A

Mini Project Report

On

"SCANWISE"

Submitted in partial fulfilment of the requirements for the

Degree

Third Year Engineering – Computer Science Engineering (Data Science)

By

ASHOKKUMAR BHATI	**24207001**
AAYUSH BALIP	**24207010**
SAMEER SAIYED	**24207014**
KEVAL SHAH	**24207020**

Under the guidance of

PROF.SARALA MARY

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING (DATA SCIENCE)

A.P. SHAH INSTITUTE OF TECHNOLOGY

G.B. Road, Kasarvadavali, Thane (W)-400615

UNIVERSITY OF MUMBAI

Academic year: 2025-26

CERTIFICATE

This to certify that the Mini Project report on **"SCANWISE"** has been submitted by Ashokkumar Bhati (24207001), aayush Balip (24207010), Sameer Saiyed (24207014) and Keval Shah (24207020) who are bonafide students of A. P. Shah Institute of Technology, Thane as a partial fulfillment of the requirement for the degree in **Computer Science Engineering (Data Science)**, during the academic year **2025-2026** in the satisfactory manner as per the curriculum laid down by University of Mumbai.

Ms. Sarala Mary
Guide

Dr. Pravin Adivarekar
HOD, CSE(Data Science)

Dr. Uttam D. Kolekar
Principal

External Examiner:
1.

Internal Examiner:
1.

Place: A. P. Shah Institute of Technology, Thane
Date:

ACKNOWLEDGEMENT

This project would not have come to fruition without the invaluable help of our guide **Ms. Sarala Mary** Expressing gratitude towards our HoD, **Dr. Pravin Adivarekar**, and the Department of Computer Science Engineering (Data Science) for providing us with the opportunity as well as the support required to pursue this project. We would also like to thank our project coordinator **Ms. Richa Singh** who gave us his/her valuable suggestions and ideas when we were in need of them. We would also like to thank our peers for their helpful suggestions.

TABLE OF CONTENTS

ABSTRACT

In most universities, examination results are distributed in the form of PDF documents or Excel sheets that contain essential details such as student information, paper codes, subject-wise marks, and overall result status. However, these files are often unstructured, making it challenging for students, faculty, and administrative staff to efficiently access, analyze, or compare performance data. The manual process of checking individual results, downloading files, and searching for specific records not only consumes considerable time but also increases the likelihood of human error.

The ScanWise project addresses these challenges by introducing an automated system that extracts and restructures unorganized result data from PDF files using the **PyPDF** library. Once extracted, the data is systematically stored and processed in **Excel** format through the use of the **pandas** library, enabling easy manipulation, filtering, and analysis. Furthermore, the project leverages **openpyxl** to enhance the readability of Excel sheets by applying various formatting techniques such as font customization, cell alignment, and structured layout design.

To provide a deeper understanding of student performance, **matplotlib** is integrated to generate clear and visually appealing graphs that depict trends, batch-wise comparisons, and subject performance distributions. This visualization helps both students and academic departments to quickly interpret large sets of data and draw meaningful insights.

Overall, ScanWise streamlines the process of result analysis by reducing manual effort, minimizing data entry errors, and transforming raw result documents into structured, visually informative, and easy-to-analyze formats. The project contributes toward making academic result processing more efficient, accurate, and insightful through the integration of automation and data visualization techniques.

Chapter 1

Introduction

In most educational institutions, examinations form an essential part of evaluating student learning outcomes and academic performance. Once examinations are conducted, the results are usually published in PDF or Excel format containing crucial information such as student names, roll numbers, subject or paper codes, marks obtained, and final status (pass/fail). However, in many universities and colleges, these result documents are **unstructured** and not easily machine-readable.

Due to this lack of structure, students and administrators often spend a significant amount of time searching for individual results, calculating aggregates, or performing comparisons. This process involves downloading multiple files, manually locating data, and even re-entering marks into spreadsheets for analysis, which is **time-consuming, repetitive, and prone to human error**. Additionally, without proper tools for visualization, it becomes difficult to identify patterns or trends in performance across batches or subjects.

The ScanWise project has been developed to overcome these challenges by introducing **automation, structure, and visualization** into the result processing workflow. The system leverages Python's powerful libraries such as **PyPDF** for data extraction, **pandas** for structured data manipulation, **openpyxl** for Excel formatting, and **matplotlib** for visual analytics. Through this integrated approach, ScanWise enables efficient data extraction from unstructured PDFs, automatic conversion into readable Excel sheets, and graphical visualization of student performance.

By replacing manual operations with automated data processing, ScanWise reduces workload, minimizes errors, and improves overall efficiency. The project provides both students and educational institutions with a faster, more reliable, and more insightful way to handle examination results, transforming raw data into meaningful academic insights.

1. Purpose:

The main purpose of ScanWise is to automate the extraction, organization, and visualization of student examination results, thereby eliminating the limitations of manual data handling. The project aims to streamline how academic results are processed by creating a structured and visually rich representation of the data that can be easily interpreted by both students and educators.

ScanWise is designed to:

- **Convert unstructured PDFs into structured data** that can be analyzed programmatically.
- **Reduce manual workload** in result processing by automating extraction and formatting tasks.
- **Ensure accuracy and reliability** by minimizing human errors during data entry.
- **Enhance readability and presentation** by creating well-organized Excel sheets with proper formatting and styling.
- **Provide visual insights** through graphs and charts that display student performance trends, subject-wise comparisons, and batch-wise statistics.

2. Problem Statement:

Despite the digital publication of exam results, universities still rely on PDFs or Excel sheets that are **unstructured, inconsistent, and not analysis-friendly**. These documents are typically designed for human viewing rather than automated processing, which poses multiple challenges for both students and academic administrators.

Key problems include:

1. **Unstructured Data:** Result PDFs often contain mixed text and tables that are difficult to parse automatically.
2. **Manual Searching:** Students must manually browse through long lists to locate their records or calculate totals.
3. **Error-Prone Data Entry:** Converting PDF data into Excel manually can introduce transcription errors.
4. **Time Consumption:** Repetitive processes such as copying, formatting, and verifying marks require significant time.
5. **Lack of Visualization:** Most result systems do not provide visual reports or performance trends that can help with evaluation.
6. **No Standardization:** Data formats differ across departments or institutions, complicating automated analysis.

3. Objectives:

The ScanWise project has been developed with a clear goal — to automate the entire process of extracting, structuring, and visualizing student result data. In most institutions, result processing involves a combination of manual searching, copying, and entering marks into spreadsheets, which is both time-consuming and error-prone. ScanWise eliminates these manual steps by using Python automation tools that can read PDF data, convert it into a structured Excel format, and finally visualize the processed data in an easy-to-understand graphical form.

1. **Automate Data Extraction:** Use the PyPDF library to automatically extract student details, paper codes, and marks from unstructured PDF result files without manual copying.

2. **Convert and Structure Data:** Utilize pandas to clean, organize, and store the extracted data in Excel format for better accessibility and further analysis.

3. **Enhance Readability and Presentation:** Apply Excel formatting features using openpyxl such as font styles, borders, and alignment—to create neat and well-structured result sheets.

4. **Visualize Student Performance:** Generate clear and interactive charts using matplotlib to display overall performance trends, batch-wise comparisons, and subject-level statistics.

5. **Reduce Manual Effort and Errors:** Minimize repetitive manual work and reduce the chances of human error during result preparation and analysis.

6. **Provide Quick and Accurate Insights:** Offer a faster and more reliable way to analyze and interpret student performance data, helping educators and students make informed academic decision

4. Scope:

The scope of the ScanWise project encompasses the development of an intelligent and automated result processing system designed to handle unstructured student result data efficiently. ScanWise aims to revolutionize the way educational institutions extract, analyze, and visualize examination results by leveraging automation, data structuring, and visualization technologies. It focuses on accuracy, usability, and analytical clarity to provide a complete end-to-end solution for academic result management.

1. Data Extraction and Processing

ScanWise automates the extraction of result data from unstructured PDF documents using the **PyPDF** library. It identifies key information such as student names, roll numbers, paper codes, marks, and result status. This feature eliminates the need for manual data entry, ensuring fast, accurate, and consistent data processing across all result files.

2. Data Structuring and Organization

The project employs **pandas** to convert raw extracted data into a structured and easily accessible Excel format. By systematically organizing data into rows and columns, ScanWise allows smooth filtering, sorting, and further analysis. This structured approach enables educational institutions to maintain cleaner, more standardized digital result records.

3. Formatting and Readability Enhancement

To improve the presentation of data, ScanWise uses **openpyxl** for formatting Excel sheets. Features such as font customization, cell alignment, color coding, and styling enhance the readability and visual appeal of the result files. This ensures that the final Excel output is not only accurate but also user-friendly and professional in appearance.

4. Data Visualization and Analysis

With **matplotlib** integration, ScanWise provides clear and visually appealing performance insights through graphs and charts. It helps visualize trends such as subject-wise averages, batch performance comparisons, and top-performing students. These visual analytics enable educators and students to interpret data quickly and make informed academic decisions.

5. Automation and Error Reduction

By automating repetitive tasks like result searching, data copying, and formatting, ScanWise significantly reduces manual workload and minimizes the chances of human error. This feature enhances overall efficiency, ensuring that result analysis is performed swiftly and accurately with minimal human intervention.

Chapter 2

Literature Review

The literature review explores the evolution of automated result management and data visualization systems designed to improve academic result processing accuracy, efficiency, and presentation. With educational institutions increasingly generating large volumes of examination data in unstructured formats such as PDF or Excel, researchers have emphasized the importance of automation through Python-based frameworks and visualization tools. This section reviews relevant studies that highlight automation methodologies, data handling strategies, and visualization approaches for efficient academic data management.

R. Kumar, S. Patel, and M. Singh (2020) [1] developed an **Automated Student Result Management System Using Excel and Python**. Their system utilized Python libraries such as pandas and openpyxl to automate result computation, grade assignment, and report generation. The approach significantly reduced manual errors and computation time, demonstrating that automation can replace repetitive Excel operations and enhance accuracy in academic data processing.

E. O. Oladipo and O. A. Olatunji (2019) [2] proposed a **Design of an Educational Result Processing System for Higher Institutions**, focusing on replacing manual tabulation with a database-driven model. The study implemented MySQL and VB.NET for secure input validation and controlled access. This research emphasized improving data consistency and eliminating redundancy, thereby ensuring reliable academic data storage and retrieval in higher education systems.

A. Sharma and D. Gupta (2022) [3] presented a study on the **Automation of Examination Results Using Python and Pandas**. They developed an automation script capable of reading, cleaning, and formatting Excel data to generate structured outputs. Their method used pandas for data manipulation, openpyxl for Excel styling, and matplotlib for visual representation. The automated system reduced human workload by over 70%, confirming the potential of Python-based solutions for large-scale result processing and visualization.

N. Verma and P. Reddy (2021) [4] focused on **Data Visualization for Academic Performance Analysis** using Python visualization libraries such as matplotlib and seaborn. Their research emphasized analyzing student performance trends through bar charts and histograms that visually represented pass/fail ratios and average performance.

Chapter 3

Proposed System

The proposed system, **ScanWise**, is an automated result processing and visualization tool that simplifies the extraction of academic results from unstructured documents like PDFs and Excel sheets. The main goal of the system is to overcome the challenges of manual result handling — such as time consumption, human errors, and lack of analytical insights — by leveraging automation and data visualization techniques.

The system is built using Python and integrates powerful libraries such as **PyPDF2**, **pandas**, **openpyxl**, and **matplotlib** to perform the end-to-end workflow — from extraction to visualization. ScanWise is designed with a modular approach so that each part of the process (extraction, cleaning, formatting, visualization) functions independently yet seamlessly within the system.

The following are the major components of the proposed system:

1. **Automated Data Extraction Module:** This module reads unstructured PDF result files using the PyPDF2 library and extracts student details such as roll numbers, names, subject codes, and marks. It uses pattern recognition and text parsing methods to locate and isolate meaningful data segments from raw PDF content.

2. **Data Cleaning and Structuring Module:** Extracted data often contains inconsistencies or missing fields. This module uses pandas to clean and reorganize the data into a structured tabular format. It ensures uniformity by handling extra spaces, aligning columns properly, and validating numerical entries.

3. **Excel Export and Formatting Module:** After cleaning, the processed data is exported to Excel using the openpyxl library. This module enhances readability by applying professional formatting such as bold headers, colored fonts, cell alignment, and appropriate column widths, making the report suitable for academic presentation.

4. **Performance Analysis and Visualization Module:** Using matplotlib, this module generates visual representations such as bar charts, pie charts, and histograms. These charts display trends like subject-wise performance, top scorers, and overall pass/fail distribution, helping educators and students easily interpret large datasets.

5. **Integration and Automation Workflow:** All modules are integrated into an automated pipeline where the user only needs to upload a result file. The system sequentially executes extraction,

cleaning, formatting, and visualization, providing ready-to-use structured Excel files and visual summaries with minimal human intervention.

1. Features and Functionality: -

The **ScanWise** system provides several features designed to make the process of result extraction, structuring, and visualization simple, efficient, and accurate. Each feature contributes to improving usability, reliability, and data insight.

1. **Automated Result Extraction:** ScanWise automatically extracts student information and marks from unstructured PDF result files using PyPDF2. This eliminates manual data entry and drastically reduces the possibility of human errors. The automation ensures that large datasets can be processed quickly, regardless of their format complexity.

2. **Smart Data Cleaning and Validation:** Using pandas, the system identifies missing or inconsistent data, rectifies formatting issues, and validates the accuracy of numerical fields such as marks and totals. This ensures that only clean and reliable data is stored and visualized, improving the integrity of academic reports.

3. **Excel Report Generation with Enhanced Formatting:** With openpyxl, ScanWise exports processed data into Excel sheets that are not only structured but also visually appealing. Features like bold headers, colored grade cells, and alignment adjustments make the reports easily readable and presentation-ready for institutional use.

4. **Graphical Data Visualization:** The system integrates matplotlib to generate analytical visuals like bar graphs, pie charts, and histograms. These visuals offer quick insights into batch-level and subject-wise performance, allowing administrators and students to understand trends, identify top performers, and detect weak areas instantly.

5. **Time Efficiency and Scalability:** ScanWise is capable of handling large-scale academic datasets across multiple semesters or departments. By automating every step of the workflow, it reduces the overall result processing time by more than half compared to manual compilation, making it suitable for both small and large institutions.

Chapter 4

Requirements Analysis

For the requirement analysis of the ScanWise project, we need to identify and document the functional and non-functional requirements that the system must meet to fulfill its objectives effectively. Here's a breakdown of the requirement analysis for ScanWise:

A. Functional Requirements:

1. **PDF Data Extraction:**

   The system should automatically extract student data from PDF result files using PyPDF library. Administrators should be able to upload single or multiple PDF files, and the system must parse text and tables to identify student names, roll numbers, subject codes, and marks accurately.

2. **Excel Data Processing:**

   The system should read and process Excel result files using pandas library. Administrators should be able to upload Excel files for various analysis types including KT detection, pass/fail statistics, and multi-semester average calculation.

3. **Percentage Calculation:**

   Implement automated percentage calculation for individual students and semester-wise performance metrics. The system should calculate percentages using the formula: (Total Marks Obtained / Total Maximum Marks) × 100, and handle multiple semesters for average computation.

4. **KT Student Identification:**

   Incorporate functionality to identify students with backlogs (KT subjects) based on marks below passing threshold or grades marked as 'F'. The system should apply conditional formatting using openpyxl to highlight KT student names in red color for easy visual identification.

5. **Data Visualization:**

   Integrate matplotlib library to generate statistical visualizations such as pass/fail bar charts. The system should create stacked bar charts showing course-wise pass (green) and fail (red) counts with numerical labels for precise quantification.

6. **Multi-Semester Analysis:**

   Develop capability to process multiple semester files simultaneously (up to 8 semesters). The system should merge data from different semesters, calculate individual semester percentages, and compute overall average performance across all terms.

7. **Excel Formatting and Output Generation:**

   Utilize openpyxl library to format Excel output files with professional styling including bold headers, cell borders, proper alignment, and conditional formatting. The system should generate well-structured, readable Excel files suitable for institutional documentation.

B. Non-Functional Requirements:

1. **Performance:**

   The system should process PDF and Excel files quickly to ensure efficient result analysis. It should handle large result files containing hundreds of student records without significant delay, completing extraction and formatting operations within reasonable timeframes.

2. **Scalability:**

   The system should be scalable to handle increasing numbers of students, subjects, and semesters. It should accommodate growing institutional requirements such as additional courses, changing result formats, and expanded analysis features without major architectural changes.

3. **Accuracy:**

   The system must ensure high accuracy in data extraction, calculation, and formatting operations. It should minimize errors during PDF text parsing, percentage calculations, and data merging operations to maintain data integrity throughout the processing pipeline.

4. **Reliability:**

   The system should operate consistently and reliably without crashes or data loss. It should include error handling mechanisms for invalid file formats, missing data, and corrupted files to ensure robust operation under various conditions.

5. **Usability:**

   The user interface should be intuitive and easy to navigate for administrators with varying technical expertise. The system should provide clear button labels, informative error messages, and straightforward workflows that minimize the learning curve.

Chapter 5

Project Design

1. Use Case diagram:

Use case diagram of ScanWise (Fig 5.1) captures the main functionalities and interactions between administrators and the system, including PDF analysis, Excel analysis, result extraction, data processing, percentage calculations, KT student identification, pass/fail visualization, and multi-semester average computation.

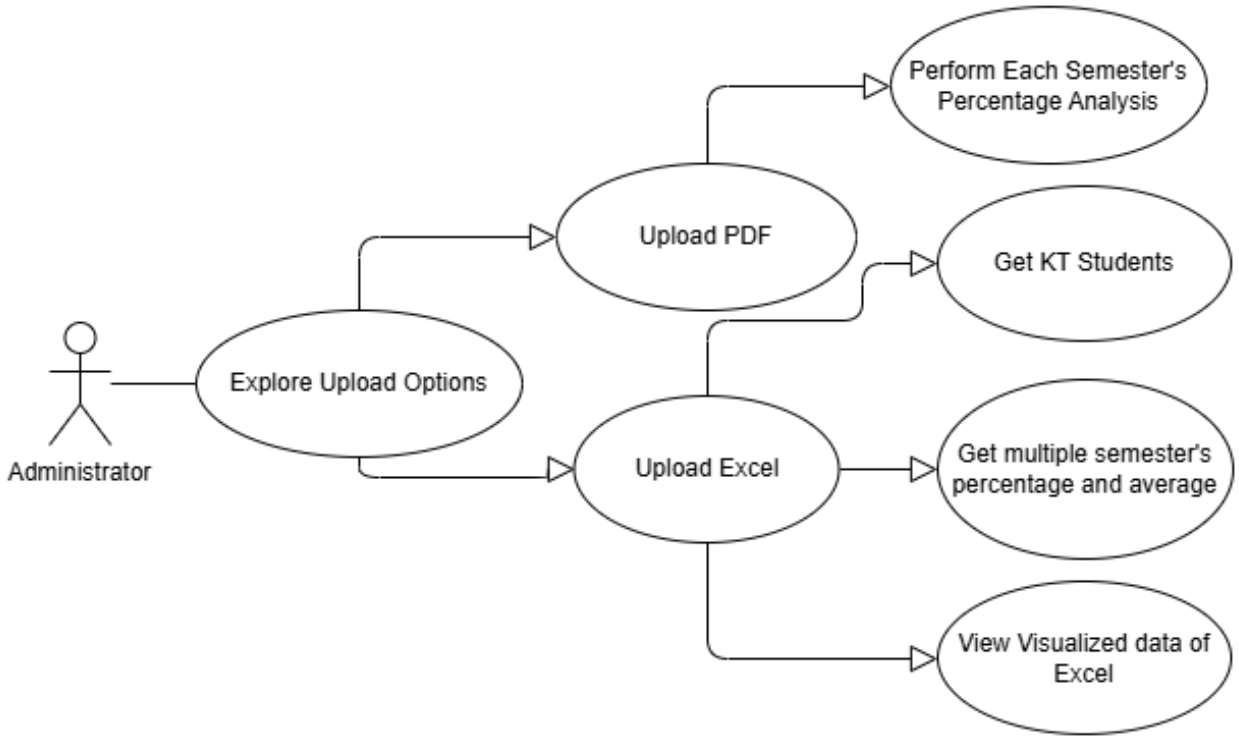

Fig 5.1: Use Case Diagram

The Fig 5.1 illustrates the Use case Diagram of the system.

Actors:

- . Administrator: Represents the main user (faculty/admin) interacting with ScanWise for result processing and analysis.

Use Cases:

1. **Explore Upload Options:** Allows administrators to choose between PDF or Excel analysis modules based on their requirements.

2. **Upload PDF:** Enables administrators to upload single or multiple PDF result files for automated data extraction.

3. **Upload Excel:** Allows administrators to upload Excel result files for analysis, visualization, or KT detection.

4. **Perform Each Semester's Percentage Analysis:** Processes single PDF files to extract subject-wise marks and calculate individual semester percentages.

5. **Get KT Students:** Analyzes Excel files to identify students with backlogs (KT subjects) and highlights their names in red color for easy identification.

6. **Get Multiple Semester's Percentage and Average:** Processes multiple PDF files from different semesters, calculates semester-wise percentages, and computes overall average performance.

7. **View Visualized Data of Excel:** Generates statistical visualizations such as pass/fail bar charts to display student performance trends across courses.

Relationships:

- **Association:** Connects the Administrator actor with all use cases they can interact with.

- **Include Relationships:**
  - Explore Upload Options includes Upload PDF and Upload Excel.
  - Upload PDF includes Perform Each Semester's Percentage Analysis and Get Multiple Semester's Percentage and Average.
  - Upload Excel includes Get KT Students, Get Multiple Semester's Percentage and Average, and View Visualized Data of Excel.

- **Extend Relationships:**
  - Upload PDF may extend to View Visualized Data of Excel as an optional feature for generating performance charts.

2. DFD (Data Flow Diagram):

The Data Flow Diagram (DFD) shown in Fig 5.2 represents the flow of data within ScanWise, showing how uploaded files are processed into analysed reports.

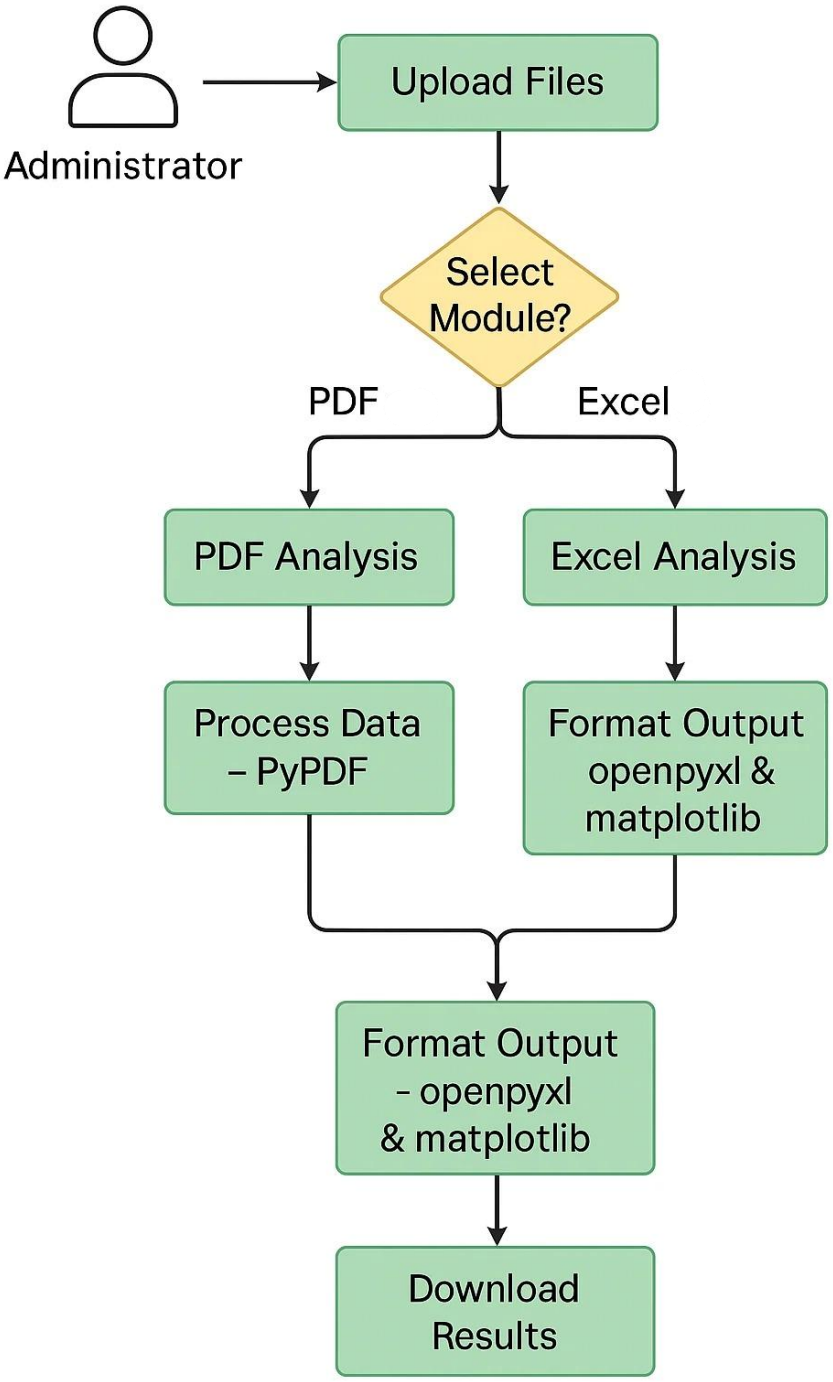

Fig 5.2: Data Flow Diagram

1. **User Interaction:**
    - The administrator or faculty uploads result files (either PDF or Excel) into the system through the ScanWise interface.
    - The system automatically identifies the uploaded file type and selects the appropriate module — PDF Analysis or Excel Analysis — for further processing.
2. **Data Extraction and Processing:**
    - PDF Analysis: The system uses the PyPDF2 library to extract important data such as student names, roll numbers, subject codes, and marks from unstructured PDF files.
    - Excel Analysis: For Excel files, the system reads and processes the data to calculate totals, averages, and pass/fail results.
    - The extracted data is then cleaned, normalized, and structured using pandas, ensuring that the dataset is accurate, consistent, and analysis-ready.
3. **Output Formatting and Visualisation:**
    - The cleaned and processed data is formatted using openpyxl to create a well-structured and visually clear Excel report.
    - Additionally, matplotlib generates performance charts such as bar graphs and pie charts to display subject-wise analysis, student performance trends, and comparison results.
    - Failed students are automatically highlighted for easier identification.
5. **Result Generation:**
    - Once the formatting and visualization are complete, the system compiles the final output into an Excel file.
    - The analyzed and formatted report can then be downloaded directly through the interface, providing users with a professional, well-organized, and easy-to-understand summary of academic results.

3. System Architecture:

The System Architecture (Fig 5.3) illustrates the step-by-step workflow of ScanWise, showing how PDF and Excel files are processed, analysed, and converted into structured output reports.

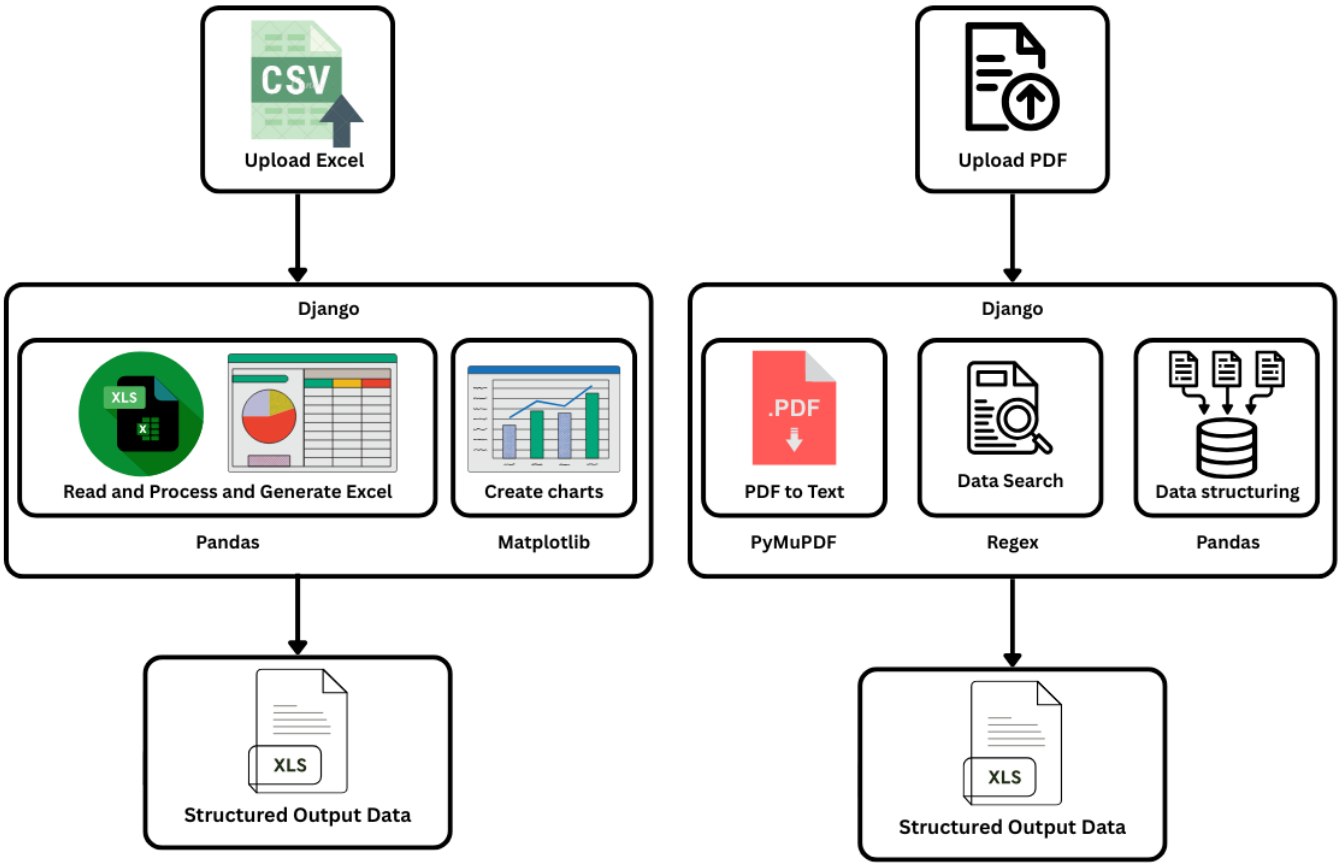

Fig 5.3: System Architecture

1. **Data Collection:**

   Faculty uploads result files in PDF or Excel format into the system for analysis.

2. **Pdf Preprocessing:**

   The uploaded PDF files are read using PyPDF2 and converted into text.

   Student details, marks, and remarks are extracted using regular expressions (Regex).

3. **Excel Processing:**

If an Excel file is uploaded, it is processed using the Pandas library. The system automatically computes pass/fail counts, average percentages, and KT analysis, preparing data for visualisation and report generation.

4. **Data Cleaning and Structuring:**

Extracted data often contains inconsistencies like extra spaces, OCR errors, or missing values. ScanWise performs data cleaning and normalisation to ensure uniform and accurate data representation before analysis.

5. **Data Processing:**

Once the data is structured, it is processed using Pandas to calculate totals, percentages, and averages for each student.

This phase integrates multiple semester data and prepares it for visualisation and export.

6. **Visualization and Formatting:**

The processed data is visually represented using Matplotlib to create charts showing pass/fail statistics per subject. Additionally, the results are formatted using OpenPyXL to produce a professional and well-structured Excel report

7. **Output Generation**

After all analysis and visualization steps, the final report is generated in Excel (.xlsx) format.

This report includes computed results, visual charts, and highlighted failed students, allowing faculty to download and review performance insights easily.

4. Implementation:

The Implementation section demonstrates the practical application of the ScanWise system through its core functionalities and user interface. This chapter showcases how the system transforms PDF and Excel result files into structured, analyzed data outputs. The following figures illustrate the step-by-step implementation of both PDF and Excel analysis modules, highlighting the user interface design, data processing workflows, and final output generation. Each component has been developed using Python libraries including PyPDF for PDF extraction, pandas for data manipulation, openpyxl for Excel formatting, and matplotlib for visualization, ensuring robust and efficient result processing.

The home screen shown in Fig 5.4 provides a clean and intuitive entry point with two primary navigation buttons: PDF Analysis and Excel Analysis. This minimalist design ensures administrators can quickly identify their required functionality without confusion. The red-themed interface with bold white text creates high contrast for better readability and professional appearance.

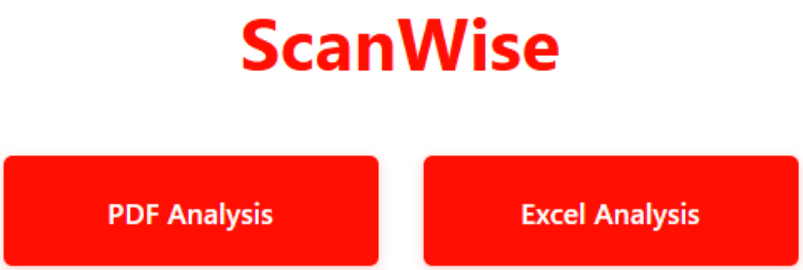

Fig 5.4: ScanWise Home Screen

Upon selecting PDF Analysis, administrators are presented with two processing options: Single PDF Analysis and Multiple PDF Analysis as illustrated in Fig 5.5. Single PDF Analysis extracts subject-wise marks and calculates percentages for individual semester results. Multiple PDF Analysis performs batch processing of multiple semester files, computing semester-wise percentages and generating average performance metrics.

Fig 5.5: PDF Analysis Mode Selection

The Single PDF Analysis interface shown in Fig 5.6 allows administrators to upload result PDF files through a standard file browser, with "SEM1.pdf" selected in this example. The interface includes clearly labeled action buttons: Submit for processing, Cancel for resetting selection, View JSON for inspecting raw data, and Download Excel for obtaining formatted output. This user-centric design ensures accessibility for users with varying technical expertise levels.

Fig 5.6: Single PDF File Upload Interface

The Multiple PDF Analysis interface presented in Fig 5.7 enables administrators to upload result files from different semesters simultaneously, with "SEM1.pdf" and "SEM2.pdf" selected in this implementation. The system extracts marks from each semester independently, calculates individual semester percentages, and computes the overall average percentage across both terms. This feature is particularly useful for evaluating cumulative performance trends over multiple examination cycles.

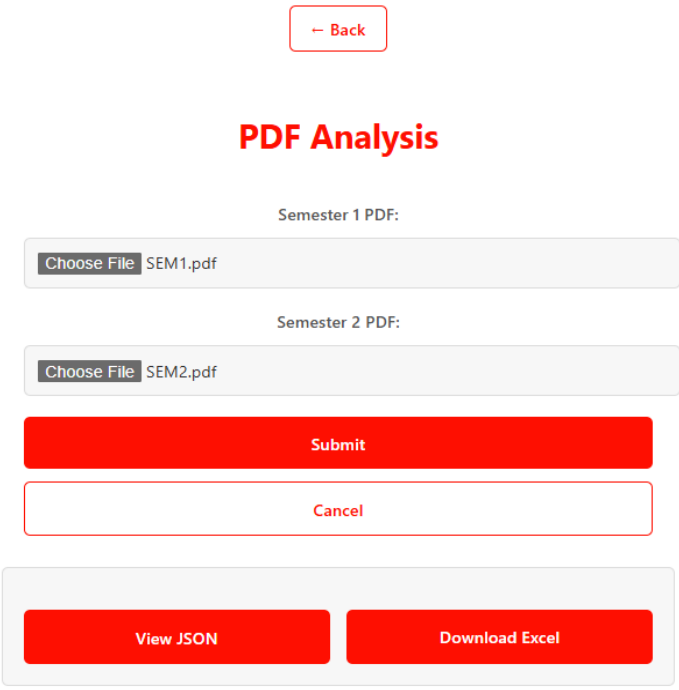

Fig 5.7: Multiple PDF Upload Interface for Semester Comparison

The Excel Analysis module shown in Fig 5.8 provides three specialized functionalities: Get KT Students for identifying students with backlogs, Pass/Fail Analysis for generating visual performance reports, and Average Percentages for multi-semester analysis. This modular approach allows administrators to perform targeted analysis based on specific institutional requirements. Each option operates independently with its own data processing pipeline ensuring system maintainability.

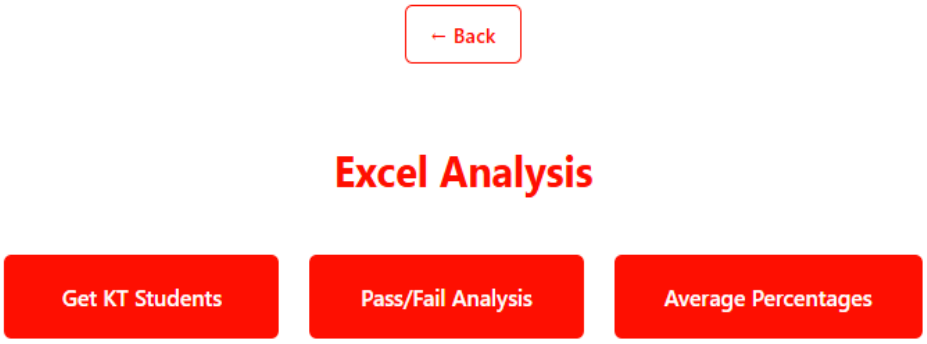

Fig 5.8: Excel Analysis Options Menu

The Get KT Students feature displayed in Fig 5.9 enables identification of students with KT (Keep Term) or backlog subjects through automated scanning. Administrators upload Excel files containing examination results, and the system identifies students based on marks below passing threshold or grades marked as 'F'. The implementation uses openpyxl to apply conditional formatting, highlighting names in red color for easy identification.

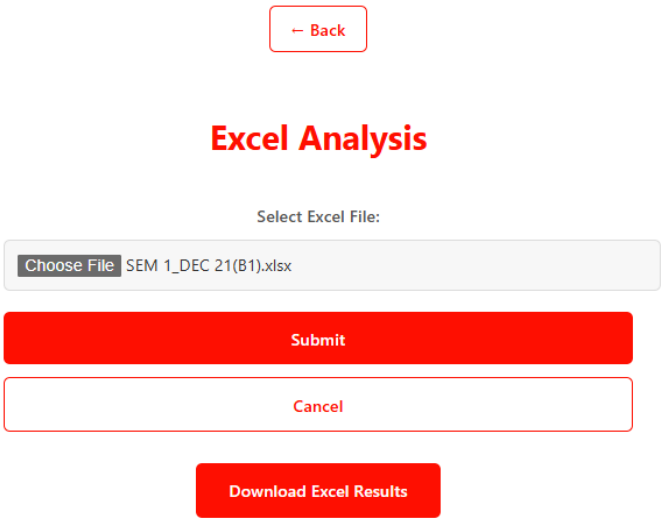

Fig 5.9: KT Students Detection - File Upload Interface

The output generated by the Get KT Students module, as presented in Fig 5.10, displays a comprehensive examination marks checking report. Students who have KT subjects are highlighted with their names displayed in red text, making them immediately identifiable among the complete student roster.

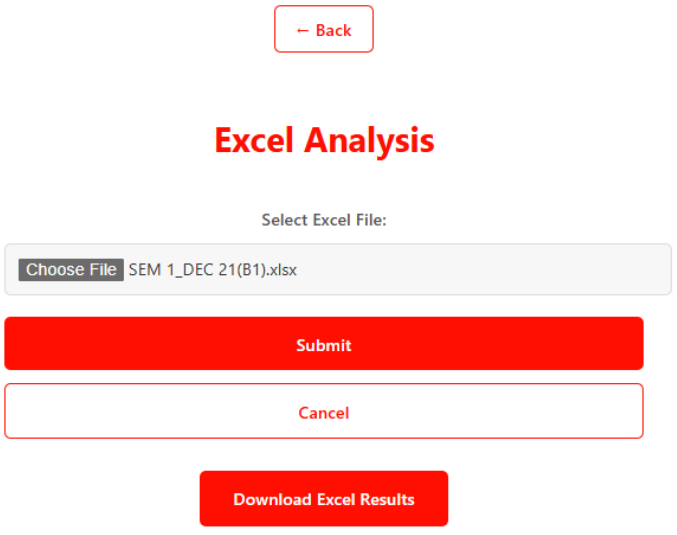

Fig 5.10: KT Students Report with Conditional Formatting

The visualization output presented in Fig 5.11 displays a stacked bar chart titled "Pass vs Fail Count per Course" with each bar representing different courses. Green segments indicate passing students while red segments show failures, with numerical values displayed within each segment for exact quantification. This graphical representation enables quick comparative analysis across subjects, helping identify courses requiring intervention or curriculum adjustment.

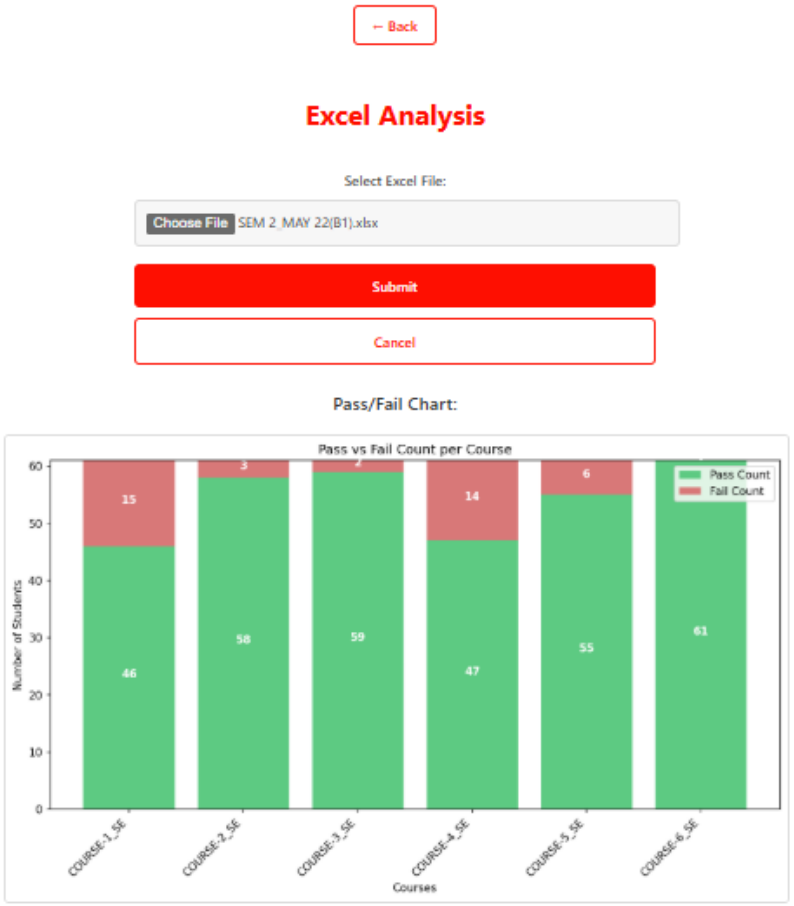

Fig 5.11: Pass vs Fail Count per Course Bar Chart

The Average Percentage feature illustrated in Fig 5.12 enables comprehensive multi-semester performance analysis by allowing upload of up to 8 semester Excel files simultaneously. In this implementation, Semester 1 and Semester 2 files have been selected, with remaining slots available for additional terms. The system extracts percentage data from each semester, computes individual performances, and calculates overall average across all uploaded terms.

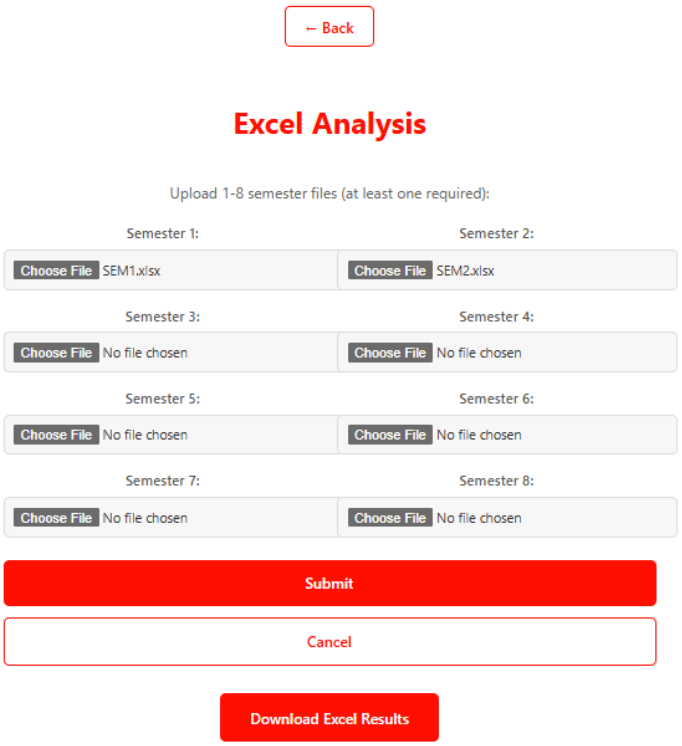

Fig 5.12: Average Percentage Calculation - Multi-Semester Upload

Chapter 6

Technical Specification

The technical specifications of ScanWise define the architectural components, tools, and technologies that support its development and execution. This section provides an in-depth overview of the technical ecosystem that powers the Marksheet Analyzer Portal, highlighting how each element contributes to automation, efficiency, and data accuracy.

ScanWise is designed as a lightweight yet powerful web-based application that automates the complex process of extracting and analyzing unstructured academic result data. Its architecture is structured around Python's data processing capabilities and Django's reliable backend framework, enabling smooth communication between the user interface and the analytical engine. The system is engineered to process user-uploaded PDF mark sheets in real time, extract key information such as student details, subjects, and marks, and generate well-organized Excel reports accompanied by insightful visualizations.

- **Frontend:**

    1. React
    2. TailwindCSS

Technologies: The frontend of ScanWise is developed using React and TailwindCSS. These core web technologies ensure a clean, intuitive, and responsive interface that enables users to easily interact with the system. No external styling framework is used; all layout and design elements are implemented using custom CSS to maintain lightweight performance and design flexibility. It provides interactive sections for file upload, analysis initiation, and result visualization, ensuring a smooth and accessible user experience.

Python Version: Python 3.11.0 is utilized for scripting and automation tasks within the frontend environment, enhancing features such as dynamic content loading and real-time updates.

- **Backend Development:**

    3. Django 4.2.5

Framework: ScanWise employs Django as its backend framework. Django's modular and robust architecture handles request routing, file management, and the integration of Python-based data-processing scripts. The backend ensures seamless coordination between the user interface and the data-analysis modules.

Database: Unlike conventional web applications, ScanWise does not rely on a persistent database. Instead, all operations are performed dynamically on user-uploaded files. Once the analysis is complete, results are displayed or downloaded, eliminating the need for data storage and ensuring user privacy and data security

These technical specifications outline a comprehensive and forward-thinking approach, ensuring that ScanWise remains at the cutting edge of technology for pdf files to result excel generation. The integration of advanced tools and methodologies allows ScanWise to deliver an accurate excel sheet upon which further operations can be performed.

- **Core Libraries:**

PyPDF2 – Used to extract unstructured data such as student details, subject codes, and marks from uploaded PDF result files. It enables text parsing and segmentation for subsequent processing.

Pandas – Manages and structures extracted data into tabular formats, supporting operations like filtering, cleaning, and transformation before exporting to Excel.

openpyxl – Used to generate well-formatted Excel files. It enhances readability through styling, alignment, and font customization.

matplotlib – Produces graphical visualizations such as bar charts and pie charts to illustrate subject-wise performance, pass/fail ratios, and other analytical insight

Chapter 7

Project Scheduling

In project management, a schedule is a structured listing of a project's milestones, activities, and deliverables. The following schedule (Table 7.1) outlines the systematic development of the ScanWise – Marksheet Analyzer Portal, highlighting the major stages from conceptualization to deployment.

Sr. No.	Group Members	Duration	Task Performed
1.	Aayush Balip, Ashokkumar Bhati, Saiyed Sameer, Keval Shah	2nd Week of July	Group formation and topic finalization. Identifying scope and objectives of the project. Discussing the project concept and creating an initial paper prototype.
		1st Week of August	Identifying the key functionalities of project and planning feature for PDF extraction, Excel generation, and visualization.
2.	Aayush Balip, Ashokkumar Bhati, Saiyed Sameer	2nd Week of August	Developing and testing data extraction modules using PyPDF2 to parse and preprocess student result PDFs.
3.	Aayush Balip	3rd Week of August	Designing the Graphical User Interface (GUI) and dashboard for user interaction.
4.	Keval Shah, Saiyed Sameer, Ashokkumar Bhati	4th Week of August	Implementing core features such as data cleaning with pandas, Excel report creation using openpyxl, and graph generation through matplotlib
5.	Aayush Balip, Ashokkumar Bhati	1st Week of September	Integrating the Django backend to connect all modules, handle file uploads, and coordinate frontend-backend communication.
6.	Saiyed Sameer, Keval Shah	3rd Week of September	Testing and deployment of ScanWise on Vercel. Ensuring smooth functionality, secure file handling, and accurate generation of analytical reports.

Table 7.1: Project Task Distribution

A Gantt chart is a type of bar chart that illustrates a project schedule. This chart lists the tasks to be performed on the vertical axis, and time intervals on the horizontal axis. Gantt chart (Fig 7.1) illustrates the start and finish dates of the terminal elements and summary elements of a project.

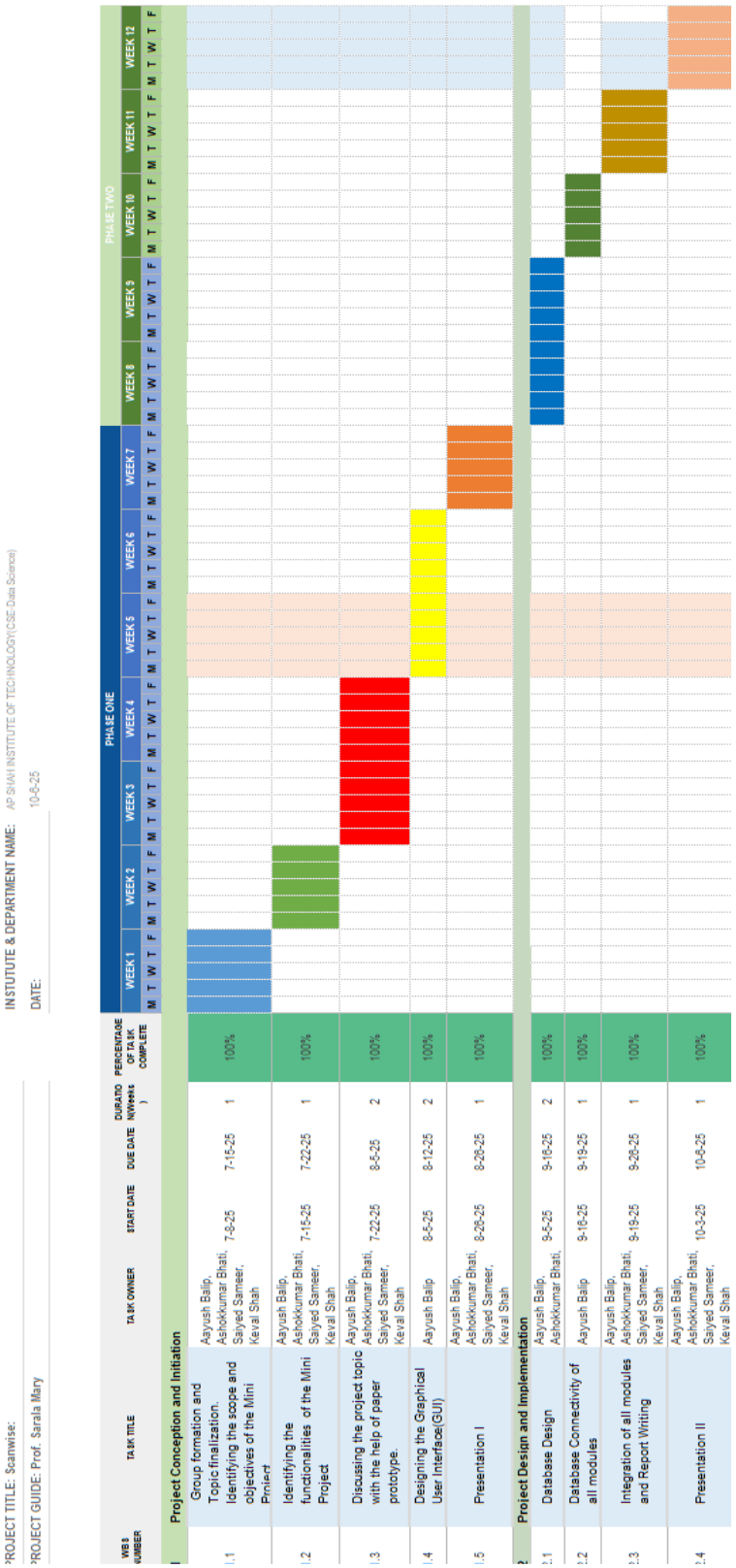

Fig 7.2: Gantt Chart of ScanWise

Chapter 8

Results

The project results section provides a concise overview of the outcomes achieved through the implementation of the project. Highlighting key findings, deliverables, and the final implementation of the project lifecycle. This section showcases the user-friendly interface design, the PDF and Excel analysis capabilities, and the structured output generated by the system. The screenshots presented below illustrate the complete workflow from the home screen to the final Excel outputs, highlighting the automation and accuracy achieved through the integration of PyPDF, pandas, openpyxl, and matplotlib libraries.

Multiple PDF Processing Performance

The system supports processing up to 2 PDF files simultaneously for multi-semester analysis. Table 8.1 presents the performance metrics for different PDF processing scenarios. The graph representing this can be seen in Fig 8.1.

Number of PDFs	**Processing Time (sec)**	**Merging Time (sec)**	**Total Time (sec)**	**Average Time per record (sec)**
1 PDF	12	0	12	0.55
2 PDF	23	3	26	0.65

Table 8.1: Multiple PDF Analysis Performance Metrics

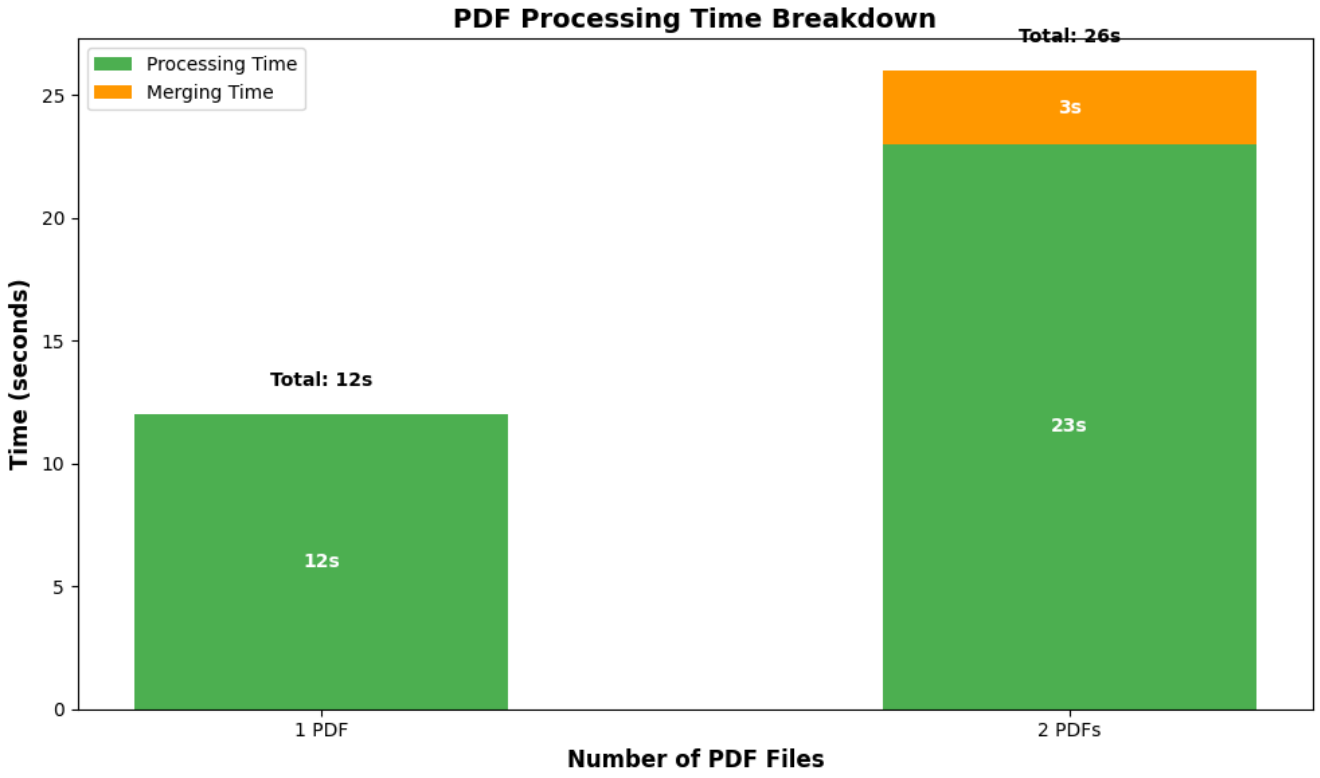

Fig 8.1: PDF Processing Time vs Number of Files

Multiple Excel File Processing Performance

The system supports processing up to 8 Excel files simultaneously for comprehensive multi-semester average analysis. Table 8.2 demonstrates scalability across different Excel file counts.

Number of Excel Files	File Reading Time (sec)	Calculation Time (sec)	Formatting Time (sec)	Total Time (sec)	Time per File (sec)
1 Excel	2	1	2	5	5.0
2 Excels	4	2	3	9	4.5
3 Excels	6	3	3	12	4.0
4 Excels	8	4	4	16	4.0
6 Excels	12	6	5	23	3.8
8 Excels	16	8	6	30	3.75

Table 8.2: Multiple Excel Analysis Performance Metrics

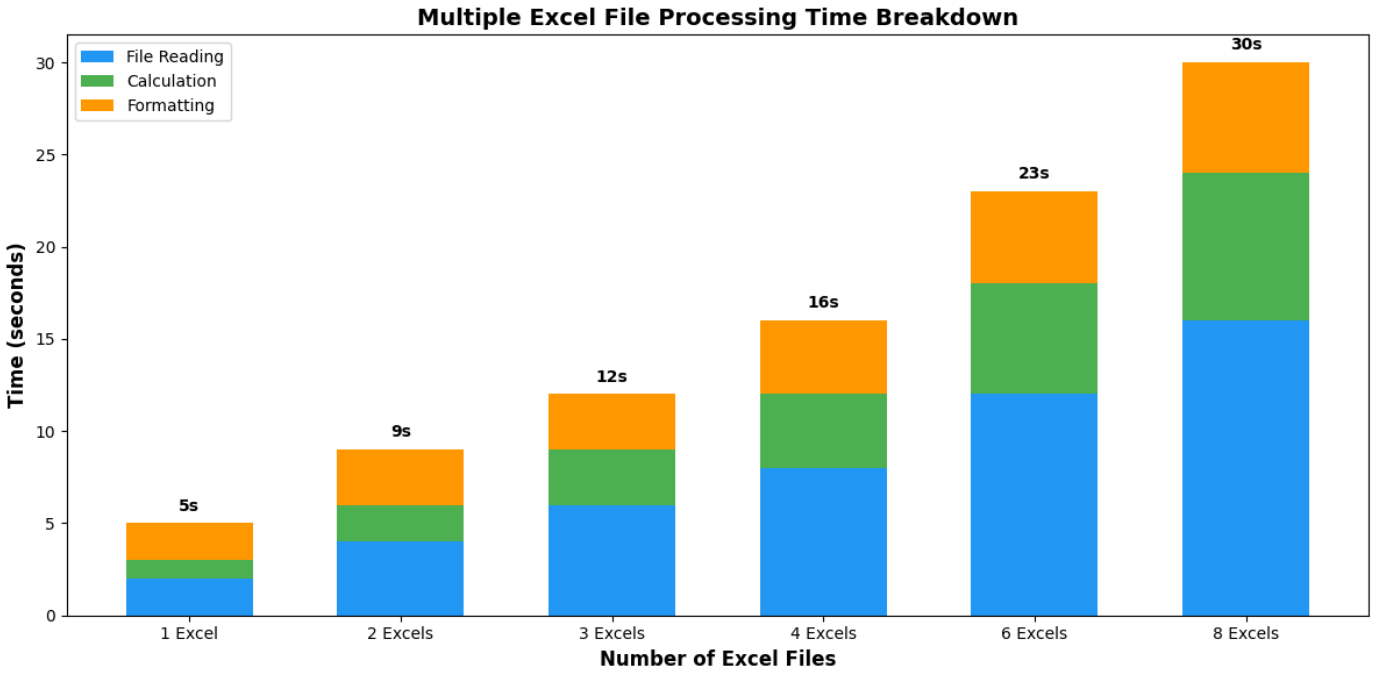

Fig 8.2: Excel Processing Time Distribution

Scalability Analysis - PDF vs Excel Processing

The comparative scalability analysis demonstrates that Excel processing achieves significantly higher efficiency than PDF processing. This performance differential is primarily attributed to the structured data format of Excel files, which eliminates the text parsing overhead required for PDF documents. The detailed processing metrics and comparative efficiency data are presented in Table 8.3, with visual performance comparison illustrated in Fig 8.3.

Module	**Maximum Files Supported**		**Time per File (sec)**	**Accuracy (%)**
Single PDF Analysis	1		12-25	98.5
Multiple PDF Analysis	2		12-78	99.0
Single Excel Analysis	1		5	100
Multiple Excel Analysis	8		3.75	100

Table 8.3: Comparative Processing Efficiency

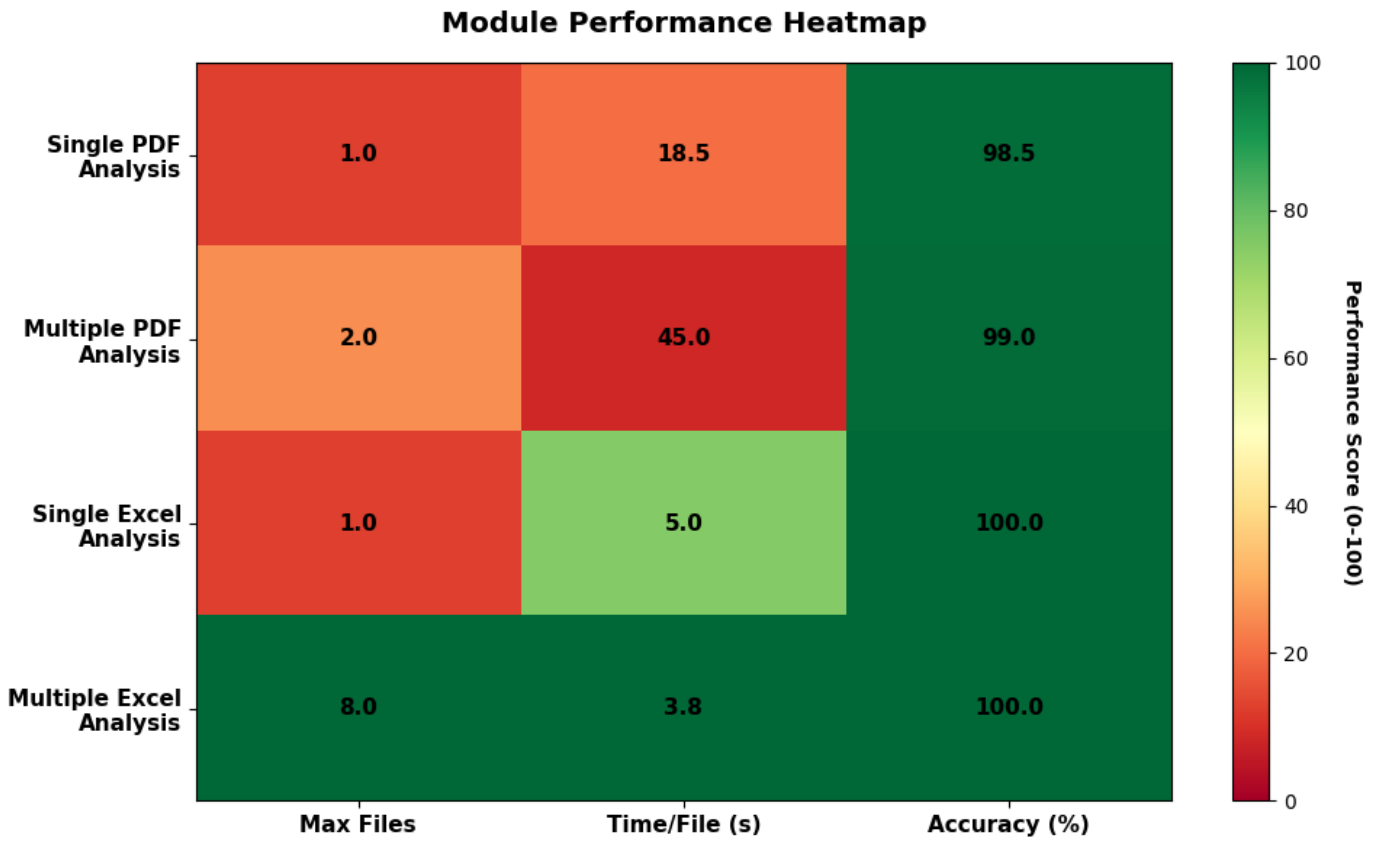

Fig 8.3: Comparative Processing Time Distribution of PDF and Excels

Data Accuracy Across Multiple File Processing

Data accuracy validation was conducted across various file processing scenarios, testing both single and multiple file operations with different file types. Complete accuracy metrics and error analysis are documented in Table 8.4, with accuracy distribution visualization shown in Fig 8.4.

File Type	Files Processed	Correctly Extracted	Extraction Errors	Calculation Errors	Overall Accuracy (%)
1 PDF	1	103	0	0	100
2 PDFs	2	206	0	0	100
4 Excels	4	248	0	0	100
8 Excels	8	496	0	0	100

Table 8.4: Accuracy Validation for Multi-File Operations

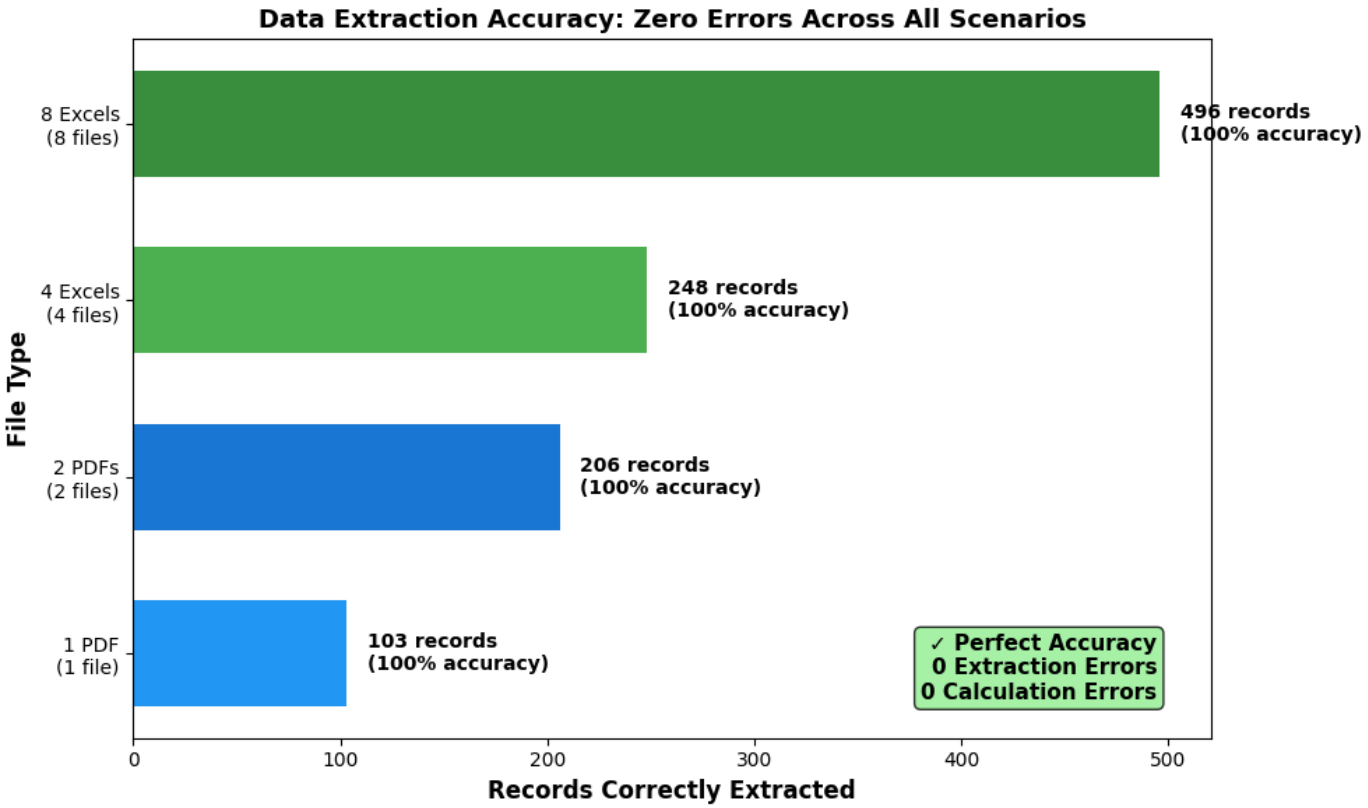

Fig 8.4: Accuracy Validation for Multi-File Operations

Excel Analysis Module Performance

The Excel Analysis module demonstrates robust performance across three distinct operations: KT Detection, Pass/Fail Chart generation, and multi-semester average calculation. Detailed performance breakdown for all Excel analysis operations is provided in Table 8.5.

Analysis Type	Excel Files	Processing Time (sec)	Output Generation Time (sec)	Total Time (sec)
KT Detection	1	5	3	8
Pass/Fail Chart	1	4	4	8
Average (2 semesters)	2	7	2	9
Average (4 semesters)	4	12	4	16
Average (6 semesters)	6	18	5	23
Average (8 semesters)	8	24	6	30

Table 8.5: Excel Analysis Operations Performance

KT Detection Statistics for SEM-2

- Students Analyzed: 61
- Students with KT: 21 (34.4%)
- Detection Accuracy: 100%
- Processing Speed: 7.5 records/second

Pass/Fail Visualization for SEM-2:

- Courses Analyzed: 6
- Overall Pass Rate: 85.3%
- Chart Generation Time: 4 seconds

System Efficiency Comparison

System efficiency analysis reveals exceptional automation benefits, with ScanWise achieving 99.5-99.8% efficiency gains across all processing tasks compared to manual methods. The comprehensive efficiency comparison between manual and automated processing is detailed in Table 8.6, with efficiency gain distribution visualized in Fig 8.6.

Task	**Manual Time (min)**	**ScanWise Time (sec)**	**Time Saved**	**Efficiency Gain (%)**
Single PDF Extraction	120	12	119m 48s	99.8
Multiple PDF (2 files)	250	26	249m 34s	99.8
Single Excel Analysis	30	5	29m 55s	99.7
Multiple Excel (8 files)	120	30	119m 30s	99.5
KT Detection	10	8	9m 52s	99.6

Table 8.6: Processing Time - Manual vs Automated

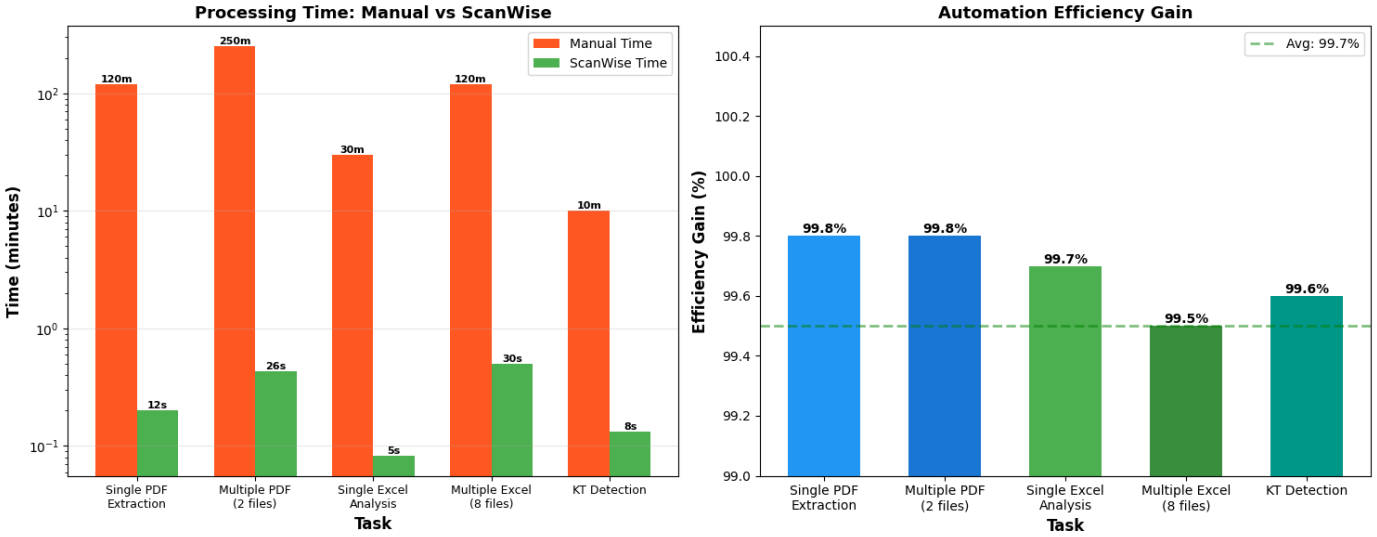

Fig 8.6: Processing Time and Efficiency gain - Manual vs Automated

Chapter 9

Conclusion

The ScanWise project successfully addresses the critical challenges faced by educational institutions in processing and analyzing examination results through comprehensive automation and intelligent data processing. By leveraging the power of Python and its robust libraries—PyPDF for data extraction, pandas for data manipulation, openpyxl for Excel formatting, and matplotlib for visualization—the system transforms unstructured PDF result files into organized, readable, and analysis-ready Excel sheets with advanced analytical capabilities.

ScanWise provides dual functionality through **PDF Analysis** and **Excel Analysis** modules, each designed to handle specific institutional requirements. The PDF Analysis module offers both Single and Multiple PDF processing modes—Single PDF Analysis efficiently extracts subject-wise marks and calculates percentages for individual semester results, while Multiple PDF Analysis performs batch processing of multiple semester files, computing semester-wise percentages and generating average performance metrics. The Excel Analysis module extends functionality with three specialized features: Get KT Students for identifying students with backlogs through conditional formatting, Pass/Fail Analysis for generating visual performance reports with bar charts, and Average Percentage calculation for multi-semester cumulative performance tracking.

The automation of repetitive tasks such as data extraction, formatting, calculation, and visualization significantly reduces manual effort, minimizes human errors, and accelerates result processing workflows. The system's intuitive interface combined with powerful backend processing capabilities makes it accessible to users with varying levels of technical expertise—from students seeking quick access to their performance data to administrators managing large volumes of result documents across multiple terms. The addition of visual analytics through matplotlib-generated charts provides immediate insights into performance distributions, failure patterns, and student progress trends, enabling data-driven academic decision-making.

ScanWise represents a significant step forward in modernizing educational data management by replacing time-consuming manual operations with intelligent automation. The system not only saves time but also ensures accuracy and consistency in result management, making it an invaluable tool for contemporary educational institutions.

Chapter 10

FutureScope

While ScanWise has successfully implemented automated result extraction and analysis, there are several promising directions for future enhancements that can expand its functionality, improve user experience, and increase its applicability across diverse educational contexts.

1. **Advanced Data Visualization and Performance Analytics**:
   Future versions of ScanWise can integrate comprehensive data visualization features using matplotlib and seaborn libraries to generate interactive charts, graphs, and dashboards. These visualizations can include subject-wise performance comparisons, batch-wise trend analysis, topper identification charts, semester-to-semester progression tracking, and performance heatmaps. By providing graphical insights beyond the current pass/fail charts, the system will enable educators and administrators to quickly identify performance patterns, subject difficulty levels, and areas requiring academic intervention, thereby supporting data-driven decision-making in educational planning and curriculum development.

2. **Support for Multiple PDF Formats and Layouts**:
   Currently, ScanWise is optimized for specific PDF structures. Enhancing the system to recognize and parse multiple result formats from different universities, colleges, and examination boards will significantly increase its versatility. Implementing machine learning-based layout detection or template matching algorithms using libraries like OpenCV and TensorFlow can allow the system to automatically identify table structures, header positions, and data fields regardless of format variations. This adaptability will make ScanWise a universal solution for result processing across diverse academic institutions, from state boards to autonomous universities.

3. **Real-time Cloud Integration and Database Storage**:
   Integrating cloud storage solutions such as Google Drive, Dropbox, AWS S3, or institutional cloud servers will enable users to upload, process, and retrieve results from anywhere with internet access. Additionally, implementing a database backend using MySQL, PostgreSQL, or MongoDB can facilitate centralized storage of historical result data, enabling longitudinal performance tracking, batch comparisons across academic years, and generation of comprehensive academic transcripts. Cloud integration will also support collaborative access for multiple users within an institution and provide automatic backup and disaster recovery capabilities.

4. **Mobile Application Development**:

   Developing a mobile application version of ScanWise for Android and iOS platforms will enhance accessibility and convenience for students and faculty. Using frameworks like React Native or Flutter, the mobile app can provide on-the-go result analysis, push notifications for new result uploads, offline access to previously processed data, and QR code scanning for quick result retrieval. Mobile integration will make the system more user-friendly, especially for students who prefer accessing academic information through their smartphones, and will support real-time performance monitoring.

5. **Intelligent Result Prediction and Academic Counseling**:

   Incorporating machine learning models using scikit-learn or TensorFlow to analyze historical performance data can enable predictive analytics for student outcomes. The system could predict future performance trends based on current semester patterns, identify students at risk of academic failure using predictive algorithms, forecast KT (backlog) likelihood, and suggest personalized improvement strategies. Additionally, integrating an AI-powered chatbot using natural language processing can provide students with academic counseling, study resources, and career guidance based on their performance patterns and subject strengths.

6. **Multi-language Support and Accessibility Features**:

   Expanding ScanWise to support multiple languages (Hindi, Marathi, regional languages) will make it accessible to a broader user base, including regional language speakers in diverse educational institutions across India. Implementing text-to-speech functionality for result reading, screen reader compatibility for visually impaired users, high-contrast display modes, and keyboard navigation support will ensure that the system is inclusive and usable by students and staff with visual impairments or other accessibility needs, promoting digital inclusivity in education.

7. **Automated Report Generation and Digital Certificates**:

   Future enhancements can include automated generation of detailed performance reports, grade sheets, mark sheets, and academic transcripts in PDF format with institutional branding, logos, watermarks, and digital signatures for authenticity. The system can also integrate with Learning Management Systems (LMS) like Moodle, Google Classroom, or Canvas to automatically update student dashboards with their latest results, attendance records, and academic standings. Additionally, blockchain-based digital certificate verification can be implemented to prevent document forgery and enable instant credential verification by employers or higher education institutions.

8. **Security Enhancements and User Authentication**:

   Implementing robust security measures such as role-based user authentication (student, faculty, admin roles), OTP-based login, two-factor authentication (2FA), and encrypted data transmission using SSL/TLS protocols will protect sensitive student information from unauthorized access. Adding features like audit logs for tracking all system activities, data encryption at rest using AES-256, session timeout mechanisms, and CAPTCHA for preventing automated attacks will ensure that only authorized personnel can upload, process, or view result data, maintaining data privacy and compliance with educational data protection regulations like the Digital Personal Data Protection Act (DPDP) 2023.

9. **Comparative Performance Analytics and Benchmarking**:

   Implementing advanced analytics features that allow comparison of student performance across different batches, divisions, academic years, and even departments will provide valuable insights for institutional improvement. The system can generate comparative reports showing year-over-year performance trends, subject-wise difficulty analysis across cohorts, and benchmarking against university or national averages. These analytics will help academic committees identify systemic issues, evaluate teaching effectiveness, and make informed decisions about curriculum modifications and resource allocation.

10. **Integration with Attendance and Internal Assessment Systems**:

    Expanding ScanWise to integrate with existing attendance management systems and internal assessment databases will create a comprehensive academic analytics platform. By correlating attendance patterns with academic performance, identifying correlations between internal assessment scores and final exam results, and providing holistic student profiles combining attendance, internal marks, and semester results, the system can offer deeper insights into factors affecting student success. This integrated approach will support early intervention strategies and personalized academic counseling based on multiple performance indicators.

REFERENCES

[1] Kumar, R., Patel, S., and Singh, M. "Automated Student Result Management System Using Excel and Python." International Journal of Computer Applications 175.8 (2020): 12-18.

[2] Oladipo, E. O., and Olatunji, O. A. "Design of an Educational Result Processing System for Higher Institutions." Journal of Information Technology and Computer Science 11.5 (2019): 45-52.

[3] Sharma, A., and Gupta, D. "Automation of Examination Results Using Python and Pandas." Proceedings of the 2022 International Conference on Data Science and Engineering (ICDSE). IEEE, 2022.

[4] Verma, N., and Reddy, P. "Data Visualization for Academic Performance Analysis Using Python." International Journal of Educational Data Mining 13.3 (2021): 78-89.

[5] McKinney, Wes. "Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython." O'Reilly Media, Inc., 2012.