Personal Finance Analyzer

America Pacheco, Bailey Bounnam, Ross Henderson, Brian Fang, Bryan Partida

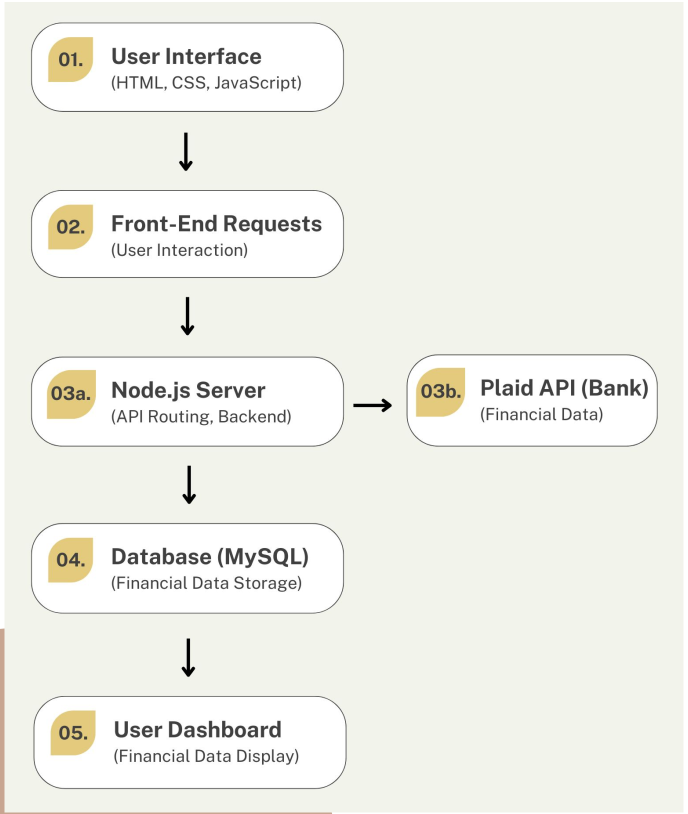
Why Create a Budget App?

- Financial Awareness Control
  - Track income, expenses, savings.
- Automation and Convenience
  - No spreadsheets, no notes.
- Expense Management and Reduction
  - Spot unnecessary expenses
- Goal Oriented Savings
  - Set financial goals
- Data-Driven Financial Insights
  - Make informed decisions
- Improved User Experience Over Traditional Methods
  - Simple financial tracking with user-friendly dashboard.

How we're going to build it

- Development Approach
  - Front-End: HTML, CSS, JavaScript for UI/UX
  - Back-End: Node.js with Express for API and business logic
  - Database: MySQL for structured financial data storage
  - External API: Plaid API for retrieving financial transactions
- Project Workflow
  - Version Control: GitHub repository for collaboration
  - Communication: Discord for team coordination
  - Deployment: Cloud-based deployment for public access
- Key Milestones
  - Set up GitHub and frameworks
  - Build front-end and integrate with backend
  - Implement Plaid API for transaction retrieval
  - Develop core features
  - Testing, debugging, and final deployment

System Architecture



Front-End: displays dashboard and sends requests

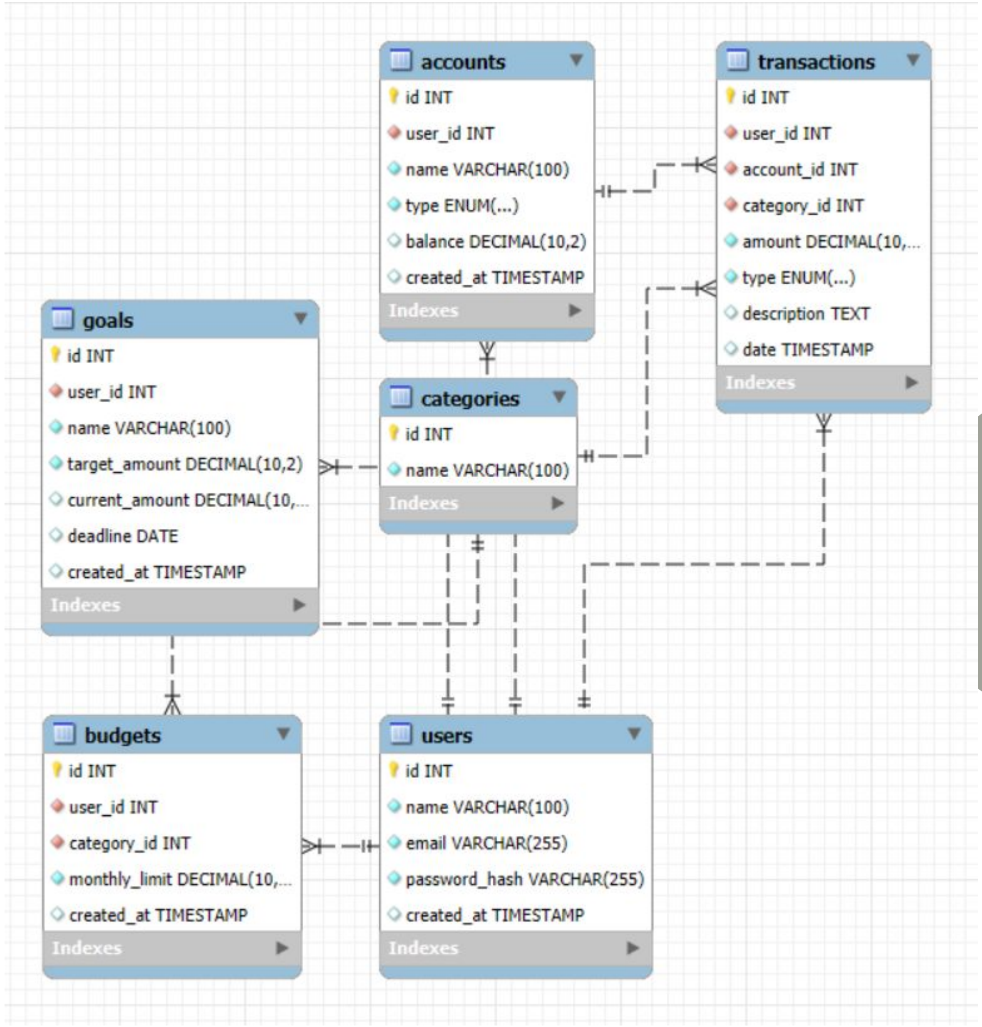
Back-End: processes logic, authentication, and API calls

Plaid API: fetches financial data securely

Database: stores user data, transactions, budgets, goals, accounts, and categories

Dashboard: shows processed data

MySQL Schema



Features & Functionality

Key Features:

- **Dashboard:** Visualizes spending, budgets, and transactions.
- **Budgeting Tool:** Set, track, and adjust budgets.
- **Subscription Tracker:** Identifies recurring payments.
- **Transaction Management:** Categorizes and filters expenses.
- **Financial Reports:** Custom spending trends and summaries.
- **Secure Bank Integration:** Uses Plaid API for real-time transaction retrieval.

Technical Overview:

- **Front-End (HTML, CSS, JavaScript):** Interactive UI with real-time data visualization.
- **Back-End (Node.js, Express):** Handles API requests, Plaid integration, and business logic.
- **Database (MySQL):** Stores users, transactions, budgets, and subscriptions.
- **Data Flow:** RESTful APIs between components for seamless data flow.

UI & User Experience

User Interface:

- **Intuitive Dashboard:** Clean design with quick access to key insights.
- **Mobile-Friendly:** Responsive layout for all devices.
- **Dynamic Visuals:** Graphs and charts for financial trends.
- **Dark/Light Mode:** Customizable user experience.

User Experience:

- **Seamless Onboarding:** Easy setup for linking accounts and setting goals.
- **Real-Time Insights:** Instant budget updates and spending alerts.
- **Smooth Navigation:** Optimized performance with minimal load times.
- **User Feedback & Error Handling:** Clear messages and confirmations.

Security

Data Encryption:

AES-256 for stored data.

HTTPS for secure data transmission.

Authentication & Access Control:

OAuth 2.0 authentication for Plaid API connections.

Secure login & session handling (JWT tokens).

Privacy Protection:

No sensitive data storage (we don’t store bank credentials).

Users can unlink accounts & delete data anytime.

Other Challenges & Risks

- API Downtime
  - Risk: Plaid API limitations may disrupt functionality
  - Mitigation: Implement caching and fallback strategies
- Database Performance
  - Risk: High data volume may affect query efficiency
  - Mitigation: Use indexing and query optimization
- Team Coordination
  - Risk: Remote team members and different schedules
  - Mitigation: Clear communication and regular check-ins

Testing Strategy

- Unit Testing
  - Tools: Mocha for backend, Jest for frontend
  - Focus: Validate each component and logic
- Integration Testing
  - Tools: Postman
  - Focus: Test API and system interaction
- Usability Testing
  - Tools: Google forms
  - Focus: Gather user feedback to improve UI/UX
- Bug Tracking
  - Tools: GitHub Issues
  - Focus: Log, assign, and resolve bugs effectively