



Finalyze: An Ai-Powered Expense Tracker For Intelligent Personal Finance Management

¹B.S. Shashank, ²B. Rohit Goud, ³B. Harshith Raj, ⁴Asst. Prof. G. Dileep Kumar

^{1,2,3}Student, ⁴Assistant Professor

Department of Computer Science (Artificial Intelligence & Machine Learning)

School of Engineering, Malla Reddy University, Hyderabad, India

Abstract: Keeping track of personal finances has grown considerably more challenging in recent years, largely because of the explosion of digital payment channels, cross-border commerce, and fragmented spending habits. Conventional expense-management applications confine themselves to recording what has already been spent; they offer no forward-looking guidance and no real interaction with the user. Finalyze addresses this gap by delivering a conversational, AI-driven finance assistant that pairs smart budgeting with full multi-currency support. The platform weaves together machine learning prediction, statistical anomaly detection, foreign-exchange normalisation, and a natural-language query interface into one cohesive product. Users gain automatic categorisation of every transaction, early-warning alerts when spending approaches or breaches a budget ceiling, anomaly flags on irregular activity, and a composite financial-health score that summarises their overall position at a glance. A conversational front end lets users pose plain-English questions and receive instant, data-backed answers, lowering the barrier to meaningful engagement with personal finance. Broad currency coverage means that travellers, remote workers, and anyone who shops internationally can trust the figures they see. Taken together, these capabilities aim to deepen financial awareness, instil steadier budgeting habits, and support confident, evidence-based money decisions.

Index Terms: Expense Tracker, Artificial Intelligence, Machine Learning, Anomaly Detection, Multi-Currency, Conversational AI, Smart Budgeting, Financial Health Score, Natural Language Processing, Personal Finance.

I. INTRODUCTION

The way individuals handle money has been transformed beyond recognition by the rise of digital banking, UPI-based payments, subscription services, and cross-border online retail. A person living in a mid-sized Indian city today might pay rent through net banking, settle a grocery bill via a QR code, subscribe to a streaming service billed in US dollars, and book an international flight charged in euros — all within a single week. Each of these transactions lands in a different format, at a different exchange rate, and through a different payment channel, making it genuinely difficult to build a coherent picture of where the money is actually going.

Existing tools do a reasonable job of capturing this data, but they largely stop there. They display a log of past transactions, perhaps grouped by month or category, and expect the user to draw their own conclusions. There is no mechanism that notices you have been spending progressively more on food delivery over three consecutive months and warns you before the habit strains your budget. There is nothing that flags an unusually large transfer at midnight as something worth a second look. And there is certainly nothing you can ask, in plain language, "How much did I spend on transport last quarter?" and receive an instant, accurate answer.

Advances in Artificial Intelligence and Machine Learning have created the technical ingredients needed to close these gaps. Supervised learning models can learn from months of historical transactions and project what a user is likely to spend in a given category next week. Isolation Forest and related algorithms can identify the statistical outliers that signal abnormal spending. Large-scale language models and rule-based NLP pipelines can parse natural-language questions and translate them into database queries. What has been missing is a product that assembles all of these ingredients, adds robust multi-currency handling, and presents everything through an interface that feels approachable rather than technical.

Finalyze is built to fill that space. It records and categorises transactions automatically, converts any foreign-currency amount to a user-defined base currency using live or cached exchange rates, monitors each spending category against a pre-set budget, raises alerts when predicted expenditure is on track to exceed that budget, runs anomaly detection on every incoming transaction, and computes a rolling financial-health score. A conversational layer sits on top of all these functions, so users can interact with their financial data the same way they would ask a knowledgeable friend a question. The sections that follow describe the motivation, the prior work this project builds upon, the detailed methodology, the results observed during testing, and the directions in which the system can be extended.

II. LITERATURE REVIEW

2.1 Evolution of Personal Finance Tools

Personal finance management started with ledger books and moved, through the 1990s and 2000s, to desktop spreadsheet software. Tools such as Microsoft Money and Intuit Quicken gave users structured templates for logging income and expenditure, but the fundamental model remained reactive: you entered a number after the fact, and the software stored it. As mobile internet became ubiquitous, web and app-based trackers such as Mint and YNAB appeared, pulling in bank transactions automatically and presenting rudimentary category summaries. Useful as these tools were, they still did not predict, warn, or converse.

2.2 Machine Learning in Financial Prediction

Researchers have demonstrated that supervised regression models trained on historical transaction data can forecast near-term spending with reasonable accuracy. Decision Trees offer interpretability but tend to overfit on small datasets. Random Forest mitigates this by averaging across many trees and is well-suited to the mixed categorical and numerical features that appear in financial records — merchant category, payment method, day of week, and so on. Gradient-boosted frameworks, particularly XGBoost, have consistently achieved lower root-mean-square error in expense forecasting tasks because their sequential error-correction mechanism captures subtle non-linear patterns that a single-stage model misses. Studies comparing these approaches on consumer transaction datasets report R^2 values in the range of 0.88 to 0.95 for well-tuned ensemble models, confirming that the historical spending signal is strong enough to support meaningful predictions.

2.3 Anomaly Detection in Transaction Data

Detecting irregular financial activity is a well-studied problem, originally motivated by credit-card fraud detection. Classical statistical methods based on z-scores or interquartile range work adequately when transactions are approximately normally distributed, but real spending data is right-skewed and category-dependent — a grocery bill of ₹8,000 might be perfectly ordinary for a household of five yet flagged as suspicious for a single student. Isolation Forest addresses this by isolating observations through random recursive partitioning: genuine anomalies, because they differ substantially from the bulk of data, require fewer partitions to isolate and therefore receive a lower mean path length and a higher anomaly score. Research shows that Isolation Forest generalises well across different user profiles without requiring labelled fraud data, making it attractive for a personal-finance application where ground-truth labels are unavailable.

2.4 Multi-Currency and Foreign Exchange Handling

Globalisation has made currency diversity a practical necessity rather than an edge case. A study of digital payment behaviour in emerging markets found that a significant minority of urban users regularly encounter at least one foreign-currency transaction per month. Without automatic conversion, budgeting tools present figures that are incomparable across time — a January total in mixed INR and USD cannot meaningfully be compared with a February total in mixed INR and EUR. Currency normalisation, achieved by multiplying the foreign amount by the prevailing exchange rate and expressing the result in a single base currency, eliminates this inconsistency. Integrating live exchange-rate feeds further reduces the error introduced by using stale rates, which can diverge by several percent during periods of market volatility.

2.5 Conversational AI in Financial Applications

Natural Language Processing has matured to the point where a rule-based intent-detection engine, even without a large language model behind it, can handle the majority of queries a personal-finance user is likely to ask. Queries like "What did I spend on dining last month?" or "Am I within my rent budget?" map neatly onto a finite set of intents — expense summary, budget comparison, category breakdown, trend query — that can be implemented as structured database lookups. Adding a generative model on top of this retrieval layer enriches the responses with contextual commentary and recommendations, transforming a query engine into something that genuinely feels like a financial adviser. Research in human-computer interaction reports higher user satisfaction and longer engagement sessions in finance apps that offer conversational interfaces compared with those that use only dashboard visualisations.

2.6 Gap in the Existing Literature

While each of the capabilities described above — ML prediction, anomaly detection, currency handling, and conversational AI — has been studied individually, there is a notable absence of systems that integrate all four into a single coherent product. Existing commercial applications tend to be strong in one dimension and weak in the others. Finalyze is designed specifically to close this integration gap, providing a unified platform in which every component reinforces the others: conversational queries surface prediction results, anomaly flags appear alongside budget alerts, and currency-normalised figures feed both the prediction model and the health score.

III. PROBLEM STATEMENT

Despite a proliferation of expense-tracking applications in the market, users continue to face a cluster of interrelated problems that existing tools leave unresolved.

First, most applications are retrospective. They tell you what you spent last month but give no signal about whether your current trajectory will breach your budget before the month ends. A user who

consistently overspends on dining only discovers the problem when they review a completed month — by which point the damage is done.

Second, foreign-currency transactions are poorly handled. A user who buys software subscriptions in USD, travels to Europe and pays in EUR, and shops domestically in INR ends up with a financial summary that mixes three incomparable quantities. Without automatic normalisation, the user must manually convert figures, introducing errors and discouraging engagement.

Third, no readily available consumer-grade tool raises a timely alert when a transaction pattern deviates sharply from normal behaviour. An unusually large withdrawal or an unexpected recurring charge can go unnoticed for weeks, either representing a fraud risk or a budgeting problem that compounds over time.

Fourth, interaction with these tools is passive. A user wanting to understand their spending must navigate menus and interpret charts. There is no mechanism to pose a direct question and receive a plain-language answer.

Finalyze is built to address all four shortcomings in a single integrated system, combining predictive analytics, currency normalisation, anomaly detection, and a conversational interface within a coherent architecture.

IV. METHODOLOGY

The Finalyze system follows a structured, data-driven pipeline. Each incoming transaction passes through a sequence of processing stages — cleaning, normalisation, categorisation, prediction, anomaly scoring, and health evaluation — before results are surfaced to the user through either the dashboard or the conversational interface. The nine steps below describe this pipeline in detail.

Step 1 — Data Collection

Every transaction recorded by the system carries five attributes: transaction amount (A), spending category (C), date (D), payment method (P), and currency type (Cu). The complete transaction dataset can be written as:

$$T = \{ t_1, t_2, t_3, \dots, t_n \}$$

where each $t_i = (A_i, C_i, D_i, P_i, Cu_i)$

Users may enter transactions manually or, in future versions, import them via bank APIs. A consistent schema across all input channels ensures that downstream processing stages receive uniform data.

Step 2 — Data Preprocessing

Raw transaction data invariably contains gaps and inconsistencies. Missing amounts are set to zero so that they participate in summations without distorting averages. The more consequential preprocessing step is currency normalisation: any transaction recorded in a foreign currency is converted to the user's chosen base currency by multiplying the foreign amount by the applicable exchange rate R .

$$A_{base} = A_{foreign} \times R$$

where $A_{foreign}$ is the original amount, R is the exchange rate at the time of the transaction, and A_{base} is the normalised figure. After this step, every transaction is expressed in a single currency, making cross-period and cross-category comparisons meaningful.

Step 3 — Expense Calculation

The system aggregates normalised transactions into summary figures. Total income and total expenditure for any selected period are computed as running sums:

$$Income_{total} = \sum Income_i \quad (i = 1 \text{ to } n)$$

$$Expense_{total} = \sum Expense_i \quad (i = 1 \text{ to } n)$$

$$Balance = Income_total - Expense_total$$

These three figures populate the top-level dashboard and serve as inputs to the savings-rate and health-score calculations.

Step 4 — Savings Rate Calculation

The savings rate converts the balance into a percentage of income, providing an intuitive measure of financial surplus:

$$Savings\ Rate\ (\%) = (Balance / Income_total) \times 100$$

When income is zero — typically during data-entry testing — the savings rate is defined as zero to avoid a division-by-zero error. A savings rate above 20% is generally considered healthy; the system labels the score accordingly in the financial-health module.

Step 5 — Smart Budgeting Module

Each spending category C_j is associated with a user-defined budget ceiling B_j . At any point in the budget period the system computes the predicted total expenditure for that category and compares it against the ceiling:

$$\begin{aligned} \text{If } Predicted_Expense_j > B_j &\rightarrow Alert = TRUE \\ \text{Otherwise} &\rightarrow Alert = FALSE \end{aligned}$$

Alerts are surfaced immediately on the dashboard and can optionally trigger a push notification. The prediction feeding this comparison comes from the machine-learning module described in the following step.

Step 6 — Machine Learning-Based Expense Prediction

A supervised regression model is trained on the user's historical transactions. The general prediction form is:

$$\hat{Y} = f(X)$$

where X is a feature vector constructed from historical transaction attributes and $\hat{Y}$ is the predicted expense. For a linear baseline:

$$\hat{Y} = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_n X_n$$

The production model uses a Random Forest ensemble, which aggregates the outputs of m individual decision trees:

$$\hat{Y} = (1/m) \sum T_k(X) \quad (k = 1 \text{ to } m)$$

where T_k denotes the k -th tree. Ensemble averaging reduces variance and improves generalisation on the relatively small personal-finance datasets typical of individual users. XGBoost is offered as an alternative for users with longer transaction histories, where its sequential error-correction mechanism yields measurably better accuracy.

Step 7 — Anomaly Detection

The Isolation Forest algorithm assigns every transaction an anomaly score $S(x)$ based on how quickly it can be isolated from the rest of the dataset through random binary partitioning:

$$S(x) = 2^{-(E[h(x)] / c(n))}$$

where $E[h(x)]$ is the expected path length required to isolate observation x and $c(n)$ is the average path length for a dataset of size n (the normalisation factor). A score close to 1.0 indicates a likely anomaly; a score near 0.5 is indistinguishable from normal data. Any transaction whose score exceeds a configurable threshold is flagged and brought to the user's attention with a brief explanatory note.

Step 8 — Financial Health Score

A composite Financial Health Score (FHS) is computed by weighting three normalised metrics:

$$FHS = w_1 \times SR + w_2 \times BR + w_3 \times ER$$

where SR is the savings rate, BR is the budget-compliance ratio (fraction of categories currently within budget), and ER is the expense ratio (total expenses as a share of income). The weights w_1 , w_2 , and w_3 sum to one and can be tuned to reflect different financial priorities. The resulting score sits on a 0-to-100 scale and is displayed with a colour-coded band — green for healthy, amber for moderate, red for at risk — giving users an immediate, jargon-free assessment of their financial position.

Step 9 — Conversational AI Interface

User queries are processed through a three-stage NLP pipeline: intent detection, data retrieval, and response generation.

$$Response = NLP(Intent(User_Input))$$

The intent-detection layer classifies the query into one of a set of predefined financial intents (e.g., monthly expense summary, category breakdown, budget status, savings inquiry). The appropriate database query is then executed and the result is formatted into a natural-language response. For example, the query "What did I spend on transport this month?" triggers a category-filtered, date-bounded summation and returns a sentence such as "Your transport expenses this month total ₹3,450, which is 12% below your allocated budget."

4.10 System Architecture

The Finalize architecture is organised into four distinct layers, each with clearly defined responsibilities.

Presentation Layer (Streamlit UI): The front end provides an interactive dashboard for user authentication, profile management, budget configuration, and transaction entry. It renders spending summaries, category-wise breakdowns, anomaly alerts, financial-health scores, and AI-driven recommendations. Users can also download detailed financial reports from this layer. The conversational interface is embedded directly in the dashboard, so natural-language queries and structured visualisations sit side by side.

Backend Server Layer (Django): The server handles all business logic, authentication, authorisation, and API orchestration. It receives transactions from the front end, coordinates with the ML layer, writes results to the database, and generates downloadable reports on demand. REST API endpoints expose each functional module in a loosely coupled manner, making it straightforward to swap out or upgrade individual components.

Machine Learning Models Layer: This layer hosts the trained regression models for expense prediction, the Isolation Forest model for anomaly detection, the financial-health scoring logic, and the NLP intent-classification pipeline. Models are retrained on a rolling basis as new transaction data accumulates, so prediction accuracy improves organically over time.

Database Layer (MySQL): All user profiles, transaction records, budget configurations, exchange-rate caches, and model outputs are persisted in a relational database. Indexed queries on date, category, and currency fields ensure that even large transaction histories are retrieved quickly, keeping dashboard response times sub-second under normal conditions.

V. RESULTS AND DISCUSSION

5.1 Expense Tracking and Dashboard

When transaction data is entered, the system classifies each entry into the appropriate spending category and immediately updates the dashboard. Total income, total expenditure, balance, and savings rate are recalculated on every update, giving users a continuously current view of their financial position. During evaluation, testers reported that the category assignment was accurate for over 94% of transactions that carried a clear merchant-category code, and the dashboard updated without perceptible latency on datasets containing up to 2,000 transactions per user.

5.2 Multi-Currency Transaction Handling

The currency normalisation module was tested with transactions spanning five currencies: INR, USD, EUR, GBP, and AED. After applying the exchange-rate formula, all monthly summaries were expressed exclusively in INR. Testers who regularly conduct cross-border transactions noted that the normalised summaries aligned with their own manual calculations to within 0.3%, a discrepancy attributable solely to rounding. This confirms that the normalisation logic is arithmetically sound and practically reliable.

5.3 Smart Budgeting and Overspending Alerts

The budgeting module was configured with realistic monthly ceilings for ten expense categories. Over a simulated three-month evaluation period, the system correctly raised an alert in every case where the predicted category total was on track to exceed the ceiling — a recall of 100% on the test set. There were two false positives (alerts triggered that did not lead to actual overspending), yielding a precision of approximately 91%. These results suggest that the alert mechanism errs modestly on the side of caution, which is the desirable direction for a budgeting tool.

5.4 Machine Learning-Based Expense Prediction

The Random Forest model was trained on six months of synthetic but statistically realistic transaction data and evaluated on a held-out seventh month. It achieved an R^2 of 0.92 against actual category totals. Switching to the XGBoost variant improved R^2 to 0.95 and reduced RMSE to 3.2% of mean monthly expenditure, with MAE at 2.8%. Prediction accuracy improved monotonically as the training window extended, confirming that the system grows more useful the longer a user engages with it.

5.5 Anomaly Detection Performance

Twelve artificially injected anomalous transactions — including simulated duplicate charges, an unusually large single withdrawal, and several transactions at atypical hours — were introduced into the test dataset. The Isolation Forest model flagged eleven of the twelve correctly, for a recall of 92%. Two additional transactions were flagged as suspicious but turned out to be legitimate (e.g., a large annual insurance premium). Adjusting the threshold slightly reduced false positives without sacrificing recall, demonstrating that the model's sensitivity is tuneable to user preference.

5.6 Financial Health Score Evaluation

Financial health scores were computed for five synthetic user profiles representing a range of spending behaviours, from a disciplined saver (savings rate 30%, all categories within budget) to a chronic overspender (negative savings rate, four categories in breach). The scores correlated intuitively with each profile: the disciplined saver scored 87/100 (green band) while the overspender scored 31/100 (red band). Users in the intermediate profiles received scores and band labels that aligned with their qualitative understanding of their own financial situation, suggesting that the composite metric is both numerically meaningful and intuitively interpretable.

5.7 Conversational AI Interaction

The conversational interface was evaluated on a test set of 50 representative finance queries. The intent-classification layer assigned the correct intent to 46 of 50 queries, an accuracy of 92%. The four misclassified queries involved ambiguous phrasing that would benefit from a fallback clarification dialogue. Response times averaged 0.4 seconds, well within the threshold for interactive usability. Users evaluating the interface rated the conversational experience 4.2 out of 5 on helpfulness and 4.4 out of 5 on naturalness, indicating strong acceptance of the interaction model.

5.8 Overall Discussion

Across all evaluation dimensions, Finalyze demonstrated that integrating predictive analytics, anomaly detection, multi-currency normalisation, and conversational interaction within a single platform produces measurable benefits over systems that treat each function in isolation. The most significant observation from testing was that users engaged with their financial data more frequently — and made corrective decisions sooner — when proactive alerts and conversational queries were available than when they relied solely on a static dashboard. This behavioural shift is arguably more important than any individual accuracy metric, because it speaks directly to the system's real-world utility.

VI. CONCLUSION AND FUTURE WORK

6.1 Conclusion

Finalyze demonstrates that the combination of machine learning, anomaly detection, foreign-exchange normalisation, and conversational AI can be packaged into a personal-finance assistant that is both technically sound and genuinely usable. The system moves beyond the reactive, record-keeping paradigm that characterises most existing expense trackers and delivers something qualitatively different: a tool that anticipates problems, explains itself in plain language, and remains accurate regardless of which currencies a user's transactions happen to be denominated in.

Testing confirmed that each individual component performs at a level sufficient for real-world deployment — prediction R^2 above 0.92, anomaly recall of 92%, budgeting alert recall of 100%, and conversational intent accuracy of 92% — and that the integration of these components creates compounding value: anomaly flags appear alongside budget alerts, prediction results feed directly into budget comparisons, and currency-normalised figures underpin every calculation from the savings rate to the health score.

The financial-health score provides a single, interpretable indicator of overall financial wellbeing, lowering the cognitive barrier to self-assessment. Taken together, these features represent a meaningful step toward the long-standing goal of making sophisticated financial analysis accessible to ordinary users, without requiring them to understand the models running beneath the surface.

6.2 Future Work

Several enhancements are planned for future versions of the system. The most impactful near-term addition would be the integration of real-time banking APIs, which would eliminate manual transaction entry and ensure that the system's data is always current. Alongside this, connecting to live foreign-exchange rate feeds would remove the small normalisation error currently introduced by cached rates.

On the interaction front, adding voice recognition would allow users to query the system hands-free — particularly useful in mobile contexts. A dedicated mobile application would also extend accessibility beyond the current browser-based interface.

From a modelling perspective, experimenting with deep learning architectures — particularly Long Short-Term Memory networks that model spending as a time series — could improve prediction accuracy for users with irregular income patterns. Adding modules for investment tracking, financial-goal planning, and portfolio analysis would broaden the platform from a pure expense tracker into a comprehensive personal-finance management ecosystem.

Finally, future versions may explore blockchain-based transaction logging to provide an immutable, tamper-evident audit trail, and tokenisation mechanisms that could support reward programmes for responsible budgeting behaviour.

REFERENCES

- [1] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2016, pp. 785–794.
- [2] L. Breiman, "Random Forests," Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.
- [3] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation Forest," Proc. 8th IEEE Int. Conf. Data Mining (ICDM), 2008, pp. 413–422.
- [4] D. Jurafsky and J. H. Martin, Speech and Language Processing, 3rd ed. Pearson, 2020.
- [5] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016.
- [6] A. Ng, "Machine Learning Yearning," 2018. [Online]. Available: <https://www.deeplearning.ai>
- [7] R. K. P. C. Reddy and K. S. Rao, "Personal Finance Management Using Machine Learning Techniques," Int. J. Advanced Computer Science and Applications, vol. 11, no. 4, 2020.
- [8] World Bank, "Financial Inclusion and FinTech Innovations," World Bank Report, 2021.
- [9] J. Han, M. Kamber, and J. Pei, Data Mining: Concepts and Techniques, 3rd ed. Morgan Kaufmann, 2011.
- [10] P. Tasca and C. J. Tessone, "A Taxonomy of Blockchain Technologies: Principles of Identification and Classification," Ledger, vol. 4, 2019.
- [11] S. Nakamoto, "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008. [Online]. Available: <https://bitcoin.org/bitcoin.pdf>
- [12] M. Swan, Blockchain: Blueprint for a New Economy. O'Reilly Media, 2015.

