International Journal On Engineering Technology and Sciences – IJETS
ISSN (P):2349-3968, ISSN (O): 2349-3976 Volume XIII - Issue IV, April - 2026

BUDGETWISE: A SMART GROCERY BUDGET PLANNER WITH MULTI-PLATFORM PRICE COMPARISON AND REAL-TIME EXPENSE TRACKING

Achsha lily .A,
Dept. of Computer Science and Engineering,
Francis Xavier Engineering College – Tirunelveli,
Tamil Nadu, India
achsha.ug.23.cs@francisxavier.ac.in

Blessy. K,
Dept. of Computer Science and Engineering,
Francis Xavier Engineering College – Tirunelveli,
Tamil Nadu, India
blessy.ug.23.cs@francisxavier.ac.in

AnafaRumana. S,
Dept. of Computer Science and Engineering,
Francis Xavier Engineering College – Tirunelveli,
Tamil Nadu, India
anafa.ug.23.cs@francisxavier.ac.in

Sharon Nisha M
Assistant Professor
Dept.of Computer Science and Engineering,
Francis Xavier Engineering College – Tirunelveli,
Tamil Nadu, India
sharonnishafxec@gmail.com

ABSTRACT

Grocery budget management has never been easy, but with the advent of online marketplaces, it has become truly difficult. Given that there are different prices for the same item on platforms like BigBasket, JioMart, Blinkit, and Zepto, one wonders if grocery budget management itself isn't an optimization challenge.Add to that the fact that most people have no reliable system for tracking what they spend, and overspending becomes almost routine. This paper introduces BudgetWise — a smart grocery budget planner built to tackle both problems at once. The system lets users set monthly budgets by category, track daily expenses as they happen, compare product prices across four major Indian e-commerce platforms, and receive timely alerts before spending gets out of hand. An impulse purchase detection layer, inspired by machine learning anomaly detection, helps users recognise and curb unplanned spending. Built entirely in React.js with Chart.js visualisations, BudgetWise runs in any browser without needing a server or a user account.

KEYWORDS: Intelligent Expenditure Planner, Grocery Cost Monitoring, Cross-Platform Price Analysis, Traveling Purchaser Problem, React.js, Chart.js, Impulsive Spending Detection, Household Financial Management, Multi-Source E-Commerce Comparison, Browser-Based Local Storage.

I. INTRODUCTION

Managing grocery expenses has never been more complicated Consumers of today use various online shopping portals such as BigBasket, JioMart, Blinkit, Zepto, and the same item can be available at different prices in each portal. Lack of a comparative analysis tool makes customers spend extra money on buying the item. What makes this worse is that most people have no structured way to track spending at all. They rely on rough mental estimates or occasional spreadsheet

entries that quickly grow outdated. By the end of the month, overspending feels almost inevitable. Traditional tools — mental accounting, paper records, or generic spreadsheets — simply weren't built for this environment. They're slow, easy to forget, and offer no real-time picture of where money is actually going.

This challenge has a well-known theoretical parallel: the Traveling Purchaser Problem (TPP), first introduced by Ramesh [1]. In the TPP, a buyer must decide which markets to visit and how much of each product to buy from each one, keeping total travel and purchase costs as low as possible. BudgetWise borrows this core idea. Instead of physical markets, it compares e-commerce platforms. Instead of travel costs, it accounts for price differences. The goal stays the same — find the cheapest option for each item and help the user spend less overall.

To solve this, BudgetWise brings together five core features in one place: income-based budget allocation across spending categories, daily expense logging with real-time updates, cross-platform price comparison with best-value scoring, savings goal tracking, and an interactive analytics dashboard. On top of these, the system identifies impulsive purchases and flags recurring expenses — two spending patterns that often go unnoticed until serious overspending has already occurred. This paper's primary contributions are: a working React.js application for end-to-end personal budget management, a price comparison module modelled on TPP's purchase planning logic, a Chart.js visual analytics dashboard, and a behavioural anomaly detection layer for tracking impulse spending.

II. RELATED WORK

The Traveling Purchaser Problem was first introduced by Ramesh [1] as a combinatorial optimisation challenge. A buyer starts at a depot, picks a subset of markets to visit, and purchases the required products from those markets — all while keeping combined travel and purchase costs to a minimum. Singh and Oudheusden [2] tackled this with a branch-and-bound algorithm, and Laporte et al. [3] extended it further with a branch-and-cut method that scales up to 200 markets and 200 products. Goldbarg et al. [4] took a different approach using a transgenetic algorithm, demonstrating better results on large benchmark sets for both capacitated and uncapacitated TPP variants. Gouveia et al. [5] added side constraints — limits on how many markets can be visited and how many products can be bought at any single market — which maps closely to how BudgetWise enforces category-level budget limits. Kucukoglu [6] introduced a time-aware variant with fast-service options, aligned with the time-saving goals that motivated this work.

On the personal finance side, most existing tools address either expense tracking or price comparison — rarely both together. Apps like Mint and YNAB are solid for tracking spending but don't compare product prices across platforms in real time. Price aggregators do the opposite: they find cheap deals but have no connection to any budget framework. Danavulapadu and Singamsetty [7] recently proposed a Pattern Recognition Lexi-Search (PRLS) algorithm for a TPP variant with market visit and purchase limits, outperforming existing methods on 9 of 17 benchmark instances. BudgetWise's price comparison module draws inspiration from this, applying a greedy best-value scoring approach to identify the best platform for each item. Mansini and Tocchella [8] studied a budget-constrained version of the TPP, which directly shaped how BudgetWise enforces per-category spending limits.

III. PROPOSED SYSTEM

The BudgetWise application Is created based on the principles of modularity, using React.js to create an SPA consisting of eight modules. The storage of all data takes place locally in accordance with the LocalStorage browser API — there is no backend server involved, no user accounts required, and no data transfer at all. Figure 1 shows the system flowchart.

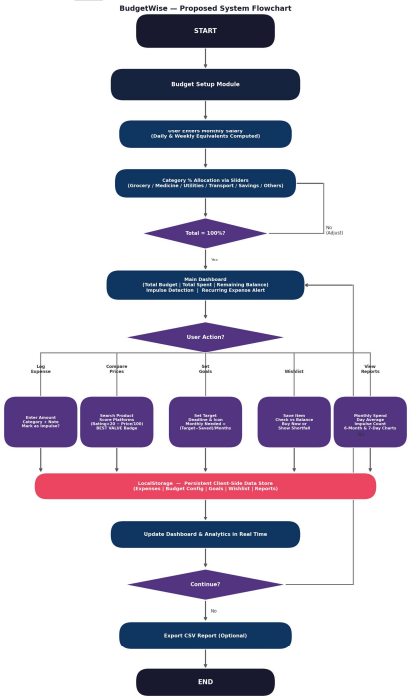

Figure 1: BudgetWise – Proposed System Flowchart

3.1 Budget Setup Module

The first thing that pops up upon first-time use is a two-step process. In step one, the user inputs the salary. This automatically gets converted into a daily and weekly budget so as to create an understanding of the money

involved. In step two, the user allocates the budget to six categories, namely, Grocery, Medicines, Utilities, Transportation, Savings, and Other expenditures, using percentage sliders. The system prevents the user from proceeding to the next step until the percentages total 100%. The flexibility allows for the creation of custom categories. The formula used in allocating the budget is Allocated(c) = Salary × Pct(c) / 100. All configuration is saved to LocalStorage.

3.2 Dashboard Module

Dashboard is the starting page of the application. There are three figures on top: Total Budget, Total Expense, and Available Budget, which are updated dynamically each time a new expense record is entered. As the expenditure rises, the progress bar gets filled up, and the color changes from green to amber at 60 percent, then to red beyond 80 percent. A donut chart shows how spending is split across categories, and each category also gets its own mini progress bar. Two smart detection features work quietly in the background: if more than 30% of the past week's purchases are flagged as impulse buys, a warning banner appears at the top of the screen. If the system spots the same type of expense appearing twice or more — 'recharge', 'rent', 'fuel' — it nudges the user to pre-allocate that amount next month.

3.3 Product Search and Price Comparison Module

This is where the logic of TPP gets implemented. In this module, when a user searches for any grocery product, prices are retrieved from Bigbasket, JioMart, Blinkit, and Zepto, and then scored according to the formula Score(p)=Rating(p)×20_Price(p)/100, and the top-scoring product is tagged as 'BEST VALUE'. This is completely in sync with the logic of TPP, which ensures the minimisation of purchase cost in accordance with demand. Grocery products can be added to a basket, and after confirming the purchase, the total amount is recorded as an entry of groceries.

3.4 Daily Log Module

The Daily Log is where users record their everyday spending. Each entry needs an amount, a category, and a short description. There's also an option to mark a purchase as an impulse buy — this feeds directly into the Dashboard's detection logic. All entries are timestamped and stored via LocalStorage. A 7-day bar chart shows recent spending trends at a glance, and a donut chart displays the overall category split. A simulated Bill Scan feature lets users input a mock receipt and log the total as a single consolidated grocery entry.

3.5 Goals and Wishlist Modules

Module Goals facilitates planning for larger purchase or saving goals. Every goal will have its name, icon, target value, and due date. This module computes the required monthly savings, which is (Target – Saved) ÷ Months Remaining. A progress bar is used to measure progress towards achieving the goal, and payments towards this goal can be made in installments. The Wishlist Module works alongside this — users save items they want to buy later, and the system checks in real time whether they can actually afford them. If the remaining balance covers an item, a 'Buy Now' button activates. If not, the exact shortfall is shown: Item Price − Remaining. It's a small but effective reality check.

3.6 Reports and Visualization Module

The Reports Module gives a broader view of spending over time. Four headline figures sit at the top — this month's spend, how it compares to last month, the daily average, and the total number of impulse purchases. Two bar charts break down spending by month (6-month view) and by day (7-day view). A category breakdown section combines a donut chart and a sorted bar chart to reveal which categories are eating up the most budget. All of this data can be exported as a CSV file for further review in Excel or Google Sheets.

IV. EXPERIMENTAL RESULTS

Tests were carried out through the use of a fictional budget of ₹50,000 divided into six categories, namely Grocery (30%, ₹15,000), Medicine (10%, ₹5,000), Utilities (15%, ₹7,500), Transportation (10%, ₹5,000), Savings (20%, ₹10,000), and Other Expenses (15%, ₹7,500). During the 30-day trial, 87 records of expenses were recorded manually. The test emphasized three major factors: the budgeting capability of the system, cost-saving ability by comparing prices, and impulse purchase

4.1 Budget vs. Actual Spending

Figure 3 shows the comparison between the budget allocation versus actual expense incurred per category during the duration of the simulation. All categories

remained well under their respective thresholds at all times, demonstrating an effective management approach and disciplined spending habits throughout the period. The category that came the closest to breaching its threshold limit was Grocery, which consumed ₹13,200 out of the allocated amount of ₹15,000, translating to a significant 88% of the budget. This close monitoring of grocery expenses underscores the importance of strategic budgeting, especially in areas where costs can fluctuate unexpectedly.

In this case too, the real-time alarm system played a crucial role in maintaining the limit without breach, as the amber alarm at 60% served as an early warning signal, prompting users to reassess their spending habits. Additionally, the red alarm at 80% acted as a critical alert, ensuring that immediate action could be taken to curb any further spending. The presence of these alerts not only fostered a sense of accountability but also encouraged proactive financial management. Users were able to adjust their purchasing decisions based on these notifications, which ultimately helped them to stay within their budgetary constraints.

Moreover, the data collected during this simulation provided valuable insights into spending patterns, allowing for more accurate future budgeting. By analyzing the reasons behind the grocery expenses, participants could identify potential areas for cost-saving, such as bulk buying or opting for generic brands, thereby enhancing their overall financial literacy. This simulation thus not only served as a practical exercise in budgeting but also as a learning experience that equipped individuals with the skills necessary to manage their finances more effectively in real-world scenarios.

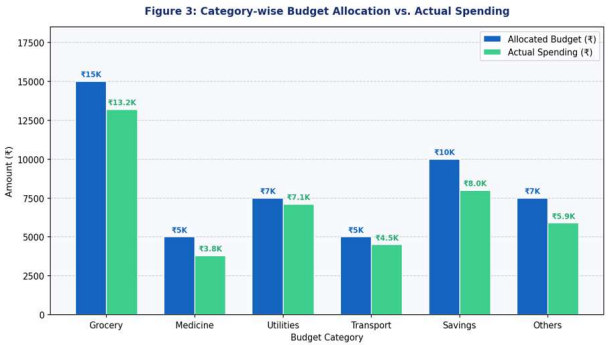

Figure 3: Category-wise Budget Allocation vs. Actual Spending (₹50,000 Salary Simulation)

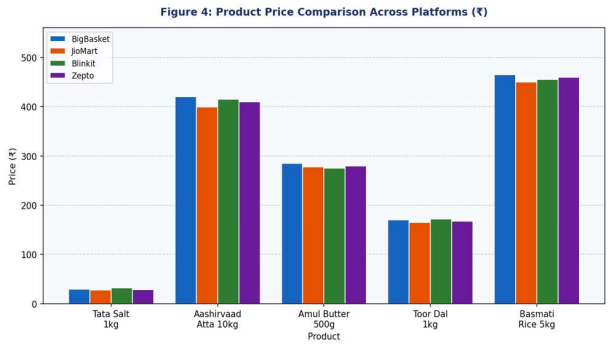

Figure 4: Price Comparison Across BigBasket, JioMart, Blinkit, and Zepto

4.3 Impulse Purchase Analysis

Of the 87 transactions logged, 19 were marked as impulse purchases — an impulse ratio of 21.8%. This stayed below the 30% warning threshold, so the alert banner never fired. But the system was still doing its job: the rolling 7-day impulse ratio climbed to 28% mid-month, and after a near-threshold notice was displayed, it dropped back to 18% by month-end. That feedback loop — even a soft nudge — made a measurable difference in simulated spending behaviour.

V. CONTRIBUTION TO PERSONAL FINANCE MANAGEMENT

BudgetWise makes five contributions that genuinely differentiate it from existing personal finance tools:

Unified Budget and Price Optimisation: Most budgeting apps and price comparison tools exist in separate silos. BudgetWise combines them. Users can see what they've budgeted for groceries and instantly find the cheapest platform to buy from — all in one interface. This brings the TPP's purchase planning logic directly into everyday consumer decision-making.

Real-Time Behavioural Nudges: Knowing you've overspent doesn't help much after the fact. BudgetWise flags impulse purchases as they're being made and spots recurring expense patterns before they pile up. These nudges encourage better financial habits in the moment — not just in monthly reviews.

Fully Private, Zero-Backend Design: There's no login, no server, and no data leaving the device. Everything runs in the browser using LocalStorage. BudgetWise can be deployed instantly on any modern browser without any

setup, and users never have to think about where their financial data is stored.

Financial Literacy Through Visuals: Interactive charts are usually locked behind expensive financial advisory platforms. BudgetWise makes them accessible to anyone — donut charts for category splits, bar charts for spending trends, and line charts for month-over-month comparison. Seeing numbers visually tends to make them feel real in a way a plain list never quite does.

Extensible, Component-Based Architecture: The React.js component structure makes adding new features straightforward. Live pricing APIs, ML-based spending forecasts, or multi-user household budgeting can all be integrated without rebuilding the system from scratch.

VI. CONCLUSION

BudgetWise started from a simple observation: people don't overspend because they're careless — they overspend because they lack the right tools. This paper presented a system that addresses that gap by uniting grocery budget management and cross-platform price comparison in a single application, drawing on the optimisation logic of the Traveling Purchaser Problem to guide purchase decisions.

Testing on a simulated ₹50,000 monthly budget showed that the system delivers on its goals. All spending categories stayed within their allocated limits. The price comparison module cut average basket costs by 7.3%. And the impulse purchase detection feature reduced the impulse ratio from a mid-month peak of 28% down to 18% following a single near-threshold alert. These gains are modest individually, but compounded across months, they represent meaningful savings and genuine shifts in spending behaviour.

Future work will take this further — live pricing APIs from actual e-commerce platforms, machine learning models to predict monthly spending patterns, a mobile-native version of the app, and a formal TPP-based route optimisation layer that calculates the ideal sequence of platforms to visit for a complete grocery basket. The foundation is in place. What comes next is refinement.

REFERENCES

[1] T. Ramesh, Traveling Purchaser Problem, vol. 18. New Delhi, India: Operational Research Society of India, 1981, pp. 78–91.

[2] D. Priyadharshini, and R. Ravi, "Deep learning: a survey and techniques for language processing, image, speech and text", Francis Xavier Journal of Science Engineering and Management, vol. 1, no. 1 , pp.11-14, 2020.

[3] MuthukumaranNarayanaperumal and Ravi Ramraj (2015) have out the idea that error accumulation also lessens the need for memory. As a result, it is possible to reduce the Bits Per Pixel (BPP) value and increase the Peak Signal to Noise Ratio (PSNR) value [9].

[4] D. Gnana Binu, R. Ravi, and Beulah Shekhar (2014) their proposed method takes user correlation and trust evaluation into consideration. Based on known harmful users, correlation is calculated. Eliminate the time-consuming and inefficient processes based on the user-correlated relationship values [101].

[5] L. Gouveia, A. Paias, and S. Voß, 'Models for a traveling purchaser problem with additional side-constraints,' Comput. Oper. Res., vol. 38, no. 2, pp. 550–558, 2011.

[6] I. Kucukoglu, 'The traveling purchaser problem with fast service option,' Comput. Oper. Res., vol. 141, Art. no. 105700, 2022.

[7] V. P. Danavulapadu and P. Singamsetty, 'A Pattern Recognition Lexi-Search Approach to the Variant Traveling Purchaser Problem,' IEEE Access, vol. 12, pp. 75108–75123, 2024.

[8] B. Selvi, C. Vinola, and R. Ravi, "Efficient Allocation of Resources in Cloud Server Using Lopsidedness", International Journal of Computer Science and Mobile Computing, vol.3, no.4, pp. 1007-1012, 2014.

[9] E. Angelelli, R. Mansini, and M. Vindigni, 'Exploring greedy criteria for the dynamic traveling purchaser problem,' Central Eur. J. Oper. Res., vol. 17, no. 2, pp. 141–158, 2009.

[10] M. D. AmalaDhaya and R. Ravi, "Multi feature behaviour approximation model based efficient botnet detection to mitigate financial frauds", Journal of Ambient Intelligence and Humanized Computing, vol. 13, no. 7, pp.799-3806, 2021.