

Role:- Prompt Engineer

Task Code:- TW-TSK-PE-25-01

Entire AI Chat Transcripts:-

Research work:-

- >Before generating prompt, one have to research over the internet from already present similar applications to understand all technicalities and core functionalities working.
- > This gives proper idea and clarifications over what should our application look like .
- > Engineers can also read documentations to understand concepts.

Prompt Generation using Gemini:-

- >list all features that you need to build an app.
- >first develop absolute core functions to provide the base for the app.Top 5 to 10 features that is mandatory to have in such app.also define the order in which these features need to be developed.
- >absolute core foundation and built the key differentiator of the app.
- >complete copy paste prompt for AI coder.
- >not too big not too small prompt.
- >give guidelines and not exact instructions.

Step-by-Step AI Interaction Log

1. Initial Prompt & First AI Answer

- **My Prompt:** "You are a senior prompt engineer who designs modern, dynamic, effective web applications. Your objective is to read the provided PDF, and generate a prompt for an AI coder or UI-generating AI, keeping in mind all the features required for the web application. Give all the necessary features and their sub-features with proper details, and arrange the important features in descending order. If possible, do research on similar web applications already present on the internet. Do not miss any features or details in the PDF. Use modern technologies to build the web application in the prompt."
- **AI Answer:** The AI provided a detailed, multi-section prompt for an application named "Pantry Pulse." It correctly identified the two user types, the

Company Admin and the **Vendor**, and organized the features based on their user-specific roles. The response included a technology stack (

Next.js, Tailwind CSS, Node.js, Express.js, MongoDB) and arranged features in descending order of importance, starting with the Admin Dashboard and its sub-features like consumption logging , inventory tracking, and reporting. It also detailed the Vendor Dashboard with its pricing , sales monitoring, and invoice generation features.

2. Refinement & Iteration

- **My Prompt:** "Reread the PDF details once again precisely, also add the references of such app present over the internet. I want you to generate a prompt which is not long not short but perfect to help an AI coder."
- **AI Answer:** The AI refined the previous response by making the prompt more concise and focused. It reread the PDF to ensure all key deliverables, such as the need for 5-6 screens and demonstration of navigation flow, were implicitly covered. The response maintained the clear, two-user structure but presented the features in a more streamlined format. Crucially, it added a new section with

references to similar applications like **PantrySoft** and **My Pantry Tracker**, fulfilling the research request. The technology stack and screen requirements were reiterated to ensure the prompt was comprehensive but not overly verbose.

Summary of Design Decisions and Learnings

- **Iterative Refinement:** The process demonstrated the importance of **iterative refinement** to improve clarity and add missing context, making the final prompt more robust.
- **Prompt Conciseness:** A key learning was that a "perfect" prompt balances detail with brevity, effectively using bullet points and clear headings to guide the AI coder.
- **Context is Key:** Providing the AI with specific instructions to "reread" and "add references" was essential to ensure a more accurate and complete output that aligned with the user's expectations.
- **Structured Prompting:** Organizing the prompt by user role and feature importance proved highly effective in guiding the AI to prioritize the most critical functionalities of the application.

AI coder for building web applications:-

“Initial detailed or main prompt to start the agent building application”

“” Pantry Pulse: Office Pantry Management Web Application

Objective: Design a 5-6 screen web application for office pantry management that serves two user types: a Company Admin and a Vendor. The application should streamline the process of logging consumption, tracking inventory, and automating billing.

The design should be modern, clean, and intuitive.

Key Features by User Role (in Descending Order of Importance)

1. Admin Dashboard (Crucial for Daily Operations)

This section is the core of the application, managing all consumption data and reporting.

Consumption Logging: A screen for the admin to log the consumption of pantry items on a daily or per-visit basis. The interface should be easy to use with fields for item name, quantity, and date.

Item Management: Admins must be able to add, edit, or remove items from the master pantry list. This includes setting item names, units (e.g., box, cup, piece), and initial stock levels.

Inventory Tracking: A module to track current stock levels of all pantry items. This feature helps prevent stockouts and aids in proactive management. A visual representation, such as a progress bar or color-coded status (e.g., green for full, yellow for low, red for out of stock), would be highly effective.

Reporting & Analytics: A dedicated screen to generate and view consolidated reports.

Filters: Allow filtering data by date range (daily, monthly, per-visit) and by specific pantry items.

Visual Reports: Display consumption patterns using interactive charts like a bar graph for daily consumption and a pie chart for total usage by item.

Export Functionality: Enable the admin to download these reports in a common format like PDF or CSV.

2. Vendor Dashboard (Crucial for Billing & Sales)

This dashboard automates the financial aspects of the pantry operation.

Pricing Management: A screen for the vendor to set and adjust prices for each pantry item. This includes an overview of all items and their current prices.

Sales & Quantity Monitoring: The vendor needs a dashboard view of total quantities sold. This can be based on the consumption logs provided by the admin.

Invoice Generation & Billing: The most critical feature for the vendor.

Automated Invoicing: A function to generate monthly invoices based on the consumption data.

Download Invoices: The ability to download these invoices for billing purposes. The download format should be a professional PDF document.

3. Application Design & Technology

Screens: The application will consist of the following screens:

Login/User Selection Screen (to authenticate both Admin and Vendor users).

Admin Dashboard (Main Overview with quick-glance stats).

Admin Consumption Logging Screen.

Admin Reports & Analytics Screen.

Vendor Dashboard (Main Overview).

Vendor Invoicing & Pricing Screen

User Interface (UI): Use a clean, modern, and cohesive UI with a consistent design language, color scheme, and layout. The style should be minimalist, leveraging a flat design and card-based layouts for easy readability.

Technology Stack:

Frontend: React.js or Vue.js for a dynamic, single-page application experience. Use a modern CSS framework like Tailwind CSS or styled-components for consistent and responsive design.

Backend: Node.js with Express.js for a robust API.

Database: A flexible NoSQL database like MongoDB for efficient data storage. """

Step 1: Initial Prompt and AI Plan

- **My Prompt (Initial Request):** I provided a detailed, multi-section prompt outlining the **Pantry Pulse** application. It specified key features for both the **Admin Dashboard** (Consumption Logging, Inventory Tracking, Reporting) and the **Vendor Dashboard** (Pricing, Invoicing). The prompt also included technical details such as the desired number of screens (5-6), a modern UI, and a technology stack (**React.js/Vue.js, Node.js, MongoDB**).
- **AI Plan & Clarification:** The AI acknowledged my request and presented a plan. It sought clarification on key implementation details to avoid assumptions. The AI's questions were:
 - **Scope:** Should the initial version include both the Admin and Vendor dashboards?
 - **Authentication:** What authentication method is preferred (e.g., email/password, Google SSO)?
 - **Technical Stack:** Specific choices for PDF generation, chart libraries, export features, and user account support.

Step 2: My Input and Project Kick-off

- **My Prompt (Answers to Clarification):** I provided direct answers to the AI's questions to move the project forward:
 - **Scope:** "admin and vendor together"
 - **Authentication:** "email and password"
 - **Technical Choices:** "browser-based pdf," "advanced interactive charts," "advanced export," and "simple role-based" authentication.
- **AI Action:** Based on my input, the AI confirmed it had all necessary details and began the implementation process. It proceeded to set up the project structure, install dependencies, and create all the core components of the application.

Step 3: Initial Delivery and My Feedback

- **AI Action (Initial Build):** The AI announced that it had created a comprehensive application with all the requested features and began testing the backend endpoints. The tests showed a high success rate (96.9%), indicating core functionalities were working correctly.
- **My Prompt (Bug Report):** I provided critical feedback on two specific issues found in the initial build:
 1. "in log new consumption there is no item to select that should be coffee, tea and biscuits"
 2. "there is no back page option"
- **AI Action (Bug Fixes):** The AI acknowledged the issues immediately. It created a plan to fix them by:
 1. **Seeding the Database:** Adding initial items like coffee, tea, and biscuits to the database.
 2. **Improving Navigation:** Implementing back buttons and a better navigation flow throughout the application.

Step 4: UI Enhancement and Final Refinement

- **My Prompt:** "I want you to enhance the UI and give me a better design for the main dashboard"

- **AI Answer:** The AI acknowledged my request and provided a plan to improve the UI. It suggested using a more modern color palette, adding card-based layouts for better data visualization, and improving the overall aesthetic appeal of the dashboard.
 - **My Prompt (Final Quality Assurance):** "I want you to test the frontend yourself first properly so that I have to tackle less issues or no issues"
 - **AI Action (Comprehensive Testing):** The AI agreed with my request, recognizing the value of pre-emptive quality assurance. It stated its intention to conduct a comprehensive frontend test to ensure a smooth user experience. This final step shows the AI's capability to understand and act on a request for a proactive, quality-focused process.
-

Summary of Design Decisions and Learnings

1. **Iterative Prompting:** The process of providing feedback and making refinements in successive prompts was crucial. This iterative approach allowed for course correction and led to a more complete and accurate output.
2. **Precision is Key:** The ability to provide precise, direct answers to the AI's clarification questions was essential for moving from a general concept to a specific implementation plan without ambiguity.
3. **Proactive Problem Solving:** The AI demonstrated an understanding of proactive problem-solving by not only fixing the reported bugs but also offering to conduct a comprehensive frontend test to prevent future issues.
4. **UI/UX Enhancement:** The prompt for UI enhancement highlights the flexibility of using AI for design iterations. It shows that the process isn't limited to a single output but can be used to refine and improve visual elements based on new requirements.
5. **Role-Based Design:** The design correctly prioritized two distinct user roles (Admin and Vendor), a critical requirement from the problem statement. This ensures the application's functionality is properly segmented and tailored to each user type.