TAHSHR.AI

ابنِ Dashboard ثلاثي الأبعاد لبيزنسك باستخدام Claude

دليل عملي يحوّل بيانات المبيعات والعملاء والمخزون إلى 3D World تفاعلي داخل المتصفح - عناصر تتحرك، تُضغط، وتكشف لك الأرقام التي تهمك.

Interactive 3D | Real-time Data | Stripe / Square | Three.js | Claude + MCP

تابعني للمزيد من أدلة الذكاء الاصطناعي
TAHSHR.AI@

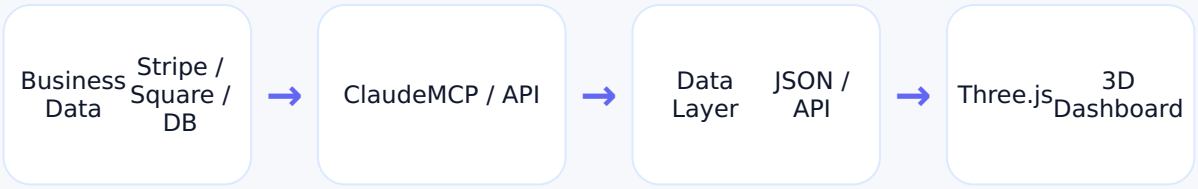
للتواصل
info@taha-ai.com

01

الفكرة الأساسية

وش قاعد تبني بالضبط؟

أنت لا تغيّر البيانات نفسها. أنت تبني طبقة عرض ثلاثية الأبعاد فوق بيانات البزنس بحيث يصبح كل جزء من العمل عنصرًا تفاعليًا.



مثال: كوفي شوب

- كل فرع = متجر مستقل
- كل طاولة = عدد الطلبات
- البار = المبيعات اللحظية
- المخزن = Inventory
- الموظفون = ساعات/إنتاجية

مثال: متجر إلكتروني

- المتجر = المبنى الرئيسي
- المنتجات = رفوف أو صناديق
- العملاء = شخصيات أو نقاط ضوئية
- الإيرادات = عداد أو شريط طاقة
- المخزون = مستوى تعبئة بصري

الفكرة الأقوى:

لا تجعل الـ3D مجرد شكل جميل. اربط كل عنصر بسؤال حقيقي في البزنس: كم باع؟ كم بقي؟ هل فيه مشكلة؟ وش يحتاج انتباه الآن؟

DATA

الخطوة 1

اربط بيانات البزنس مع Claude

الـDashboard يصبح مفيدًا فقط لما يسحب بيانات حقيقية. استخدم Connector جاهز، MCP، أو API حسب النظام اللي تستخدمه.

Square

مناسب للكافيهات، المطاعم، المتاجر ونقاط البيع. تقدر تسحب Orders، المبيعات، Inventory، Catalog والعملاء عبر الـAPI.

Stripe

مناسب للإيرادات، المدفوعات، الاشتراكات، الفواتير، العملاء، والـrefunds. إذا كان عندك متجر أو SaaS، هذه غالبًا أسهل بداية.

CSV كبداية

إذا ما عندك API أو Connector، ابدأ بملف CSV. خلي Claude يبني النسخة الأولى على بيانات ثابتة، وبعدها بدّلها ببيانات live.

Shopify / CRM / Database

لو بياناتك في Shopify أو CRM أو قاعدة بيانات، اربطها عبر Connector/MCP متاح أو Backend بسيط يستخرج البيانات ويرجعها للـDashboard.

مهم جدًا:

لا تحط API keys داخل كود الواجهة الأمامية. خزن الأسرار في environment variables أو backend/server layer. الـDashboard في المتصفح لازم يشوف البيانات فقط، مو المفاتيح.

Prompt

```
I want to build a live 3D business dashboard. First inspect the connected business data source and list the metrics we can safely read. Group them into: revenue, customers, products, inventory, orders, and alerts. Do not build the UI yet. Return a clean data map with field names, update frequency, and any permission/security concerns.
```

MAP

قبل التصميم

حوّل الأرقام إلى Data Map واضح

أفضل Dashboard يبدأ من خريطة بيانات بسيطة: وش المقياس؟ من وين يجي؟ وكيف يظهر في العالم الثلاثي الأبعاد؟

البيانات	المصدر	العنصر ثلاثي الأبعاد	التفاعل
Revenue Today	Stripe / Square	لوحة مضيئة فوق المبنى	Click → تفاصيل اليوم/الأسبوع
Active Customers	CRM / Payments	نقاط/شخصيات داخل العالم	Hover → العدد والتغير
Inventory	Catalog / Inventory API	صناديق أو رفوف	اللون يتحول للأحمر عند الانخفاض
Top Products	Orders	منتجات على منصات	Click → المبيعات والهامش
Delivery Status	Logistics / DB	سيارات تتحرك	Click → route / ETA / status
Alerts	Rules Engine	Beacon / flashing icon	Click → المشكلة والإجراء المقترح

قاعدة ممتازة:

كل Object لازم يجاوب على سؤال. إذا ما فيه سؤال واضح خلف العنصر، غالبًا هو مجرد زينة ويزيد تشتيت المستخدم.

Inventory Alerts
7
needs attention

Customers
1,284
active today 46

Revenue
€24.8K
vs last week 18%+

WORLD

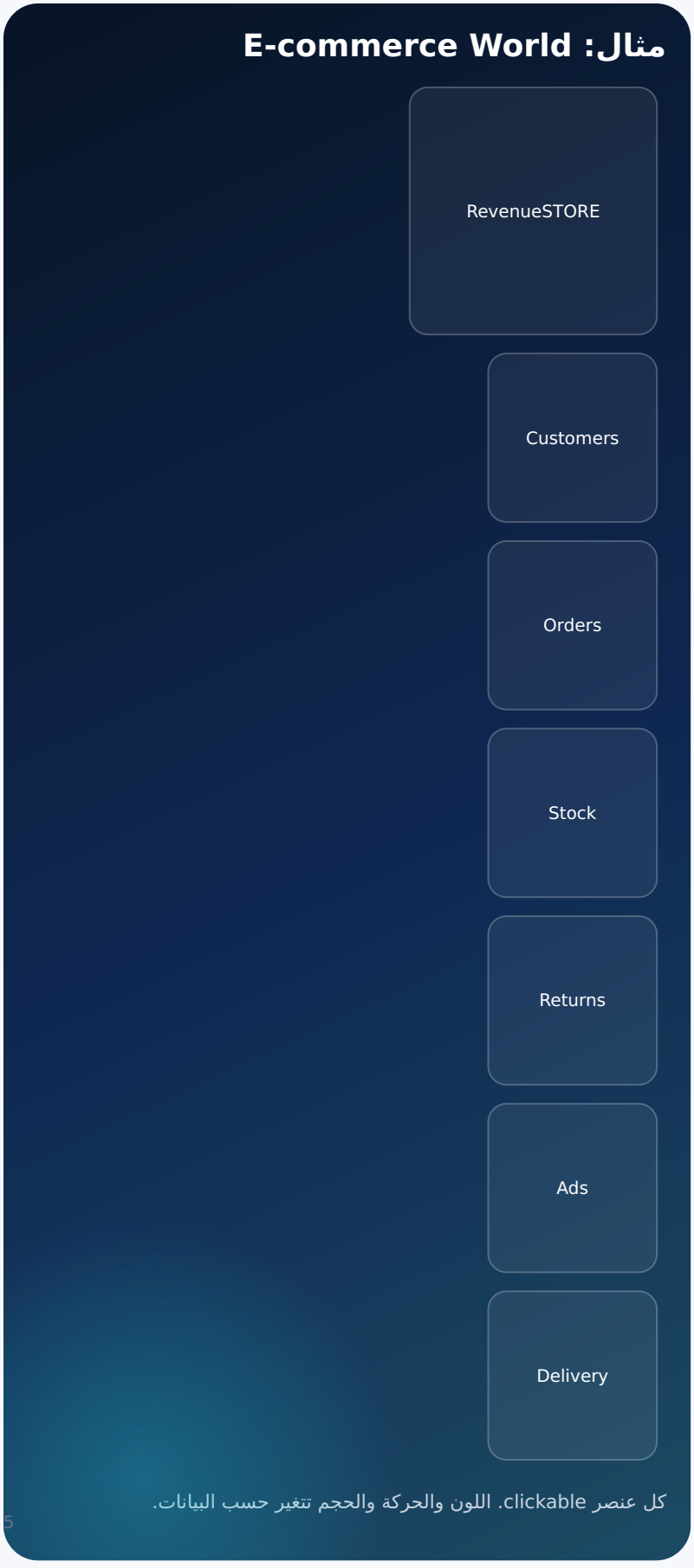
الخطوة 2

صمّم البزنس كأنه 3D World

اعطِ Claude صورة للمكان، Sketch، Screenshot، أو وصف نصي. المطلوب ليس نسخة معمارية دقيقة - المطلوب عالم بصري يساعدك تفهم البزنس بسرعة.

أعطِ Claude هذه القرارات

- نوع العالم: متجر، مدينة، مكتب، مستودع...
- الكاميرا: Orbit / Isometric / Free Look
- ستايل: realistic / minimal / game-like
- كيف يمثل كل KPI بصريًا
- وش يظهر عند الضغط
- وش يتحرك بشكل مستمر
- وش يصير عند وجود Alert



PROMPT

بروميت التصميم

خلّ Claude يحوّل وصفك إلى Scene Map

قبل كتابة الكود، خله يقترح خريطة العالم: وش كل عنصر؟ وين مكانه؟ وش البيانات اللي يمثلها؟ ووش يصير عند الضغط عليه؟

قبل ما تقول Build:

اطلب أولًا Scene Map وData Map. راجعها، عدّلها، وبعدها خله يبدأ التنفيذ. هذا يوفر عليك إعادة بناء العالم كامل بعدين.

```
Design Prompt
Design my business as an interactive 3D world. Use the business description and reference image I provide. Map every important business metric to a visible object in the scene. Clicking an object should open a clean analytics panel with the related KPIs. Use visual hierarchy: the most important business areas should be larger, brighter, or more central. Keep it useful, not decorative.

Before coding, return:
1. Scene map
2. Data-to-object map
3. Interaction map
4. Camera behavior
5. Alert behavior
6. Mobile fallback plan
```

راجع المنطق البصري

تأكد أن الحجم، اللون والحركة لها معنى واضح، مو مجرد مؤثرات.

اطلب نسخة بسيطة أولًا

ابدأ بـ5 إلى 7 objects فقط. لا تحاول تمثيل كل KPI في أول نسخة.

الخطوة 3

قل له يبنيها باستخدام Three.js

Three.js هي طبقة الـ3D داخل المتصفح. منها تحصل على Scene، Camera، Lights، Animation، Raycasting للنقر، وWebGL rendering.

Camera

حدد Orbit controls أو Perspective camera عشان المستخدم يتحرك حول العالم.

Scene

العالم اللي يحتوي كل المجسمات، الإضاءة، الأرضية، البيانات البصرية.

Animation Loop

يحرك السيارات، المؤشرات، الـparticles، والأرقام بناءً على البيانات.

Raycaster

يسمح للنقر على 3D object ومعرفة العنصر اللي اختاره المستخدم.

Data Adapter

طبقة تفصل الـAPI عن المشهد. إذا تغير مصدر البيانات ما تضطر تعيد بناء العالم كامل.

UI Overlay

استخدم HTML/CSS panel فوق الـcanvas لعرض التفاصيل بدل حشر النص داخل 3D.

أفضل Architecture:

Data Source → Backend/API → normalized JSON → Three.js scene → clickable analytics panel. خلك تفصل البيانات عن التصميم.

Build Prompt

```
Build this dashboard as a responsive web app using Three.js for the 3D scene and standard HTML/CSS for the analytics panels. Use a clean data adapter layer so the scene never talks directly to Stripe/Square credentials. Add raycasting for click interactions, OrbitControls, smooth camera transitions, loading states, error states, and mobile fallback.
```

COPY

البرومبت الرئيسي

Master Prompt جاهز للنسخ

هذا البرومبت مناسب لـClaude Code أو أي بيئة Claude تقدر تبني فيها المشروع فعليًا.

```
You are building an interactive 3D business intelligence dashboard.

GOAL
Turn my business into a clickable 3D world where each object represents a real business area and displays live metrics.

DATA
Use the connected data source(s) I authorize. Never expose API keys in frontend code. Create a normalized server-side data layer that returns only the metrics the dashboard needs.

3D STACK
Use Three.js. Build a responsive scene with PerspectiveCamera, OrbitControls, lights, smooth animations, raycasting for object selection, and graceful mobile fallback.

INTERACTIONS
- Hover: highlight object + show quick KPI
- Click: smoothly focus camera + open detailed analytics panel
- Alerts: change color / pulse / badge
- Filters: date range, location, product/category
- Refresh: show last-updated time and loading state

DESIGN
Make it premium, clean, cinematic, and useful. Avoid visual clutter. Use the reference image/business description to shape the world.

DATA OBJECTS
Map revenue, customers, orders, products, inventory, refunds/returns, marketing, and operations into distinct clickable objects.

DELIVERABLES
1. Working app
2. Clear folder structure
3. README with setup
4. .env.example
5. Mock-data fallback
6. Test checklist
7. Deployment instructions

Before coding, show me the proposed data map and scene map for approval.
```

مثال 1

ECOM

متجر إلكتروني: من Dashboard عادي إلى 3D Store

استخدم العالم كخريطة للبزنس نفسه بدل صفوف من الأرقام.

رفوف المنتجات

كل Shelf = Category. حجم المنتج يمثل Sales volume، واللون يوضح Stock health.

المدخل الرئيسي

إجمالي Revenue، Orders Today، Conversion trend. إذا اليوم قوي، الإضاءة أقوى أو المبنى يتوهج.

منطقة Returns

إذا زادت المرتجعات، المنطقة تبدأ تومض. Click يعرض المنتجات اللي تسبب المشكلة.

منطقة العملاء

Customer count، repeat customers، new vs returning. عند الضغط يظهر cohort سريع.

Marketing Tower

ROAS، Spend، CAC، top campaign. يتغير ارتفاع البرج حسب performance.

Delivery Zone

سيارات/طرود تتحرك. السرعة أو اللون يعكس shipping status أو delays.

Prompt

```
Turn my ecommerce store into a 3D retail world. Make categories into shelves, top products into hero displays, inventory into stock levels, customers into a visitor zone, returns into a warning area, and delivery into moving vehicles. Clicking any zone must open the related real metrics and trend charts.
```

مثال 2

CAFE

كوفي شوب: كل فرع وكل منتج قدامك

مناسب للبزنس اللي فيه POS ومخزون وفروع. Square أو أي POS API يقدر يكون مصدر البيانات.

Storage

Milk / beans / cups / low stock.

Menu Wall

أفضل وأسوأ المنتجات.

Counter

Revenue per hour + order volume.

Staff

ساعات العمل / workload إذا كانت البيانات متاحة.

Branches

كل فرع Mini shop مع KPI خاص.

Tables

طلبات، متوسط ticket، peak time.

فكرة قوية:

سوِّ Time Slider من 8:00 إلى 22:00. لما تحركه، يتغير العالم وبعرض كيف يتحرك الطلب والمبيعات خلال اليوم.

Prompt

```
Create a 3D coffee-shop operations dashboard. Represent each branch as a separate shop, the counter as sales velocity, menu items as clickable products, and the storage room as live inventory. Add a time-of-day slider so I can replay the business day and see when sales, queues, and stock pressure peaked.
```

OPS

مثال 3

لوجستكس / توصيل: حوّل العمليات إلى خريطة حية

هذا النوع يستفيد أكثر من الحركة: السيارات، الـroutes، الطلبات المتأخرة، والمناطق الساخنة.

Warehouses

حجم المستودع بصريًا يعكس volume أو packages waiting.

Moving Vehicles

كل van يتحرك داخل الخريطة حسب آخر موقع أو route مبسط. اللون يوضح الحالة.

Click Vehicle

حسب - Driver / packages / ETA / route / status البيانات المسموح بها.

Hot Zones

Heatmap أو Beacons للمناطق اللي فيها ضغط أو delays.

Prompt

```
Build a 3D logistics operations world. Use moving vehicles for active routes, warehouses for package volume, color-coded zones for delays, and clickable delivery objects for ETA/status details. Add filters for date, depot, route, and driver. Animate only what communicates operational state.
```

لا تبالغ في الواقعية:

إذا البيانات ما فيها GPS لحظي، لا تخترع مواقع دقيقة. اعرض آخر status أو حركة رمزية واضحة بدل ما توهم المستخدم أنها live tracking.

UX

التفاعلات

وش يصير لما المستخدم يضغط؟

الـ3D يجذب الانتباه، لكن القيمة الحقيقية تأتي من التفاعل الذكي مع الأرقام.

التفاعل	وش يسوي	أفضل استخدام
Hover	Highlight + Quick KPI tooltip	نظرة سريعة بدون فتح panel
Click	Camera focus + detail panel	تفاصيل Revenue / Stock / Orders
Double Click	Drill-down	Category → Product → SKU
Color Shift	Green / amber / red	Health / risk / alert
Pulse	Subtle animation	شيء يحتاج انتباه الآن
Scale	Object grows/shrinks	مقارنة volume بين أقسام
Camera Fly-to	Smooth transition	التنقل بين أجزاء العالم
Timeline	Replay historic state	مبيعات يوم كامل أو أسبوع

قاعدة UX:

لا تجعل كل شيء يتحرك. الحركة المستمرة في كل مكان تخلي الـDashboard يبدو لعبة. استخدم Motion فقط عندما يحمل معنى: flow، alert، trend، أو activity.

LIVE

البيانات الحية

كيف تخلي الأرقام تتحدث بدون Refresh يدوي؟

الـDashboard ما يحتاج يسحب كل شيء كل ثانية. اختَر update strategy تناسب نوع البيانات وتكلفة الـAPI.

Webhooks

المصدر يرسل Event لما يحصل Payment/Order/ Inventory change. ممتاز للتحديثات المهمة.

Polling

كل 30-60 ثانية يسحب endpoint صغير. بسيط ومناسب لمعظم KPIs.

Manual Refresh

بعض التقارير ما تحتاج live. زر Refresh واضح أفضل من API calls بلا فائدة.

Cached Snapshot

Backend يجمع البيانات كل فترة ويخزن نسخة جاهزة للواجهة. أسرع وأرخص.

مهم:
اعرض دائمًا Last updated. إذا انقطع المصدر أو فشل الطلب، المستخدم لازم يعرف إن البيانات قديمة بدل ما يظنها live.

Prompt

```
Add a production-safe data refresh strategy. Use webhooks for important events when available, otherwise poll lightweight endpoints at a sensible interval. Cache expensive queries. Show last-updated time, loading state, stale-data warning, and retry behavior. Never block the 3D scene because one API call failed.
```

PERF

الأداء والنشر

خلّ الـ3D سريع، مو ثقيل

أجمل Dashboard يفشل إذا يأخذ وقت طويل أو يعلق على الجوال.

Lazy Load

حمّل المناطق الثقيلة فقط لما المستخدم يحتاجها.

Keep Geometry Light

استخدم low-poly objects لما التفاصيل ما تضيف قيمة. خفف textures والـshadows.

Separate UI from Canvas

Charts والجداول داخل HTML/CSS تكون أوضح وأسهل من Text داخل scene.

Mobile Fallback

على الأجهزة الضعيفة، اعرض 2D cards أو simplified scene بدل forcing full 3D.

Deploy

انشر الـfrontend على خدمة hosting مناسبة، والـbackend في بيئة تحفظ secrets بشكل آمن.

Mock Data Mode

خلّ المشروع يشتغل ببيانات تجريبية إذا connector غير متاح.

Performance Prompt

```
Audit this Three.js dashboard for performance. Target smooth interaction on a normal laptop and a simplified usable mobile experience. Reduce geometry, textures, shadows, draw calls, and unnecessary animation. Keep analytics panels outside the 3D canvas. Add a low-power fallback mode.
```

FIX

أخطاء شائعة

الأشياء اللي تخرب الفكرة بسرعة

إذا تجنبت هذه الأخطاء، النتيجة تكون أداة بزنس فعلية بدل Demo جميل فقط.

1. كل شيء يتحرك

الحل: Motion فقط للحالة، activity أو alerts.

2. API مباشرة في Browser

الحل: Backend/server layer + environment secrets.

3. 3D بدون سؤال تجاري

الحل: اربط كل object بقرار أو KPI.

4. تحميل بيانات كثيرة

الحل: aggregate قبل الواجهة، ولا ترسل كل السجلات للعميل.

5. ما فيه Empty/Error State

الحل: صمم ماذا يظهر إذا connector فشل أو ما فيه بيانات.

6. ما فيه تاريخ تحديث

الحل: Last updated + stale warning.

7. نسخة desktop فقط

الحل: mobile fallback أو simplified 2D cards.

8. تفاصيل 3D مبالغ فيها

الحل: performance أولًا. الهدف فهم البيانات.

أفضل طريقة للبداية:

ابنِ Version 1 ببيانات Mock، 5-7 Objects فقط، و3 KPIs لكل Object. بعد ما يثبت الـUX، اربط البيانات الحقيقية.

MIN MAP 15

CHEAT SHEET

الخطة كاملة في صفحة واحدة

هذه أسرع طريقة من فكرة إلى Dashboard شغال.

DATA .1	حدد المصدر: CSV / DB / CRM / Square / Stripe. اعمل Data Map قبل أي UI.
METRICS .2	اختر فقط KPIs المهمة: Revenue، Customers، Orders، Inventory، Alerts.
WORLD .3	حوّل البزنس إلى 3D World له منطق بصري واضح. كل KPI = Object له معنى.
THREE.JS .4	Scene + Camera + OrbitControls + Raycaster + Animation + UI overlay.
DATA LAYER .5	لا تربط secrets بالواجهة. Backend يرجع JSON normalized فقط.
INTERACT .6	Hover = quick KPI. Click = focus + detail panel. Alerts = color/pulse.
LIVE .7	Polling / Webhooks / Cache حسب الحاجة. اعرض Last updated.
TEST .8	جرّب أولًا empty state، API failure، mobile fallback، desktop، Mock data.
DEPLOY .9	Frontend سريع + backend آمن + monitoring + env secrets.

ابدأ صغير:

أول نسخة ممتازة = مشهد واحد، 5 metrics، 5 Objects، وclick panel واحد. لا تبدأ بمدينة كاملة.

مرجع تقني: Claude Connectors/MCP للوصول إلى الخدمات الخارجية، Stripe MCP لبيانات الدفع، Square APIs للمبيعات والمخزون، وThree.js للـ3D والـinteraction.

TAHSHR.AI

خلّ بيانات البزنس تُفهم بنظرة واحدة.

الفكرة مو إنك تسوي Dashboard شكله مجنون. الفكرة إنك تبني عالم بصري يخليك تشوف وش قاعد يصير في البزنس وتعرف وين تضغط مباشرة.

- اربط البيانات بشكل آمن.
- حوّل الـKPIs إلى Objects لها معنى.
- استخدم Three.js للحركة والتفاعل.
- خلّ Claude يبني الـworkflow والكود معك.
- ابدأ صغير وبعدها وسّع العالم.

تابعني للمزيد من أدلة AI
TAHSHR.AI@

للتواصل
info@taha-ai.com