Advanced User Experience & Strategy Guide

For the Replit Development Agent

Version: 1.0
Date: 2025-08-14
Author: Replit Coach Too
Status: Draft

1. Objective

This document provides a detailed technical guide for the Replit Agent to implement a suite of advanced, user-facing features. These features are designed to enhance the user experience, provide strategic value, and make the platform's data more actionable and communicable. The guide is broken down into three core features: **1) The Guided Journey**, **2) The Goals & Progress Dashboard**, and **3) Flexible Reporting**.

2. Feature 1: The 'Guided Journey'

This feature set is designed to proactively guide the user and eliminate overwhelm.

2.1. Database & Backend Enhancements

- Modification to companies Table: Add a column to store the user's primary motivation.
Column Name	Data Type	Constraints	Description
... (existing columns)			
primary_motivation	VARCHAR(255)	NULLABLE	Stores the user's main goal, selected during onboarding.
- **New Backend Logic: SuggestionService**
  - Create a new service that analyzes a user's current state (e.g., data completeness, goals set) and generates a prioritised list of 1-3 "What's Next?" actions.
  - Create a new API endpoint GET /api/suggestions/next-steps that returns these actions.

2.2. Frontend Implementation

- **Onboarding Wizard (First-time login)**
  - **UI:** A 5-6 step modal wizard that appears for new users.
  - **Step 1: Motivation:** Ask "What's the main reason you're here today?". Options: "Measure my carbon footprint", "Prepare for a certification", "Answer retailer questions". The result is saved to primary_motivation.

    - **Step 2-4: High-Level Details:** Ask for product_category, number_of_employees, etc.
    - **Final Step:** The wizard's final button should be a direct call to action, e.g., "Great! Let's start by uploading your first electricity bill," which navigates the user directly to that task.
- **Contextual Help Icons**
    - **UI:** Implement a small ? icon next to every data input field across the platform.
    - **Component:** Create a reusable ContextualHelp React component.
    - **Content:** The component will display a popup on hover/click with three sections: "What It Is," "Why It Matters," and "Where to Find It." The content for each field will be stored in a central JSON file for easy management.
- **"What's Next?" Dashboard Module**
    - **UI:** A new, permanent module on the main dashboard (/app/dashboard).
    - **Logic:** On page load, this module will call the GET /api/suggestions/next-steps endpoint and display the returned actions as a dynamic to-do list.

3. Feature 2: The 'Goals & Progress' Dashboard

This feature transforms the dashboard into a strategic command centre.

3.1. Database & Backend Enhancements

- New Table: company_goals
Column Name	Data Type	Constraints	Description
id	UUID	PRIMARY KEY	Unique identifier for the goal.
company_id	UUID	FOREIGN KEY (companies.id)	Links to the company that set the goal.
kpi_name	VARCHAR(100)	NOT NULL	The KPI this goal relates to (e.g., 'energy_intensity').
target_value	NUMERIC	NOT NULL	The target value for the KPI.
target_date	DATE	NOT NULL	The deadline for achieving the goal.
start_value	NUMERIC	NOT NULL	The value of the KPI when the goal was set.
- **New API Endpoints:**
    - POST /api/goals: Creates a new SMART goal.
    - GET /api/goals: Fetches all active goals for the company.
    - GET /api/kpi-data: A new endpoint to provide historical and current data for the main dashboard KPIs.

3.2. Frontend Implementation

- **KPI Section (Top of Dashboard)**

    - **UI:** A row of 4-6 large, clear KPICard components.
    - **Required KPIs:** Total Carbon Footprint, Energy Intensity, Water Intensity, Waste Diversion Rate, Packaging Circularity.
    - **Logic:** These cards will be populated by data from the GET /api/kpi-data endpoint.
- **SMART Goal-Setting Module**
    - **UI:** A modal form that is triggered when a user clicks a "Set a New Goal" button.
    - **Logic:** The form will guide the user through the SMART framework with clear prompts for each step (Specific, Measurable, Achievable, Relevant, Time-bound), as described in the user request. On submission, it calls POST /api/goals.
- **Dynamic Progress Visualisation**
    - **UI:** For each goal set, a new GoalProgressCard will appear on the dashboard.
    - **Component:** This card must contain:
        - A **gauge chart** or **progress bar** (using the Recharts library) showing current progress towards the target_value.
        - A **trend line chart** (using Recharts) showing the historical data for that KPI, with a dotted line extending to the future target_date and target_value.

4. Feature 3: Flexible Reporting

This feature provides a modular, template-based system for communication.

4.1. Database & Backend Enhancements

- New Table: custom_reports
Column Name	Data Type	Constraints	Description
id	UUID	PRIMARY KEY	Unique identifier for the custom report.
company_id	UUID	FOREIGN KEY (companies.id)	Links to the company that owns the report.
report_title	VARCHAR(255)	NOT NULL	The user-given title for the report.
report_layout	JSONB	NOT NULL	Stores an ordered list of 'content block' IDs and their custom text.
- **New AI Integration:**
    - The backend must integrate with a Large Language Model (LLM) API (e.g., Gemini). The API key must be stored in Replit Secrets.
    - Create a new endpoint: POST /api/reports/refine-text.
        - **Request Body:** { "text": "we used less water", "audience": "Investor" }

        - **Logic:** This endpoint will construct a prompt for the LLM, sending the user's text and the target audience, and return the refined text.

4.2. Frontend Implementation

- **Report Creation Flow**
    - **UI:** A new page (/app/reports/create) where the user starts by selecting an **Audience Template** (Investor, Retail Buyer, Customer-Facing).
    - **Logic:** Selecting a template pre-populates the report builder with a recommended set of 'content blocks'.
- **'Content Block' Library & Drag-and-Drop Interface**
    - **UI:** The report builder will have a sidebar containing a library of available 'content blocks' (e.g., "Total Carbon Footprint Chart," "Packaging Circularity Table," "Founder's Story Text Block").
    - **Logic:** The user can drag and drop these blocks into the main report canvas to add, remove, and reorder sections. The layout is saved to the custom_reports.report_layout field.
- **AI-Assisted Narrative & Tone Editor**
    - **UI:** Within each text-based content block, there will be a "Refine with AI" button.
    - **Logic:** Clicking the button opens a modal where the user can select a target audience. This triggers a call to the POST /api/reports/refine-text endpoint. The returned text is then shown to the user, who can choose to accept or reject the suggestion.