Article

Apr 2025

Bespoke Software vs Low-Code – why not both?

We use a broad spectrum of technology to build software solutions. While many of our applications remain bespoke to foster innovation and create cutting-edge consumer-facing SaaS products, we are also embracing low-code approaches. So how do we decide on the best approach? Let's take a look.



Mike Dent
Head of Digital Solutions
Email mike.dent@waterstons.com

When to choose low-code

Building on what you have

You may already have Microsoft 365 licenses which include the ability to use standard features of the Power Platform. If you have people in your organisation that are keen to create their own applications but not trained in software development (we use the term "citizen developers"), and the governance to ensure there's no shadow IT, low code solutions are worth exploring. Existing Microsoft licenses can be used as an out of the box authentication mechanism, particularly if you're looking to build on top of permissions you already have in your M365 applications. A popular requirement we see is to build front-end interfaces on top of SharePoint libraries, which saves building a whole document management system from scratch.

Cost-saving

If you can build a low-code application with the same features as a completely bespoke system, it will always be cheaper due to the out of the box functionality and templating available in low-code platforms. We'll discuss more about the 'if you can' when we talk about when to go bespoke. This also links closely to speed of deployment, where we've spun up simple applications in days that would have taken weeks with a bespoke approach.

Pre-built integrations and connectors

Most low-code platforms include in-built integrations connectors to interface with popular SaaS products such as the Microsoft 365 suite, Cloud providers, SAP, Sage and other vendors. The power here is we can make heavy use of built-in data-mapping, data-transformation and connectors - something that takes a lot of effort with a bespoke approach and doesn't require knowledge of software development.

When to go Bespoke

Scale

Many low-code platforms have throttling limits and expectations about usage. For instance, with Power Apps, there's a hard limit on the resources dedicated to your app and there's not much you can do other than expect throttling or performance issues. With a bespoke app and infrastructure, you can choose how little or as much resources as you need depending on your application's needs.

Other limitations on low-code platforms might involve licensing restrictions, platform limitations, request restrictions (e.g. limited to 2000 item retrieval from a database as in the case of Power Apps).

UI / UX Customisation

Some low-code platforms like PowerApps come with some really slick templates and responsive design as standard, meaning you can use the apps on a mobile device without much thought during development. Some low-code platforms look like they were used to design Windows 3.1. Either way, if your brand depends on how the experience of the application is used (especially for consumer facing products) nothing beats a unique and bespoke design tailored to the user's journey through the application.

Complexity - the unknown unknowns

If you're replacing a system like-for-like or have the exact requirements for the application (people rarely do!), a low-code solution is the way to go. If you're working in a truly agile way, and using user feedback to guide your requirements, there may be a point of no return where you must create something bespoke to deliver that killer feature. We would rarely use a low-code approach to the R&D projects or proof-of-concepts that we build as there are too many unknowns if it's something that hasn't been done before. For instance, integrating with something obscure or complex will usually require custom code.

Why not both?

Choosing a hybrid model can sometimes be the best of both worlds. A low-code front end that can call some complex bespoke logic behind the scenes can de-risk a project and keep down cost, if you have the right people. All good low-code platforms allow you to hook up to custom APIs (logic from other systems), and by writing these custom APIs we see a solution that's far less complex than building low-code workflows to perform the same function. If you can outsource the heavy lifting (e.g. a complex database call) to a process outside the platform, like an API, you can tailor the resourcing power needed on those APIs rather than the no code platform.

Choose a hybrid model when:

- The user interface doesn't need to be complex, but the logic of the application does
- The authentication doesn't need to be complex (i.e. only available to internal users)
- You're going to be restricted by resources of the low-code ecosystem but still want to leverage the benefits of low-code UI

Conclusion

There are good reasons to consider each of these approaches to building software products and applications. If you're considering building a new application, come and have a chat with us so we can share our experience so you can decide which approach you might want to take.

Email our Head of Digital Solutions Mike Dent at mike.dent@waterstons.com

To read more about bespoke software and its benefits take a look at this article What is bespoke software development and how does it add value.