Studio
Mapa mental
Vibe Coding: Soluciones de IA con Estrategia Profesional
Basado en 1 fuente
Vibe Coding: Soluciones para el Desarrollo con IA

<
El 'Vibe Coding' y su Traición Oculta
<
Prompts Esenciales para 'Vibe Coders'
<
Conclusión
<
Implicación: 'Vibe Coding' = Irreflexión
Narrativa Común: Fallos de Seguridad, Apps Pobres
Diferencia Crucial
<
Ejemplos de Pensamiento Necesario
<
1. Prompt de Product Manager
<
2. Prompt de Ingeniero UX
<
3. Prompt de Arquitecto de Sistemas
<
4. Prompt 'Bola de Demolición' (The Wrecking Ball)
<
5. Prompt de Pulidor de UI (UI Polisher)
<
Construir sin Sintaxis NO significa Construir sin Pensar
El Proceso es Clave, no Solo las Herramientas
Pensar como un Equipo de Producto Completo
Significado: Construir sin sintaxis
NO Significa: Construir sin pensar
Redis Cache y Escalamiento
Pipeline de Despliegue
Propósito: Definir qué, por qué y cómo medir el éxito
Evitar: Conceptos vagos (ej. 'app de seguimiento del sueño')
Estructura del Plan de Producto
<
Ejemplo de Aplicación: Color IQ (Extensión de Chrome)
<
Propósito: Diseñar experiencias de usuario e interfaces visuales
Traducción de: Historias de PM a sistemas de diseño
Ventajas
<
Tareas Clave
<

Output del Prompt (Ejemplo Color IQ)
<
Consideración de Modelos: Claude Opus para Contextos Grandes
Propósito: Detalles técnicos para implementar funciones
Ejemplo: Forkcast (App de Cocina con IA)
<
Pregunta Central: ¿Cómo lo hará la aplicación?
Decisiones y Compensaciones (Ej. Escalamiento para 500 usuarios)
Consideraciones Clave
<
Output del Prompt (Ejemplo Forkcast)
<
Propósito: Rectificar desviaciones del plan inicial
Función: Comparar lo Intendido vs. lo Construido (Product Manager, UX Engineer)
Problema Común: Funcionalidad avanzada construida, pero atajos en la implementación
Rol: Especialista en Validación Meticulosa de Requisitos
Proceso de Validación
<
Output del Prompt (Ejemplo Forkcast MVP - Primera Iteración)
<
Propósito: Mejorar la estética y pulir la UI
Problema: App funcional pero 'fea' o poco inspiradora
Función: Calificar el rendimiento actual vs. especificaciones de diseño originales
Beneficio: Separar Codificación de Diseño -> Mejores Resultados
Input al Prompt
<
Output del Prompt (Ejemplo Forkcast)
<
Enfoque Primero en el Problema
Resumen Ejecutivo
<
Especificaciones Detalladas de Funciones
<
Concepto: Match de ropa con estación de color personal
Basado en: Análisis de color estacional
Output del Prompt (Ejemplo)
<
Imagen Clara de la App
Mejor Diseño (procesamiento enfocado)
Filosofía de Diseño Cohesiva
Principios UX (meta de pantalla, presentación, revelado gradual)
Sistema de Diseño Integral
<
Documentación Robusta
Archivo Readme (estructura del directorio)
Análisis UX Detallado por Función
<
Guía de Estilo Integral

<
Flujo de Selección de Temporada de Color
Procesamiento de Imágenes
Sistemas RAG Agentic
Modelos de Visión
Chat AI
Arquitectura del Sistema
Infraestructura Subyacente
Modelado e Interfaz de Datos
Diseño de API
Integración con Servicios de Terceros
Seguridad y Rendimiento
Stacks de Frontend y Backend
Base de Datos y Almacenamiento
Resumen Ejecutivo (ej. Tech Stack y Requisitos)
<
Detalles del Stack Tecnológico
<
Servicios Externos
<
Plan de Arquitectura de Alto Nivel
<
Beneficio: Asegurar la Base, Evitar Decisiones Improvisadas
1. Requisitos
<
2. Comportamiento Previsto vs. Construido
<
3. Resumen Ejecutivo y Desglose por Función
Resumen Ejecutivo: % de Requisitos Cumplidos (ej. 62%)
Mejoras / Brechas Principales
<
Beneficio: Asegurar que el Brainstorming se Refleje en la App Final
Guías UX
Funciones
Guías UX/UI Detalladas
Requisitos del Producto
Diagnóstico: Fundamentos Técnicos Sólidos, pero Calidad de Diseño Insuficiente
<
5 Mejoras Críticas
<
Ejemplo Visual: Inconsistencias entre funcionalidades (chat AI vs. libro de cocina)
Siguiente Paso: Usar un agente de ingeniería frontend para implementar cambios
Elevator Pitch (una frase)
Declaración del Problema Central
Público Objetivo
Diferenciación de la Solución
Medición del Éxito
Nombre de la Función

Historia de Usuario (como un usuario...)
Problema Raíz a Resolver
Requisitos Funcionales
Elevator Pitch
Declaración del Problema Principal (desperdicio de dinero en ropa)
Público Objetivo (mujeres conscientes de la moda)
Funciones (Ejemplos)
<
Flujos de Usuario
Gestión de Estado y Datos
Objetivos de Rendimiento
Tokens (color, tipografía, espaciado)
Componentes (compartidos y específicos)
Interacciones y Estados de Error
Sistema de Movimiento y Animación
Meta Primaria
Puntos de Dolor Superados
Especificación de Diseño Visual (Mockups)
<
Colores (primarios, secundarios, semánticos, acento)
Tipografía
Espaciado
Componentes
Tokens (animaciones, colores, espaciado)
Frontend: React Native, Expo
Backend: FastAPI
Base de Datos: Supabase
Procesamiento AI: OpenAI
Despliegue: Railway
Requisitos de Rendimiento (tamaño de imagen, tiempo de respuesta AI, usuarios activos)
Frontend (React Native, Expo, Zustand, React Query, TypeScript)
Backend (Python FastAPI, GraphQL, Redis Cache, Redis Q)
OpenAI (API de Visión, Recetas, Embeddings)
Cloudinary (alojamiento de imágenes)
Sentry (monitoreo de errores)
Comunicación entre Servicios
Dependencias Internas y Externas
Compatibilidad de Librerías
Almacenamiento del Estado de Frontend (autenticación, recetas, usuario, chat AI)
Coincidencia con Implementación
Identificar: No Construido / Construido No Solicitado
Calidad e Implementación de Viajes de Usuario
Autenticación/Onboarding: Brechas
Captura de Fotos: 95% (falta detección de poca luz)
Input de Menú: 90% (auto detección de cocina omitida intencionalmente)
Generación de Recetas: 70%
<
Historial y Gestión de Recetas: Parcialmente Completado

<
Función AI (no hecha en esta sección)
Problema Principal: Implementación inconsistente y falta de pulido
Aplicar el sistema de diseño consistentemente
Implementar sombras, gradientes, profundidad
Completar pantallas
Mejor tipografía y flujo visual
Pulido con sensación premium (microinteracciones)
Feedback en Tiempo Real (superposición de color)
Recomendación de Color Inteligente
Indicador de % de Match
Vista Detallada del Análisis
Estados de Match (Perfecto, Bueno, Malo, Muy Malo)
Posicionamiento (superposición en imagen)
Jerarquía Visual
Animaciones (carga)
Snippets de Código
Largo Tiempo de Procesamiento
Falta de Notificaciones Push
Necesidad de Más Detalles en la Vista de Recetas
Lista de Recetas
Vista Detallada
Capacidades de Edición Básicas
Búsqueda y Filtrado (backend construido, no implementado)