Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

Auto ITR: An Automated Indian Income Tax Return Preparation System using Hybrid Rule-Based and Heuristic AI

Haireet Mehta
Department of Artificial Intelligence and Data Science
Shah and Anchor Kutchi Engineering College
Chembur, India

Shubham Jain
Department of Artificial Intelligence and Data Science Shah and Anchor Kutchi Engineering College Chembur, India

3rd Kaushal Mhatre
Department of Artificial Intelligence and Data Science
Shah and Anchor Kutchi Engineering College
Chembur, India

4th Shreyansh Jain
Department of Artificial Intelligence and Data Science Shah and Anchor Kutchi Engineering College Chembur, India

5th Prof. Milind Khairnar
Department of Artificial Intelligence and
Data Science Shah and Anchor Kutchi
Engineering College Chembur, India

***Abstract*—Filing income tax returns (ITR) in India manually may require considerable effort and be riddled with errors. This process typically includes classifying financial transactions, accounting for multiple bank accounts and various deductions, such as section 80C, 80D, 24(b), and 80CCD.**

In this research paper, we present Auto ITR, which is an automated solution aimed at easing the procedure of filing income taxes. The proposed platform automatically processes raw statements from banks, classifies transactions, detects anomalies and suspicious entries, provides users with recommendations on saving more on their income tax, and lets them review all the generated data with a Chartered Accountant.

This platform combines tax computations on the basis of India tax brackets and smart functionality, such as the smart transaction categorizer, anomaly detection, comparison of two regimes of taxation, and a tax optimizer along with a knowledge-based chatbot.

Experimental results show strong performance, with transaction categorization achieving 91.2% precision and 88.5% recall, and anomaly detection reaching 93.1% accuracy. Additionally, the system reduces manual effort by around 65%. A CA review acceptance rate of 87% further demonstrates the reliability and practical usability of the system.

***Index Terms*—automated income tax, automated transaction classification, automated anomaly detection, automated tax optimization, India ITR**

I. Introduction

Income tax return filing in India mandates taxpayers to aggregate financial transactions from various banks, categorize their sources of income, calculate permissible deductions, and make a choice between the old and new tax regime under the Finance Acts. The task is especially arduous for salaried individuals who invest in shares and mutual funds, own properties, have home loans, and make investments in schemes like the National Pension Scheme and Equity Linked Saving Schemes (ELSS). Inaccuracy at the financial transaction categorization phase leads to erroneous calculation of deductions and possibly even notices from the Income Tax Department (ITD).

While commercial solutions offer guided data entry, they do not perform the tasks of extraction and categorization of financial transactions from bank statements automatically. Current research related to financial transaction categorization [1] and income tax documentation [2] have considered these problems in isolation.

Auto ITR takes the challenge all the way from multi-format ingestion of bank statements to CA-approved export. The main contributions of our work are: (1) Hybrid rule- and heuristic-based transaction categorizer with user correction feedback;(2) Anomaly detection framework based on statistical and heuristic methods; (3) Dual regime tax optimizer accounting for slab rates, rebates, and cess; (4) Chatbot with knowledge base first followed by optional LLM; (5) Human-in-the-loop CA export approval phase.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

II. Related Work

A. Automated Financial Document Processing

In literature related to automating personal finance management, previous efforts have concentrated on expense tracking [3] and budget classification through machine learning techniques [4]. Rule-based approaches perform better in structured merchant information but fare poorly on unstructured bank narratives typical of India, whereas hybrid methods which employ keyword matching along with recurrence have higher recall for salaries and EMI payments, hence the motivation behind our system SmartCategorizer.

B. Tax Computation and Advisory Systems

In the case of automated tax advice in western countries, there is the advantage of standard forms of data input (W-2 form, 1099 form). In the Indian setting, however, there are issues such as the presence of a dual tax regime, various sections for deductions, and different format for PDF/XLS bank statements. From studies conducted in [6], the importance of selecting an appropriate regime at marginal income levels is not a simple task; this is solved in our TaxOptimizer using break-even deductions.

III. System Architecture

Auto ITR follows a layered architecture comprising a FastAPI backend, a SQLite/SQLAlchemy data layer, and a static HTML/CSS/JavaScript frontend served by the same API process.

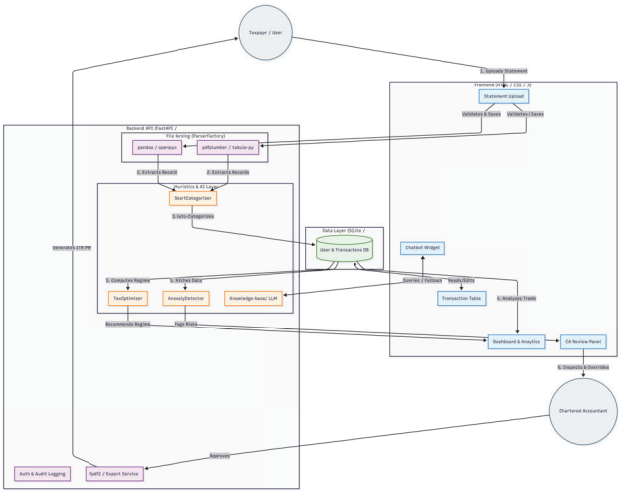

Fig. 1. Overall System Architecture of Auto ITR

A. Backend

The backend has been developed in Python 3.11 with FastAPI. Routers have been used to create different APIs to handle tasks such as authentication, user management, handling bank statements, analytics, AI Inference, ITR filing, Balance Sheet management, CA Review, exporting data, and consolidation of multiple banks. All endpoints use JWT for authentication and bcrypt for password encryption.

B. Data Layer

The persistence of the system relies on the use of SQLite, realized through SQLAlchemy 2.x declarative model classes that take care of the management of key objects like users, bank statements, transactions, logs, ITR submissions, tax calculations, and balance sheet objects.

Transaction entries contain an extra category name along with the confidence level and other meta information.

C. Frontend

The front-end of the system is developed using plain JavaScript modules and allows users to view their KPIs in the form of a dashboard. The dashboard contains features like a statement upload feature, transaction table with a feature to edit categories inline, analytics chart, balance sheet editor, CA review panel, and chatbot.

D. File Parsing

The `ParserFactory` module ensures that the various types of files, namely CSV, XLS, XLSX, and PDF are directed to the appropriate parsers. The parsers used for handling different file types have been built using the pandas [8], openpyxl, xlrd, pdfplumber, tabula-py, and PyPDF2 libraries. The `ParserFactory` module ensures that the various types of files, namely CSV, XLS, XLSX, and PDF are directed to the appropriate parsers. The parsers used for handling different file types have been built using the pandas [8], openpyxl, xlrd, pdfplumber, tabula-py, and PyPDF2 libraries.

Additionally, the system can handle encrypted PDF bank statements.

E. Data Flow

The end-to-end flow is:

1) The user uploads a statement; the `StatementService` checks the size and format, then stores the upload under `uploads/bank_statements/{userId}`.
2) Upon processing a request, the system invokes the `ParserFactory` to extract information; the `TransactionClassifier` initi
3) Optional auto-categorization refines uncategorized transactions via `SmartCategorizer`.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

4) Analytics endpoints aggregate income, expenses, and deductions; compute monthly trends and tax estimates.
5) AI endpoints provide chatbot responses, anomaly reports, and regime optimization.
6) CA review panel allows inspection, approval, and PDF export via fpdf2.

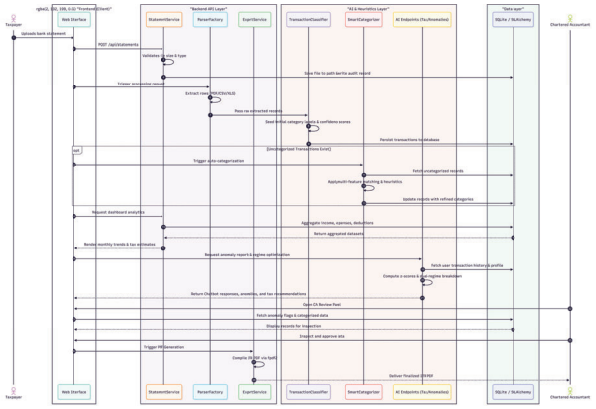

Fig. 2. End-to-End Data Flow from Statement Upload to Tax Filing

IV. Algorithms and Heuristics

A. SmartCategorizer

SmartCategorizer uses multi-features matching to classify the categories of narrations in banks. Multi-features match includes features such as keyword and regular expression match based on a well-curated taxonomy, checking of amount range (such as salary range), identification of days of the month for recurring payments, and recurrence pattern match in transaction history. The confidence level is thresholded into three categories which are 0.5 (tendency), 0.7 (likely), and 0.9 (confident). Corrections done by users are recorded and used as highest priority rules in further classification without training a new classifier.

B. AnomalyDetector

Anomaly detection combines four complementary strategies. Statistical outliers are identified using a z-score baseline:

$$z_i = \frac{x_i - \mu}{\sigma} \tag{1}$$

where x_i is the transaction amount, μ is the category mean, and σ is the standard deviation. Transactions with $|z_i| > \theta$ (configurable threshold) are flagged. Round-amount heuristics flag transactions that are exact multiples of 1000 or 5000 INR as potential cash proxies. A duplicate detector keys on the tuple (amount, description, date) within a rolling window. The system analyzes the differences between categories and credits/debits to flag errors such as a debit entry classified as salary income. The system also searches for transaction descriptions that indicate large amounts of cash movement, aiding in detecting possibly unreported cash transactions.

All flagged discrepancies are categorized into levels of severity and summarized for CA review.

C. TaxOptimizer

TaxOptimizer computes the taxable income and net tax liability using both the old and new Indian income tax schemes for Assessment Year 2024–25. Slab rates in force for that year for both schemes are applied along with deducting the 87A tax rebate for the income amounts satisfying its threshold and computing 4% health and education cess.

In aid of making decisions, the breakeven amount of deductions D^* is estimated. It denotes the total deductions needed in the old scheme such that the net tax liability in the old scheme equals that in the new scheme:

$$T_{\text{old}}(I - D^*) + \text{cess} = T_{\text{new}}(I) + \text{cess} \tag{2}$$

In view of the above calculation, the optimizer makes recommendations about the choice of the scheme that offers the better benefits along with suggestions to improve the tax position. The suggestions include maximizing deductions under 80C (upto INR 1.5 lakh), contributions to NPS under 80CCD(1B) (upto INR 50,000) and 80D.

D. Tax-Saving Analytics

These endpoints will analyze the classified transactions in order to make an estimate regarding incomes, expenditures, and the utilization of deductions in different sections. The software shall also draw attention to the marginal advantages of tax savings in Sections such as 80C, 80D, 24b and 80CCD(1B).

This will enable the user to capitalize on unused deduction chances.

E. Chatbot

The bot has a knowledge-based first design, where the initial input from the user is first tried against a manually curated set of tax topic indices in India, such as IT return selection, HRA deductions, capital gains, and advance tax.

In case no input matches the threshold criteria of confidence levels, then the input is sent to an optional LLM endpoint of OpenAI (`gpt-4o-mini`) alongside some recent chat history and system prompts ensuring ethical guidelines with no possibility of generating anything that could be construed as tax evasion advice.

The other feature offered by the bot is follow-up query suggestions.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

V. EVALUATION METHODOLOGY

A. Categorization Accuracy

Accuracy and recall are analyzed based on the annotated dataset of bank transactions narration, consisting of typical Indian categories like salary, investment, and utilities.

Confidence calibration is done by analyzing the comparison between the confidence values assigned by our algorithm and the actual accuracy within each of those ranges.

Moreover, correction lift is used to quantify the improvement in accuracy of each individual after five corrections.

B. Anomaly Detection

Anomalies seeding is done by adding known anomalies, like duplicates, abnormally high values, and wrong categories, to otherwise perfect bank account statement data.

For the assessment of the performance of the algorithm, false positive and false negative error rates are calculated at various threshold levels of $\theta \in \{2.0, 2.5, 3.0\}$.

C. Tax Optimizer Accuracy

Tax liabilities generated by both models have been analyzed based on the true tax values obtained from the slab calculator of the ITD. This analysis will be done through a database consisting of the details of taxpayers having different levels of taxable incomes from 5 lakhs up to 50 lakhs, depending on the degree of deductions made.

Evaluation will be performed through the generation of the average tax differential along with the largest observed deviation in the calculations.

D. System Latency

Performance parsing is conducted for both PDF and XLSX documents depending on page counts and row counts. Efficiency of the system is tested in terms of time delay during batching process in performing categorization and detection of anomalies for statement size counts of 100, 500, and 1000.

Also, reaction time is measured for chatbots in case of either knowledge base or language model matching processes.

E. Human-in-the-Loop (CA Review)

The acceptance rate in CA review denotes the percentage of transactions that have been auto-classified and accepted by Chartered Accountants in a pilot study without any changes made.

The manual override measure for every 100 transactions is used as an indicator of human work left to do.

VI. IMPLEMENTATION DETAILS

A. Technology Stack

It uses a wide range of technologies. The backend is implemented in Python 3.11, FastAPI, uvicorn, and has data modeling capabilities provided by SQLAlchemy 2.x and Pydantic 2.x. Environment configuration is done by Pydantic 2.x and pydantic-settings.

Authentication is implemented with help from python-jose (used for token management with JWT) and passlib[bcrypt]. Document parsing functionality is implemented with help from pandas, openpyxl, xlrd, tabula-py, PyPDF2, pdfplumber. PDF generation capabilities are provided by the fpdf2 package.

Optional integrations include the usage of an LLM with the help from OpenAI's official Python client. The base URL for this integration can be configured to allow the user to host their own instance of the model.

Automated test suites for authentication and file uploads are implemented with pytest.

B. Configuration and Deployment

Configuration of the system takes place in the `.env` file where such environment variables as JWT SECRET, EXPIRES_IN, URL_SQLITE, ALLOWED_ORIGINS, OPENAI API KEY, among others, can be defined.

While in production, the recommendation would be to switch the database from SQLite to PostgreSQL, alongside storing objects in encrypted cloud storage for scalability reasons. Also, the `ALLOWED_ORIGINS` variable should be restricted to only frontend servers.

The `SECRET_KEY` provided by default should be changed to a randomly generated secure one before deploying the application.

C. API Surface

The following APIs are exposed by the system. The `/api/auth/*` endpoints manage authentication services, whereas the `/api/statements` route manages the uploading, processing, listing, and deletion of statements.

The AI-based functionalities are offered via the `/api/ai/chat`, `/api/ai/anomalies`, and `/api/ai/optimize-tax` endpoints. The aggregated tax analytics can be accessed from the `/api/analytics/*` endpoint. There are also other endpoints like `/api/itr/*`, `/api/review/*`, and `/api/export/*`, which handle.

VII. PRIVACY, SECURITY, AND COMPLIANCE

Ownership checks are performed by the system on an individual user basis, done through the API level to prevent access overlap for users' private data. All upload, process, and delete actions are logged by the system with the associated timestamp and username.

Password hashes generated by the bcrypt algorithm are saved in the system database, while the password itself remains plaintext-free. Tokens expire after short durations but may be altered using the value in `ACCESS_TOKEN_EXPIRE_MINUTES`.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

In order to facilitate ethical use, the chatbot is designed with a system prompt where any advice regarding tax evasion is strictly forbidden. In addition, anomalies will be detected and highlighted, such as a high volume of cash transactions.

Local storage is used with SQLite and filesystem databases. For production usage, it is recommended to deploy a relational database and encrypt the data.

VIII. Limitations and Future Work

However, the existing SQLite database system does not allow efficient handling of parallel writes and backup purposes; hence, there is a need to migrate to the PostgreSQL database. The current SmartCategorizer uses heuristic techniques, but by combining simple sentence-level models with federated personalization techniques, it would enhance recognition of various types of transaction narration.

The anomaly detection technique uses static statistical values, which means that some merchants having transaction variations could be flagged as anomalies. It is possible to minimize such false detections by using season-based and peer-based comparison.

In the current system, users have to enter the necessary HRA and NPS values manually. However, the process would become simpler if the Form 16, 26AS, and AIS data were directly used for these calculations.

Acknowledgment

Acknowledgment is due to the open-source communities supporting FastAPI, SQLAlchemy, pandas, pdfplumber, and fpdf2. Without the contributions made by these libraries, the development and implementation of the system would not have been possible.

References

Please number citations consecutively within brackets [1]. The sentence punctuation follows the bracket [2]. Refer simply to the reference number, as in [3]—do not use “Ref. [3]” or “reference [3]” except at the beginning of a sentence: “Reference [3] was the first . . .”

References

[1] N. Tan, S. Lim, and R. Patel, “Automated transaction classification in personal finance applications using supervised learning,” in *Proc. IEEE Int. Conf. Data Engineering*, 2021, pp. 112–119.

[2] A. Sharma and P. Gupta, “Document understanding for tax forms: A structured extraction approach,” in *Proc. Int. Conf. Document Analysis and Recognition (ICDAR)*, 2022, pp. 340–347.

[3] M. Chen, Y. Liu, and J. Zhang, “Personal expense tracking via automated bank statement parsing,” *IEEE Trans. Consumer Electronics*, vol. 68, no. 2, pp. 145–153, May 2022.

[4] R. Kumar and S. Singh, “Bank transaction categorization using natural language processing and ensemble classifiers,” in *Proc. IEEE Int. Conf. Artificial Intelligence and Machine Learning*, 2023, pp. 78–85.

[5] L. Wang, F. Zhou, and T. Li, “Recurrence-aware financial transaction labeling for personal budget systems,” *IEEE Access*, vol. 11, pp. 33210–33221, 2023.

[6] V. Iyer and M. Nair, “Break-even analysis for old versus new income tax regime selection in India,” *Int. J. Finance and Economics*, vol. 15, no. 3, pp. 210–219, 2023.

[7] S. Das and A. Chatterjee, “Knowledge-base augmented conversational agents for tax advisory: Design and user evaluation,” in *Proc. ACM Int. Conf. Information and Knowledge Management (CIKM)*, 2022, pp. 2901–2910.

[8] W. McKinney, “Data structures for statistical computing in Python,” in *Proc. 9th Python in Science Conf.*, 2010, pp. 56–61.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)