

CLIENT GUIDE

Vibe-coded apps: a business owner's guide

AI makes it easy to build something that looks finished. Making it secure, lawful, affordable to run and connected to your business is the hard part.

Total Group International Ltd

Managed IT, cyber security and compliance

September 2026



AI tools make it easy to build something that looks finished. Making it secure, lawful, affordable to run and connected to your business is the hard part, and it is where most self-built apps come unstuck.

01 The honest picture

An AI tool can produce a working screen in an afternoon, and a hosting platform can put it online for a few pounds a month. It feels like 80% of the job is done. In our experience it is closer to 20%.

What you can see (screens, buttons, forms) is now the easy part. What you cannot see decides whether the app is safe to use with real customers: who can get in, where the data lives, what it costs at scale, and whether you can move it or connect it to anything else.

Unguided "vibe coding" means prompting AI to build and deploy a whole application with nobody reviewing the code line by line. It changes the person's role from builder to uncritical consumer of code they cannot read, and that creates failure modes traditional, human-reviewed engineering was designed to catch.

We have yet to see a well-built business application created by someone with no software development experience. That is not a criticism of the people building them. **The tools make the visible part effortless, so the invisible part never comes up until something goes wrong.**

Vibe coding is not AI-assisted engineering

AI is not the problem. How it is used is. Engineering leader Addy Osmani draws the line clearly: the same tools can produce a throwaway experiment or production-grade software, depending on the discipline around them.

	Vibe coding	AI-assisted engineering
Who is in control	The AI. Suggestions are accepted with little scrutiny	The engineer. AI speeds up routine work; people own the design
Planning	Straight to prompting, no specification	A written spec and design first, often drafted with AI
Review and testing	Skipped, or "it seems to work"	Every change reviewed and tested before release
Accountability	Unclear when something breaks	A named person signs off security and quality
Right for	Prototypes, hackathons, one-off experiments	Anything with real users, real data or real money

02 What the research shows

Independent research from 2025 puts numbers on what we see in client reviews. Newer models write code that works more often, but Veracode found no matching improvement in security: better AI does not remove the need for review.

Source	Finding	What it means for you
Veracode 2025 GenAI Code Security Report	45% of AI-generated code samples, across 100+ models, failed security tests with OWASP Top 10 flaws.	A high baseline risk of injection and authentication flaws reaching production
Wiz Research 2025	One in five organisations building on vibe-coding platforms were exposed to systemic risks such as open databases and exposed keys.	Wide-open databases and admin pages reachable by anyone on the internet
Apiiro June 2025	3 to 4 times more code, but 10 times more security findings. Privilege escalation paths rose 322%.	Review and testing capacity is overwhelmed: "fixing the typos but creating the timebombs"

03 Where vibe-coded apps fall down

The same six problems appear in almost every self-built app we review. Each is fixable, but fixing it after launch costs several times more than building it in at the start.

Area	What usually happens	What it costs you
Production costs	Hosting, databases, AI calls and email are billed by usage. A low entry price rises sharply with real users, and nobody set a spending limit.	Monthly bills that grow faster than revenue, or a surprise invoice after a busy week
Data security	Passwords and API keys sit in the code. Users can see each other's records by changing a number in the web address. Admin pages are unprotected.	A data breach you must report to the ICO within 72 hours, and to your customers
Data location	The app is deployed to the platform's default region, often the USA. Logs and backups hold personal data nobody knew about.	GDPR exposure, and failed customer or tender security questionnaires
Security and testing	No penetration test, no patching of the libraries the AI chose, no monitoring or alerts.	Problems found by attackers or customers, not by you
Portability	The app is tied to one platform's database, login system and deployment tools. Only the AI session that wrote it understands it.	Expensive or impossible to move, hand to a developer or sell
Connections	No proper link to your real data and tools: Microsoft 365, accounts, CRM, file storage. Data is copied by hand or duplicated.	Two versions of the truth, extra admin, and more copies of personal data to protect

04 The flaws we find most often

These are the specific weaknesses behind the figures above, grouped by where they bite.

Critical security flaws

Login checks in the browser

AI often puts access control in code that runs in the user's browser, where anyone can edit it and walk past the check.

Broken access control (IDOR)

Records are fetched by ID without checking the user may see them. Change the number in the address and you see another customer.

Exposed keys and secrets

Asking the AI to "connect Stripe" or "add OpenAI" often leaves the secret key in the front-end code or a public repository.

Injection

Database queries built by joining text together instead of safe parameters, opening the door to SQL and command injection.

Supply chain

Slopsquatting

AI invents package names that do not exist. Attackers register those names with malicious code, which installs the moment the app is built.

Unvetted dependencies

Every quick feature pulls in more libraries, each with its own dependencies and unpatched vulnerabilities nobody has looked at.

Maintainability

Nobody can debug it

When a serious bug appears at scale, the person who prompted it cannot fix it: they never wrote or read the logic.

Technical debt

AI optimises for "it works now", not for clean design. Each revision makes the code more brittle and harder to change.

Governance

Data residency

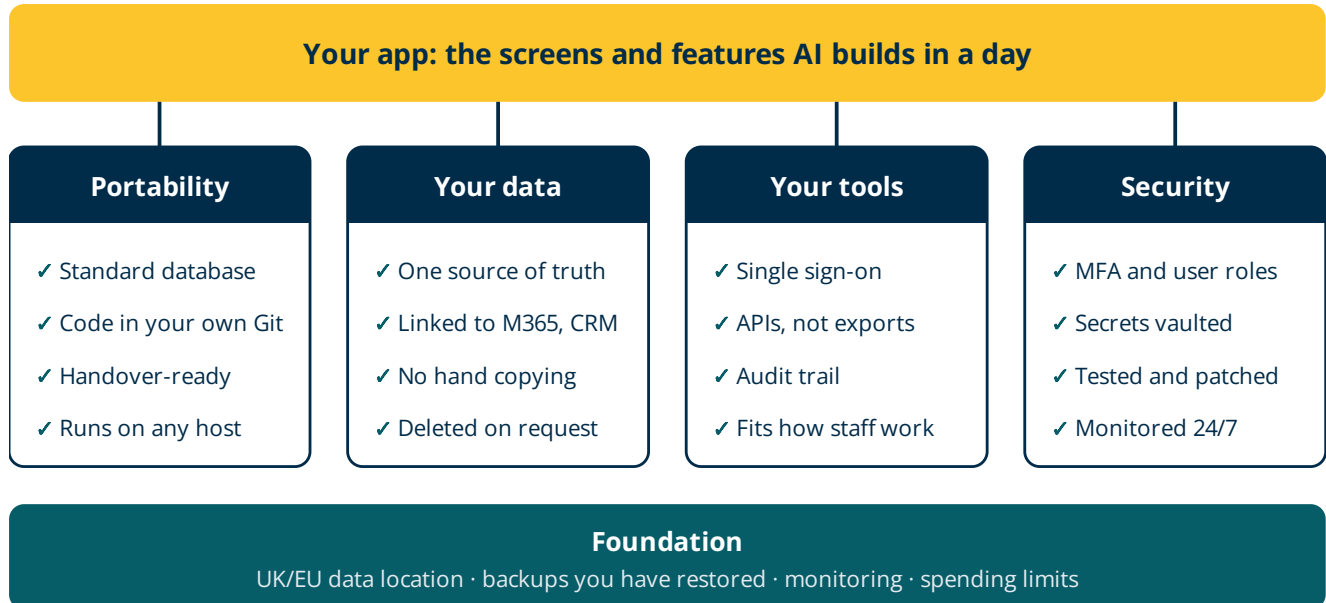
Prototyping tools default to shared public cloud, often in the US, which can breach GDPR (or HIPAA) the moment real data goes in.

Unclear IP ownership

Code produced wholly by models trained on open-source repositories has uncertain legal provenance, a risk if you ever sell, license or raise investment.

05 A firm chassis

Think of the screens as the bodywork and the foundations as the chassis. AI tools build excellent bodywork. The chassis has to be designed, and everything else is bolted to it.

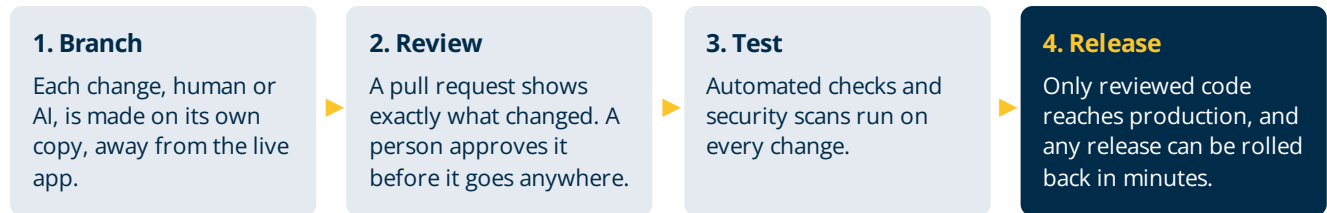


- **Portability:** a standard database and your code in a Git repository you own, so any developer or host can take it on.
- **Connections to your data:** the app reads and writes your real records (Microsoft 365, accounts, CRM, SharePoint) through proper links, rather than keeping its own copy.
- **Connections to your tools:** staff sign in with their existing work accounts, and other systems talk to the app through APIs with an audit trail.
- **Security:** multi-factor sign-in, roles so people only see what they should, keys kept out of the code, and regular testing and patching.
- **Foundation:** data held in the UK or EU, backups you have restored at least once, monitoring and alerts, and spending caps on every usage-billed service.

Get the chassis right first and the screens can change as often as you like. **Get it wrong and every new feature makes the problem bigger.**

Version control: safe change management with Git

Git records every change to the code: what changed, when, who (or which AI session) made it, and why. It is how professional teams change software without breaking it, and it is the single cheapest safeguard a self-built app can have.



Without Git, an AI tool that "fixes" one thing and silently breaks three others leaves you with no way to see what happened or get back to the version that worked.

06 The AI layer needs its own assessment

If your app calls an AI model (to chat, summarise, classify or generate), that model is a separate service handling your data. It needs its own privacy and cost assessment, just as the hosting does.

Question	Why it matters	What good looks like
Which models, from which provider?	Each AI provider is another processor with its own terms, locations and sub-processors	Named models and providers on your record of processing, each with a signed DPA
Public or private model?	A public consumer service and a private deployment carry very different risks	Business or API terms as a minimum; a private or on-premise model for sensitive data
Is your data used for training?	Consumer tiers may use prompts to improve models; business and API tiers usually do not, but check	Training switched off in writing, and short or zero data retention
Where is it served (geo)?	Prompts often contain personal data. Where they are processed and logged is a GDPR question	UK or EU processing where offered, or a documented transfer mechanism
Does a frontier model run everything?	Sending every request to the largest model is slow and expensive, and often unnecessary	Small, cheap models for routine tasks; the frontier model only where it earns its cost
What does it cost at scale?	AI is billed per token. Cost grows with users x requests x prompt size, and agents can loop	A cost model per user per month, hard spending caps, and alerts before the bill arrives

A frontier model can build the whole app and then answer every user request. **That is the most expensive way to run it, and the cost scales with every user you add.** Design the AI layer so the right-sized model does each job, and know the cost per user before you launch.

07 Is my app GDPR compliant?

A compliant hosting platform does not make your app compliant. The platform secures its servers; you are responsible for everything you build and store on them. Under UK GDPR you are the **controller** and the platform is your **processor**.

As a worked example, here is what we found for Railway, a platform several of our clients use:

Question	Railway's answer
Who are they?	Railway Corporation, a US company based in San Francisco
Independent audits	SOC 2 Type II and SOC 3, with reports in their trust centre
GDPR contract	Self-service Data Processing Agreement, with EU standard contractual clauses and the UK Addendum for transfers
Where data can live	US West, US East, EU West (Amsterdam) or Singapore. New projects use your account's default region
Primary processing	United States, according to their DPA
Breach notice	"Without undue delay", with no fixed number of hours

That is a reasonable platform. It still leaves the app's owner to:

1. Deploy every service and database to the EU West region, and check none are left in the US.
2. Sign the platform's DPA, and list every other service that touches personal data (AI APIs, email, logins, analytics), each with its own DPA and location.
3. Publish a privacy notice, keep a record of processing, and complete a DPIA where the data is sensitive.
4. Be able to find, export and delete one person's data when they ask.
5. Secure the app itself: sections 03 and 04 show where breaches actually come from.

08 Before real customers or real data go in

If you cannot tick every line, the app is not ready for personal data.

- ☐ Every service and database is in a UK or EU region, and I can show a screenshot of it
- ☐ I have signed the platform's DPA and listed every other service that sees personal data
- ☐ No passwords or API keys are in the code or visible in the browser
- ☐ Each user can only see their own records, tested by trying to open someone else's
- ☐ Staff and admins sign in with multi-factor authentication
- ☐ Backups run automatically, and I have restored one successfully
- ☐ Spending limits and alerts are set on hosting, database and AI services
- ☐ The code is in a Git repository my business owns, not only in the AI tool
- ☐ A developer who has never seen it could run it on another host
- ☐ It connects to our real systems through proper links, not copies or exports
- ☐ It has had an independent security test, and the findings are fixed
- ☐ We have a privacy notice, a record of processing and a plan for data requests
- ☐ Every change goes through Git, is reviewed before release, and can be rolled back
- ☐ Every AI model the app uses has had a privacy assessment: provider, training, location, retention
- ☐ We know the AI cost per user per month, and hard spending caps are in place

09 How Total Group can help

We will not tell you to throw your work away. We will tell you honestly what is safe, what needs fixing and whether it is worth fixing, then help you get there in three steps.

1 Check where you are

Three AI Readiness Assessments, so nothing is judged in isolation:

Application

The app you are building: code, logins, data handling, AI models and cost

Platform

Where you are building it: hosting, region, contracts, backups and portability

Environment

Your entire IT: Microsoft 365, devices, identity, network and the data the app connects to

2 Mind the gap

That's where cost, risk and viability reside.

We measure the gap between where you are and recognised standards: technical controls such as the **OWASP** Top 10 and **SOC 2**, and legal duties such as **UK GDPR**. You get independent advice on deploying and securing the hardware and software to close it, and an honest view on whether to fix, rebuild or stop.

3 Secure, monitor and maintain, in perpetuity

An app is never finished. We make sure it stays secure, patched, monitored and backed up for as long as you run it, with 24/7 monitoring, continuous testing and a team that is accountable when something changes.

Sometimes the honest answer is that the prototype has done its job: it proved the idea. **Then the cheapest route is to rebuild it properly on a firm chassis, using the prototype as the specification.**

Talk to a local team

01727 881 224 · hello@totalgroup.co.uk

totalgroup.co.uk · Hertfordshire, the Home Counties and London

