



# BudgetBox

**Assignment Name:** BudgetBox << Please mention this name in Submission Form >>

**Role:** Frontend / Fullstack Developer

**Goal:** Build a real, working Offline-First Personal Budgeting App.

---

## 🎯 Problem Statement

Most budgeting tools fail when the internet drops. Users want something that:

- Works **completely offline**
- Auto-saves **every keystroke** locally
- Syncs safely when the network returns
- Behaves like **Google Docs offline mode**
- Never loses their data

Your task is to build **BudgetBox** following Local-First principles.

---



## What You Need to Build

### 1. Add / Edit Monthly Budget

A single form with these fields:

<u>Field</u>	<u>Description</u>
Income	Monthly income input
Monthly Bills	Rent, EMI, utilities
Food	Groceries + dining
Transport	Fuel, cab, commute
Subscriptions	OTT, SaaS, apps
Miscellaneous	Others

- Allow **edit anytime**
- Auto-save field value instantly
- Work offline with nowhere to get stuck

---

## 2. Auto-Generated Dashboard

Once data is saved locally, generate a dashboard:

### Analytics Required

- 🔥 **Burn Rate** (Total expenses / Income)
  - 💰 **Savings Potential** (Income – Total Spend)
  - 📅 **Month-End Prediction** (Based on current trend)
  - 🎂 **Category Pie Chart** (Use any chart lib like Chart.js or Recharts)
  - ⚠️ **Anomaly Warnings**
    - e.g., “Subscriptions are 30% of your income — too high!”
    - rule-based only, **no GPT usage required**
- 

## 3. Local-First Data Behavior (Mandatory)

You must implement:

### Local DB

- IndexedDB via LocalForage/rxDB/Zustand-persist

### Offline Behavior

- App works even with **0 internet**
- Every keystroke is **auto-saved**
- Show offline indicator

### Sync Logic

A **Sync** button with clear statuses:

<u>Status</u>	<u>Meaning</u>
Local Only	Saved locally, never synced
Sync Pending	Edits waiting for network
Synced	Both server & local are aligned

---

## 4. Optional AI Suggestions (Simple Rule Engine)

Not GPT-based. Just rule-based examples:

- If food > 40% of income → “*Reduce food spend next month.*”
- If subscriptions > 30% → “*Consider cancelling unused apps.*”
- If savings negative → “*Your expenses exceed income.*”

Keep it simple.

---

## Tech Requirements

### Frontend (Mandatory)

- **Next.js 15** (App Router)
- **React 18 + TypeScript**
- **State:** Zustand
- **Styling:** TailwindCSS

Bonus:

- Service Workers (optional)
  - PWA install (optional)
- 

### Backend

Choose one:

#### Option A — Node.js (Express/Next API Routes)

#### Option B — FastAPI (Python)

Mandatory REST Endpoints:

##### **POST /budget-sync**

Push local data → server  
Request body: budget object  
Response: success + timestamp

##### **GET /budget/latest**

Fetch last saved server version  
Return: latest budget object

---

## Database

**Local DB:** IndexedDB (LocalForage / rxDB / Zustand Persist)

**Server DB:** PostgreSQL

Sync Storages (Optional but plus points)

- Save exported JSON locally
- User can optionally sync to:
  - Local
  - Google Drive
  - iCloud
  - DigiLocker

(Not required, just extra credit.)

---



## Deliverables

### 1. GitHub Repository

Should include:

- `/frontend`
- `/backend`
- `README.md`
- Architecture diagram
- Setup steps
- Instructions to test offline mode
- Screenshots / GIF demo

### 2. Hosted Live Demo

Use:

- Vercel (frontend)
- Railway / Render / Supabase / Fly.io (backend)

The app must open and work at all times.

---

# Demo Login (Must Be Hard-Coded in Seed Data)

Email [hire-me@anshumat.org](mailto:hire-me@anshumat.org)

password HireMe@2025!

This user **must exist** in your hosted deployment, so reviewers can log in **without registration**.



# Evaluation Criteria

<u>Category</u>	<u>Weightage</u>
Local-first data design	30%
Offline experience	25%
Sync correctness	15%
UI/UX quality	10%
Code structure	10%
README & architecture clarity	10%



# Submission Format

Submit this as a comment in the assignment form:

Form Link: [forms.gle/Bjmec4ajeJtc7697A](https://forms.gle/Bjmec4ajeJtc7697A)

XML  
Name :  
GitHub Repo :  
Live Demo Link :  
Tech Stack Used :  
Notes for Reviewer (optional) :

