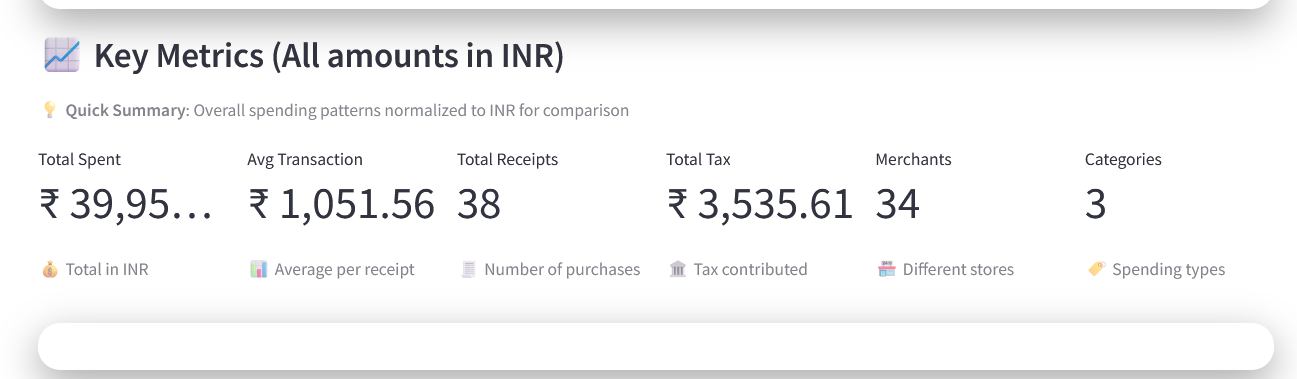
**Description:**

Important financial indicators are shown at the top of the dashboard:

* Total spending
* Average spending per receipt
* Number of receipts
* Total tax
* Categories

**Importance:**

These metrics provide a quick summary of financial activity without needing to interpret charts.



**5.Treemap Chart – Spending by Category**

**Description:**

The treemap visually represents total spending across different categories using rectangular blocks. The size of each block corresponds to the amount spent in that category, allowing easy comparison of expenses at a glance.

**Purpose:**

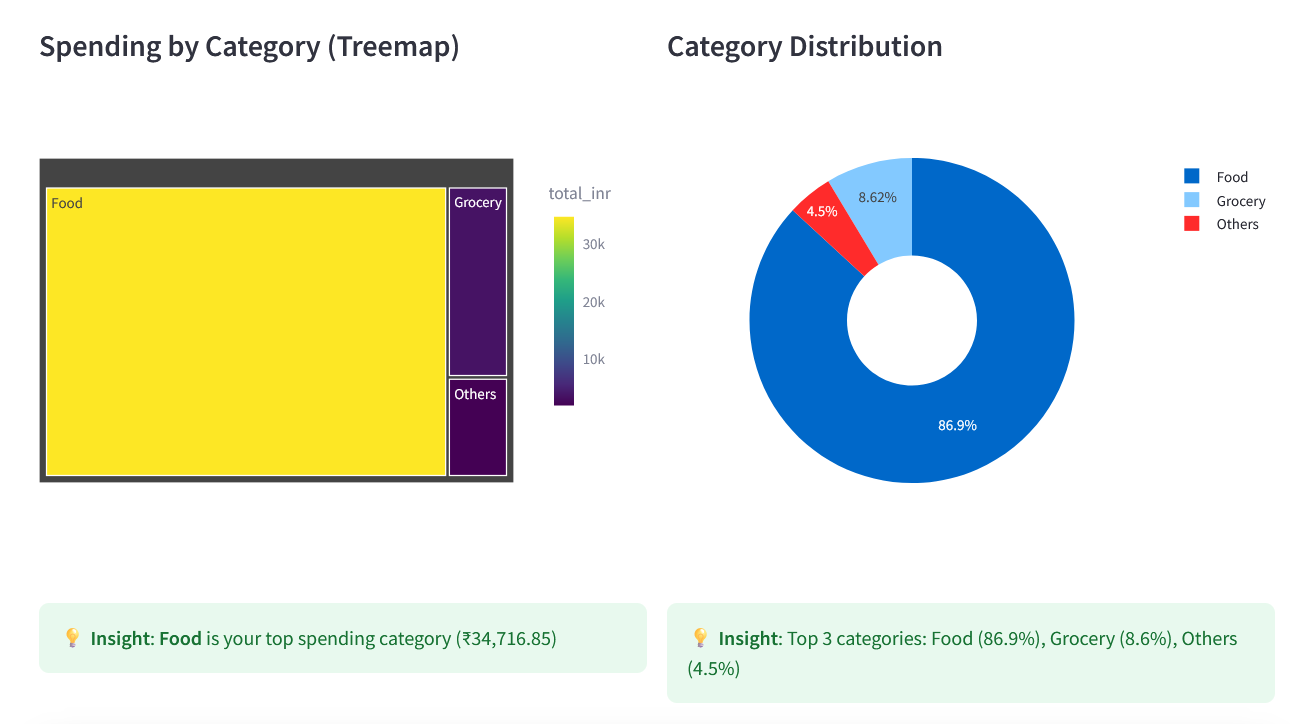
* To identify the highest spending category.
* To compare expenses across categories.
* To understand spending concentration.
* To support budgeting decisions.

6. **Donut Chart (Category Distribution)**

**Description:**

The donut chart displays the percentage distribution of total expenses among different categories. Each segment represents a category, and its size indicates its share of overall spending.

**Purpose:**

* To show proportional expense distribution.
* To identify dominant and minor categories.
* To analyse spending balance.
* To support financial planning.

### 7.Top Merchants by Spending (Bar Chart)

**Description:**  
Shows the top merchants where the user spends the most money.

**Purpose:**

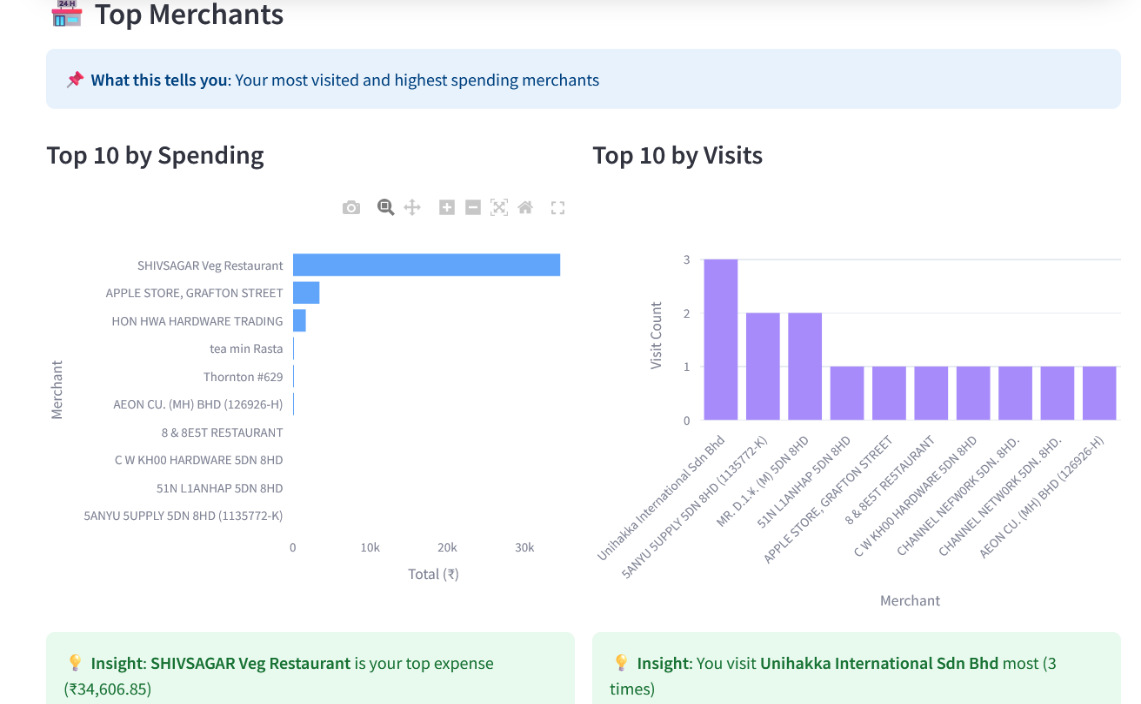
* Identifies major expense sources.
* Highlights frequent or costly stores.
* Helps control unnecessary spending.

8. **Top 10 by Visits (Bar Chart)**

**Description:**

This chart shows the top 10 merchants based on visit frequency. The height or length of each bar represents how often transactions were made with that merchant.

**Purpose:**

* To identify frequently visited merchants.
* To analyse transaction frequency patterns.
* To understand purchasing habits.
* To compare visit frequency with spending amount**.**

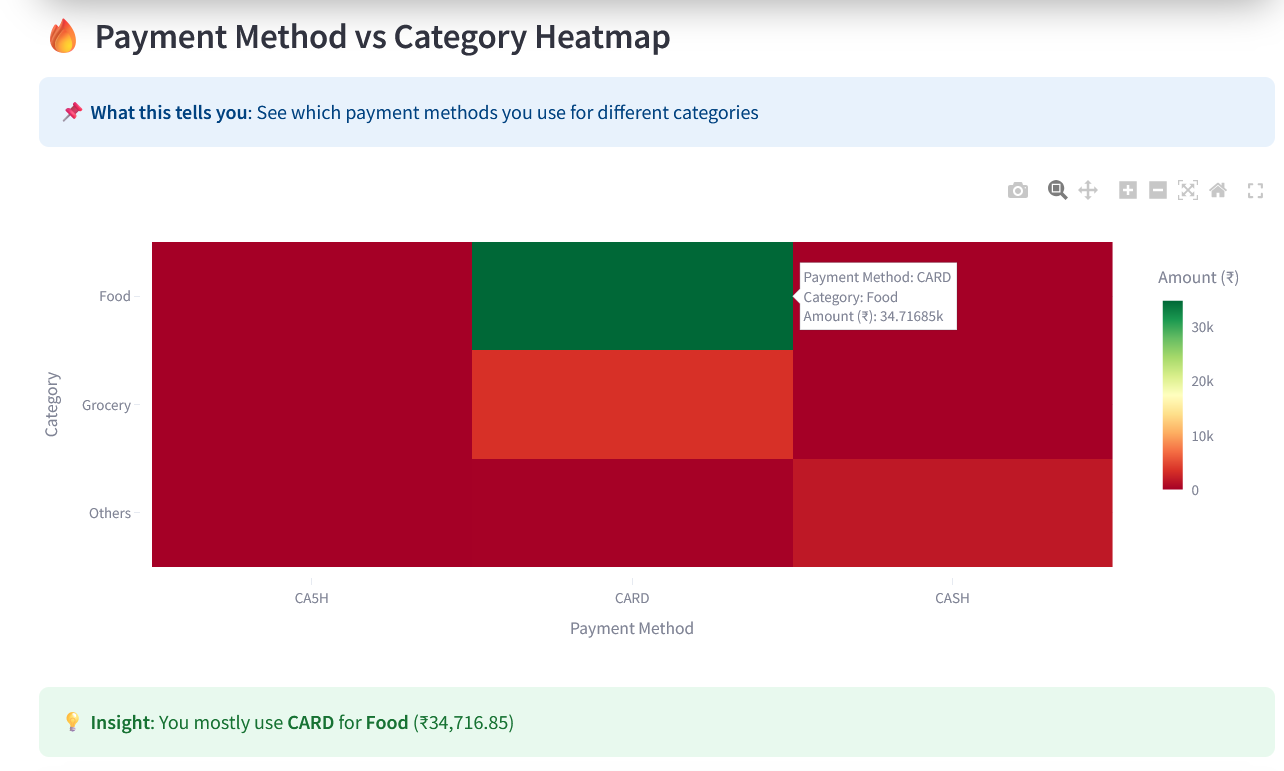
**9. Payment Method vs Category (Heatmap)**

**Description:**

The heatmap illustrates the relationship between payment methods and spending categories. Colour intensity represents the level of spending for each combination.

**Purpose:**

* To analyse payment method preferences.
* To compare spending across payment types.
* To identify dominant payment patterns.
* To understand transaction behaviour**.**



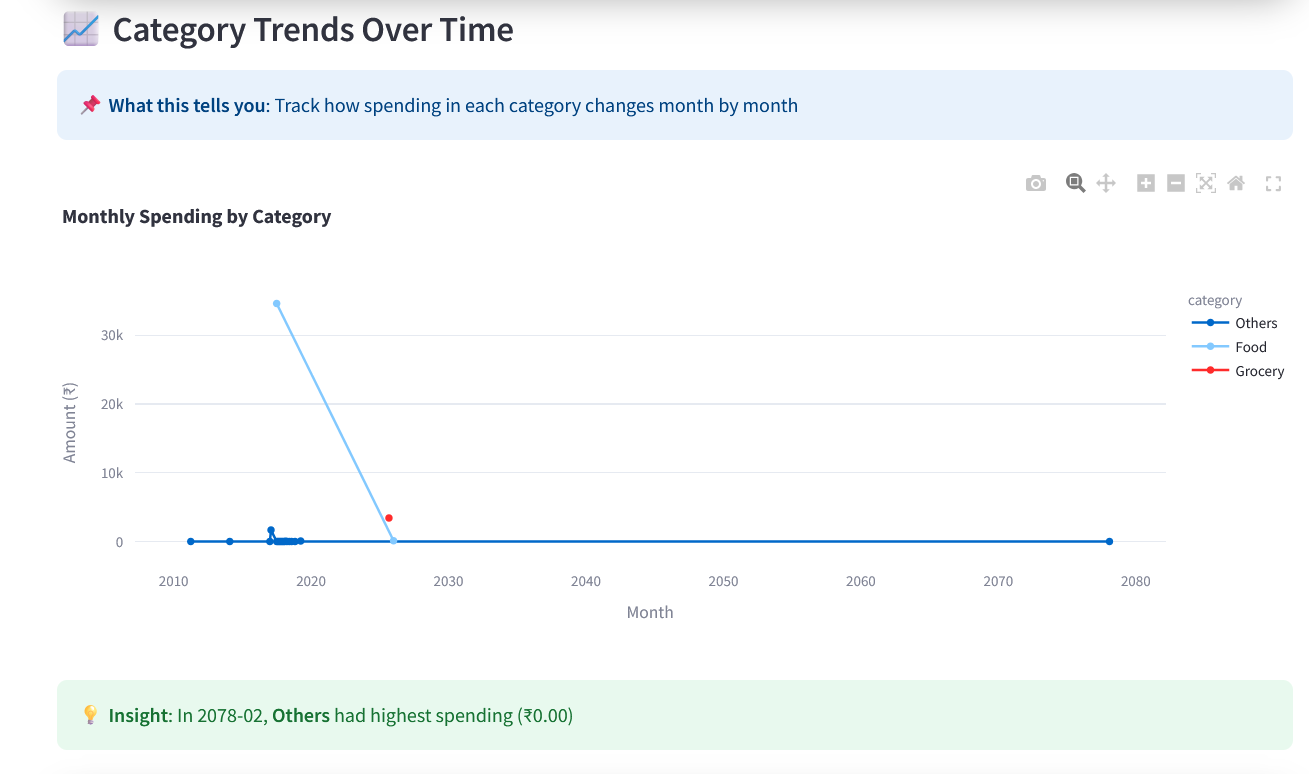
**10. Category Trends Over Time (Line Chart)**

**Description:**

The line chart shows how spending in different categories changes over time. Each line represents a category, and the movement of the line indicates increases or decreases in expenses across months.

**Purpose:**

* To track monthly spending changes.
* To detect trends and seasonal patterns.
* To identify unusual spikes or drops.
* To support long-term financial analysis.

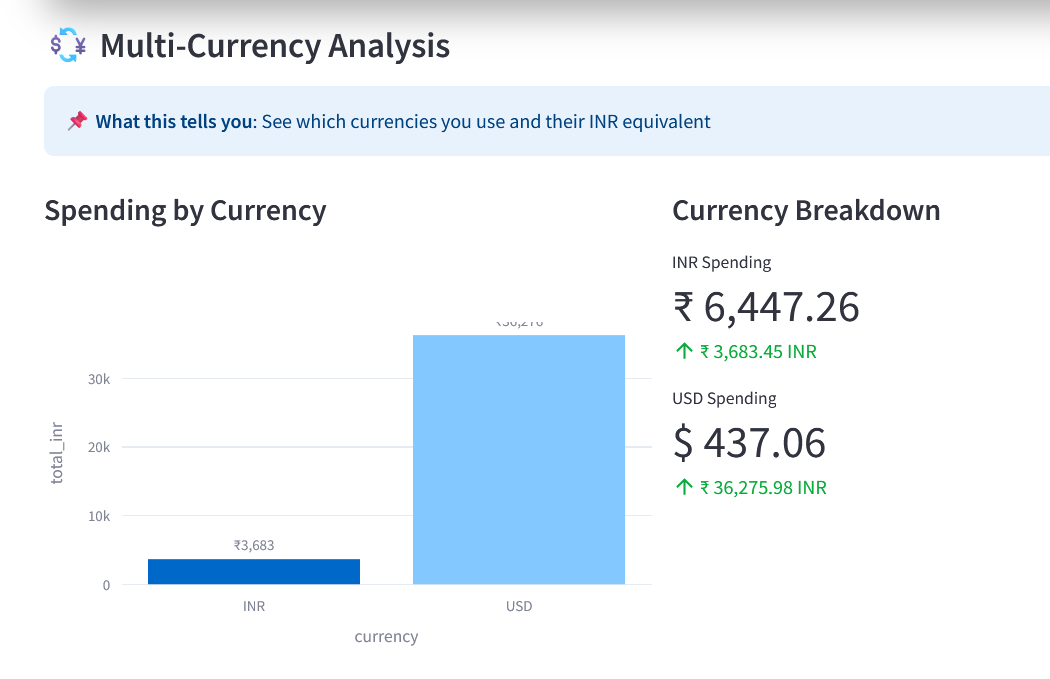


**11.Spending by Currency (Bar Chart)**

**Description:**

This section compares spending across different currencies and provides a summary of currency-wise expenses. It highlights how spending varies based on currency usage.

**Purpose:**

* To compare expenses across currencies.
* To analyse foreign currency usage.
* To evaluate exchange impact on spending.
* To support international transaction analysis.

**12. Currency Breakdown (Summary Panel)**

* This section provides a numerical summary of spending in each currency.
* It shows the total amount spent in each currency separately.
* It also displays the equivalent value in INR for comparison purposes.
* The upward indicator represents the converted or total equivalent value.
* This breakdown helps in understanding the impact of foreign currency spending when converted into INR.
* It is useful for tracking cross-border transactions and managing exchange rate implications.

### 13.Monthly Spending Trend (Line Chart)

**Description:**  
Shows total spending for each month.

**Purpose:**

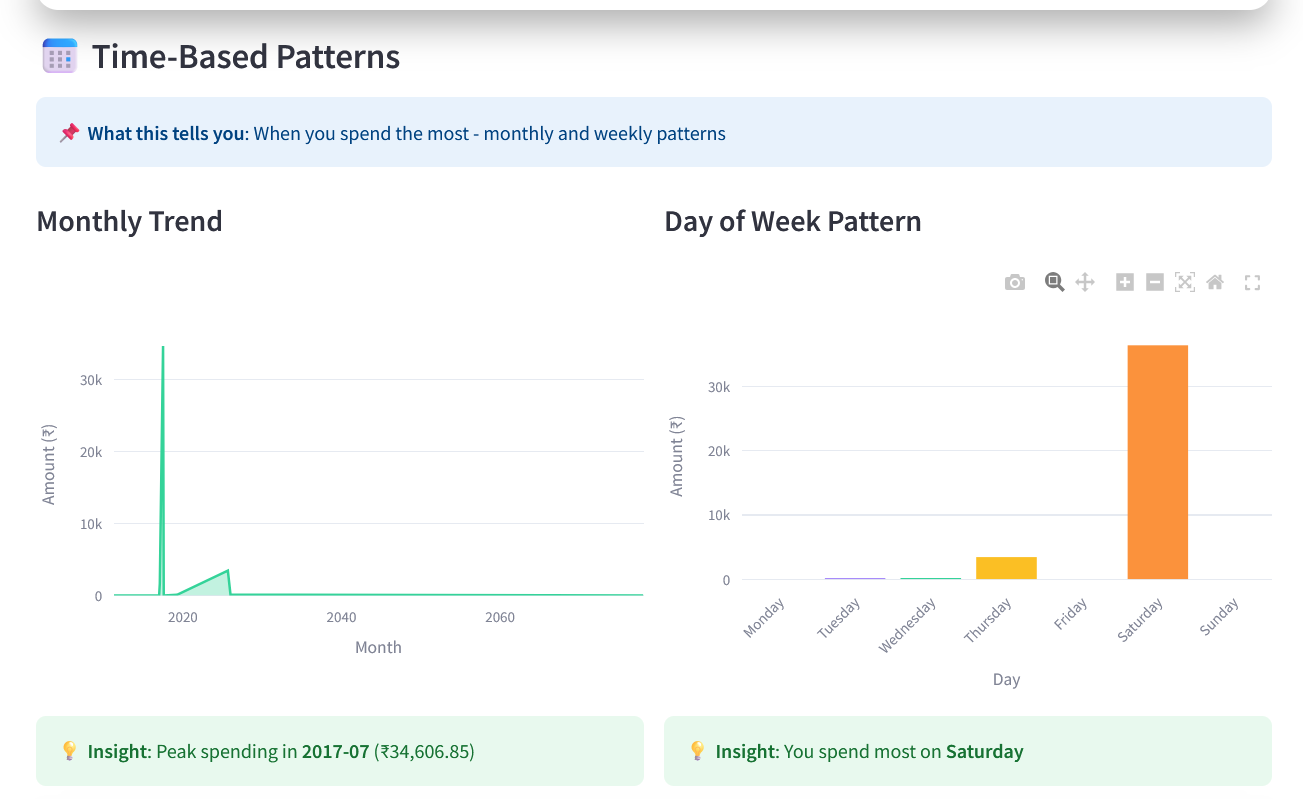
* Tracks spending changes over time.
* Detects increasing or decreasing trends.
* Helps in monthly budget planning.

### 14. Spending by Day of the Week (Bar Chart)

**Description:**  
Displays spending distribution across days (Monday–Sunday).

**Purpose:**

* Identifies peak spending days.
* Helps users adjust shopping habits.



## Export Functionality

The system includes a data export feature to allow users to download their receipt records.

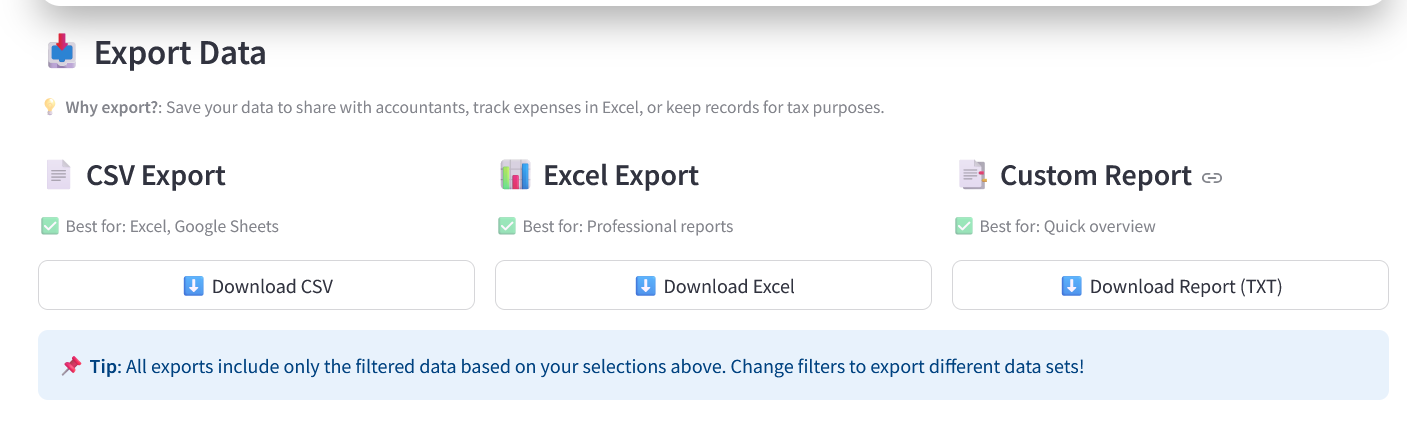
### Supported Formats

* **CSV (.csv)**
* **Excel (.xlsx)**

### Purpose of Export

* Store records offline.
* Perform advanced analysis in tools like Excel.
* Share reports for academic or financial purposes.
* Maintain backups of transaction data.

This feature enhances the practical usability of the application beyond the dashboard.



**16. Real-Time Data Updates**

**Description:** The dashboard automatically updates visualizations when new receipts are uploaded.

**Importance:** Ensures the analysis always reflects the most recent data.

**17. Secure Database Interaction**

**Description:** Each dashboard module fetches data directly from the backend database using isolated sessions.

**Importance:** Improves data integrity, performance, and system scalability.

**Overall Benefit of the Dashboard**

The dashboard converts raw receipt data into clear insights, enabling users to:

* Track spending habits
* Identify trends and anomalies
* Make informed financial decisions

## Overall, Purpose of the Analytics Module

The analytics component is designed to:

* Provide clear visual insights into spending behaviour.
* Help users identify trends and patterns.
* Support better financial decision-making.
* Make receipt data easy to understand through charts and metrics.

**Conclusion**

This milestone successfully enhanced the system by improving the user interface, strengthening dashboard analytics, and implementing export functionality. The user interface was designed to be clean, interactive, and easy to navigate, ensuring a smooth user experience.

The dashboard effectively converts raw transaction data into meaningful visual insights through various analytical charts, helping users understand spending patterns, merchant behaviour, time-based trends, and payment preferences.

Additionally, the export feature enables users to download reports and summaries, making the system more practical for documentation and sharing purposes.

Overall, this milestone improves usability, analytical depth, and reporting capability, making the application more efficient, user-friendly, and decision-oriented.