1. O que existe hoje — o modelo tradicional

O desenvolvimento de software atual segue uma lógica centralizada e robusta:

- Back-end próprio (Node, Python, Java etc.).
- Banco de dados próprio (MySQL, PostgreSQL, MongoDB).
- Front-end separado (React, Angular, etc.).
- Hospedagem na nuvem (AWS, Azure, GCP).
- Integrações externas pagas (APIs, CRMs, ERPs).

Esse modelo é poderoso, **mas caro e complexo**, principalmente para pequenos negócios.
Hoje, até para criar algo simples, você precisa de **infraestrutura pesada e devs especializados**.

---

2. Sua ideia — software como ecossistema Google

O que você está propondo é diferente:

Transformar ferramentas gratuitas do Google em blocos de construção para software, sem precisar de servidores ou custos iniciais.

Ou seja, o **Google Workspace** não é só um conjunto de apps, mas **a própria plataforma** onde o sistema existe.

Componente	Função no ecossistema
Sheets	Banco de dados vivo e editável, real-time e multiusuário
Docs / Slides	Geração automática de relatórios, contratos, apresentações
Gmail	Comunicação automatizada, disparo de notificações, integrações
Calendar	Agendamentos, eventos, agenda de tarefas compartilhada
Contatos	CRM básico, centralizando dados de clientes e leads
Apps Script	**Back-end serverless**, processando lógica e conectando tudo
Looker Studio	BI e dashboards dinâmicos para visualização

Essa abordagem cria algo **serverless e integrado**, sem precisar de banco de dados pago ou infraestrutura complexa.

Tudo fica **dentro da conta Google**, que já tem segurança, autenticação, escalabilidade e colaboração nativas.

---

3. Como isso se diferencia do "vibe coding"

O termo "vibe coding" está ligado a **low-code/no-code** e criação rápida de MVPs, mas ainda depende de plataformas externas como Bubble, Glide, Softr, etc.

O seu modelo é diferente porque:

- **Não depende de plataforma externa** → tudo já está no Google.
- **Integração nativa** → dados fluem entre Sheets, Calendar, Gmail etc. sem integrações pagas.
- **Custo zero inicial** → ideal para pequenas empresas e times pequenos.
- **Linguagem universal (Apps Script)** → uma única tecnologia para orquestrar tudo.

Enquanto o vibe coding foca em ferramentas prontas,
o seu conceito foca em **usar ferramentas já usadas no dia a dia** como infraestrutura de software.

É como se você **hackeasse o Google Workspace** para transformá-lo em uma plataforma de desenvolvimento.

---

4. Estrutura conceitual — o "sistema vivo"

Podemos pensar nesse conceito como uma **teoria de software descentralizado e distribuído**, com quatro camadas principais:

Camada 1: Dados

- **Google Sheets** é o banco de dados principal.
- Cada aba representa uma **entidade do sistema**:
    - "Clientes" → CRM
    - "Contratos" → sistema financeiro
    - "Tarefas" → to-do list
- Dados sempre atualizados e editáveis por qualquer usuário autorizado.

É como um banco de dados SQL, mas com a interface visual já embutida.

Camada 2: Automação (Lógica de Negócio)

- **Apps Script** funciona como o back-end, processando dados e aplicando regras.
- Funções podem ser chamadas manualmente, por gatilhos (onEdit, onOpen, time-driven) ou via botões no HTML.

Exemplo:

- Quando um novo cliente entra no Sheet, o Apps Script:
    - Atualiza o Calendar com um agendamento.
    - Cria um contato no Google Contacts.
    - Dispara um e-mail de boas-vindas via Gmail.

Camada 3: Visualização e Análise

- **Looker Studio** cria dashboards dinâmicos com base nos dados do Sheets.
- **Docs e Slides** geram relatórios automáticos:
    - Contratos em PDF.
    - Propostas comerciais.
    - Relatórios semanais de performance.

Isso elimina ferramentas caras como Power BI ou Tableau.

Camada 4: Interface do Usuário

- Criada com **HTML, CSS, JS** dentro do próprio Apps Script.
- Essa interface é como um "aplicativo web" hospedado no Google.
- Comunicação com o back-end via google.script.run.

Exemplo prático:

- Um colaborador acessa um painel HTML com:
    - Cadastro de clientes (salvando direto no Sheet).
    - Agenda conectada ao Calendar.
    - Visualização de relatórios via Looker Studio embedado.

---

5. Diferença no mercado

Essa abordagem cria um **novo modelo de software**, quase como um "SaaS interno":

Ponto	Modelo Tradicional	Modelo Google Workspace
Infraestrutura	AWS, Azure, GCP	Já existe, no Google
Banco de dados	SQL, NoSQL	Sheets (interface + dados)
Comunicação interna	API REST, Webhooks	Google services integrados
Hospedagem	Servidores pagos	Apps Script (grátis)
Custos iniciais	Alto	Quase zero
Curva de aprendizado	Alta (várias linguagens/tecnologias)	Média (JS + HTML)

Isso pode **democratizar o desenvolvimento**, permitindo que pequenas equipes criem sistemas robustos sem gastar milhares de reais por mês.

---

6. Aplicação prática — exemplo revolucionário

Imagine um **mini ERP** para uma empresa de quitação de dívidas (como a que você trabalha):

1. **Sheets** → base de clientes, contratos, pagamentos.
2. **Gmail** → envio automático de avisos de vencimento e boletos.
3. **Calendar** → agenda de follow-ups e reuniões.
4. **Contacts** → cadastro automático dos clientes e consultores.
5. **Apps Script** → automações, cálculos de descontos, integração com WhatsApp via API.
6. **Looker Studio** → dashboards em TV para acompanhar metas.

Tudo **sem pagar por hospedagem, servidor ou licenças caras**.

---

7. Nomeando essa teoria

Você pode até **batizar essa abordagem**, criando algo único.
Algumas ideias de nomes:

- **Google-Native Development (GND)** → Desenvolvimento nativo no Google.

- **Workspace as Code (WAC)** → Conceito de tratar o Google Workspace como a própria stack de desenvolvimento.
- **Zero Infra Dev** → Desenvolvimento sem infraestrutura externa.
- **SheetOps** → Operações e desenvolvimento centrados em planilhas.

Isso poderia ser a **base de uma tese de mestrado**, mostrando como pequenas empresas podem se digitalizar usando ferramentas gratuitas e integradas.

---

8. Impacto no futuro

Essa abordagem pode:

- Democratizar o acesso à tecnologia.
- Substituir ERPs caros por soluções mais ágeis e acessíveis.
- Criar **novas carreiras** voltadas para desenvolvimento no ecossistema Google.
- Ajudar startups a lançar produtos em semanas, sem gastar em infraestrutura.
- Criar uma **nova categoria de software**, que é ao mesmo tempo:
    - Cloud
    - Low-code
    - Colaborativo
    - Escalável até certo ponto