HACKATHON PHASE 1

| | | |
|---|---|---|
| **COLLEGE CODE** | **:** | **001** |
| **COLLEGE NAME** | **:** | **AU-CEG** |
| **DEPARTMENT** | **:** | **B.E CSE** |
| **STUDENT NM ID** | **:** | **0F2EFDC0474697FA7984EE2A52C6028B** |
| **ROLL NO** | **:** | **2023103588** |
| **DATE** | **:** | **07-11-2025** |
| **TECHNOLOGY** | **:** | **Front End/Node JS** |
| **PROJECT NAME** | **:** | **EXPENSE TRACKER** |

**SUBMITTED BY:**

**NAMES:**

- ARUNA M – 9600703124

- SRI BAVAN AKASH S-9500284495

- PARIL T-9910788403
- SHARAN SAMINATHAN-8903766244(TL)
- HARSIKA V-8072619155

## Problem Statement

People struggle to understand where their money goes because manual spreadsheets are error-prone and slow. They need a simple, friendly way to record expenses, see trends, and stick to budgets.

Goal: Build a single-page Expense Tracker frontend that:

- Makes adding expenses fast and delightful (keyboard-first, mobile-friendly).
- Gives clear visual summaries of spending vs. budget.
- Encourages habits via nudges (e.g., “you’re 80% through your Dining budget”).

Scope: SPA demonstrating robust CRUD for transactions, categories, and budgets with great UX/accessibility and mock/real APIs.

## Users & Stakeholders

### Primary Users

- Students, families, solo professionals tracking daily expenses.
- Novice users who need guidance and accessible UI.

**Secondary Users / Stakeholders**

- Product owner/instructor (approves deliverables)
- UX designer (reviews wireframes & a11y)
- Frontend developers (implement/maintain)
- QA/testers (edge cases & regressions)
- Backend dev (optional APIs for auth/sync/import)

## User Stories

1. As a user, I want to quickly add an expense with amount, category, date, and notes.
2. As a user, I want category budgets and to see progress against them.
3. As a user, I want monthly/weekly summaries and charts of spending.
4. As a user, I want to filter/search transactions by date range, category, min/max amount, and text.
5. As a user, I want recurring transactions (e.g., rent) to auto-populate.
6. As a user, I want CSV import/export to migrate data.
7. As a user with assistive tech, I want an accessible UI (ARIA, keyboard nav, screen-reader announcements).
8. As a tester, I want a suite of calculation tests (totals, budgets, currency formatting).
9. As a developer, I want modular services/utils with docs for easy maintenance.

## MVP Features

- **Transactions CRUD**: amount, type (expense/income), date, category, account (optional), merchant/note.
- **Categories**: preset + custom; color/icon; soft delete; merge.
- **Budgets**: monthly per category (amount + period); progress bar with remaining.
- **Dashboards**: current month total, top categories, trend line.
- **Filters/Search**: date range, categories, text; saved filters.
- **Validation & UX**: inline feedback (positive numbers, required fields, date in range); undo toast for delete.
- **Accessibility**: aria-labels/roles, focus management for dialogs, contrast ≥ 4.5:1, keyboard shortcuts.
- **Responsive**: mobile-first; sticky add button on small screens.
- **Prevent destructive mistakes**: confirm modals; optimistic UI with rollback on API error.

**Additional Features:**

- Recurring rules (daily/weekly/monthly/custom).
- Multi-currency with base currency; FX rates mock.
- Attach receipts (mock upload).
- Reminders (budget threshold, bill due).
- Shared budgets (invite a collaborator).
- PWA offline support + local caching.
- Simple insights (“Spending 20% more than last month on Groceries”).

**Wireframes (Low-Fi Descriptions)**

1) Dashboard (Home)

- Header: month selector; KPIs (Total Spent, vs Budget, Income, Net).
- Cards: “Top Categories (pie)”, “Spending Over Time (line)”, “Budget Progress (list with bars)”.
- FAB/primary button: “Add Transaction”.

2) Transactions List

- Toolbar: filters (date range, category, type), search box, CSV Import/Export.
- Table/list rows: amount (right-aligned), category chip, date, merchant/note, edit/delete icons.
- Pagination or infinite scroll.

3) Add/Edit Transaction (Modal/Drawer)

- Fields: Type toggle, Amount, Date (picker), Category (select with search), Merchant, Note, Account.
- Inline validation + submit; success toast.

4) Budgets

- Grid/list of categories with monthly limit, spent, remaining; progress bars and color thresholds.

API Endpoint List (Mock/Optional)

- Auth (optional)
  - POST /api/login → { token }
- Transactions
  - GET /api/transactions?from&to&category&type&q&page
  - POST /api/transactions → create
  - PUT /api/transactions/:id → update
  - DELETE /api/transactions/:id
- Categories
  - GET /api/categories / POST /api/categories / PATCH /api/categories/:id
- Budgets
  - GET /api/budgets?month / POST /api/budgets

- Reports
    - GET /api/reports/summary?month → totals, top categories, trend

## Acceptance Criteria

- Adding/editing/deleting transactions updates **totals, budgets, and charts** immediately.
- Budget progress shows **spent, remaining, and %**; color changes at 80%/100%.
- Filters affect list and summary consistently; zero-state messages appear appropriately.
- Forms prevent submit on invalid input; error text is human-readable (“Enter a positive amount”).
- Keyboard-only users can: open Add dialog, tab through fields in order, save, and see focus return.
- Screen readers announce dialog open/close and validation errors via aria-live="polite" or role="alert".
- Responsive layout works ≥ 320px width; charts remain readable.
- Unit test coverage ≥ **80%** for calculation utilities (totals, grouping, budget math) and formatters.
- Performance: initial bundle ≤ **200KB gzip** (target), interactions < **100ms**.