EPISODE 1014

[INTRODUCTION]

[00:00:00] JM: Low-code tools can be used to build an increasing number of applications. Knowledge workers within a large corporation can use these low-code tools to augment their usage of simple tools, like spreadsheets. Entrepreneurs can use low-code tools to start businesses without even knowing how to write any code. Modern low-code tools have benefited from steady improvements in cloud infrastructure, frontend frameworks like ReactJS and browser technology, such as the V8 JavaScript engine. These building blocks like V8 and React and AWS, they have been what has allowed us to get to low-code products such as WebFlow and Bubble and Retool and Airtable, and these low-code tools truly are a new layer of abstraction and they're supported by a broad selection of APIs, like Stipe and Twilio and Zapier that give them quite rich functionality. What can you actually do with these tools?

Ben Tossell runs Makerpad, a site devoted to low-code and no-code applications. Makerpad describes how to use these tools to design sophisticated applications that do not require you to write code, but they do require a different kind of software engineering. To create applications intelligently with low-code tools, you need to know how these tools fit together and you need to be willing to persist through a process of iteration and debugging that is similar to traditional software engineering. Ben joins the show to talk about his experience building low-code applications, the use cases for low-code tools and how he believes they will impact the future of software.

[SPONSOR MESSAGE]

[00:01:54] JM: DigitalOcean makes infrastructure simple. I continue to use DigitalOcean because of the low friction and attention to user experience. DigitalOcean has kept the experience simple and I can spin up a server in less than a minute and get high quality performance for a low price. For an application that needs to scale, DigitalOcean has CPU optimized droplets, memory optimized droplets, managed databases, managed Kubernetes and many more products. DigitalOcean has the flexibility to choose the right instance for the right workload and he could mix-and-match different configurations of CPU and RAM.

If you get stuck, DigitalOcean has thousands of high-quality tutorials, responsive Q&A forums and a customer team who treats customers respectfully. DigitalOcean lets developers focus on what they are building. Visit do.co/sedaily and receive $100 in credit over 60 days. That $100 can be put towards hosting or infrastructure and that includes managed databases, a managed Kubernetes service and more.

If you want to get started with Kubernetes, DigitalOcean is a great place to go. You can use your $100 to start building your distributed system and you can get that $100 in credit for free at do.co/sedaily.

Thank you to DigitalOcean for being a sponsor of Software Engineering Daily.

[INTERVIEW]

[00:03:28] JM: Ben Tossell, welcome to Software Engineering Daily

[00:03:30] BT: Thanks for having me.

[00:03:32] JM: You've been studying and writing about the low-code ecosystem for a while and I'd like to start with a discussion of the types of prototypical users that might interact with low-code tools starting with the modern corporate knowledge worker. I think of knowledge workers on a spectrum. On one end, you have programmers who are very technical and can build whatever they want to. On the other end, you have spreadsheet people who might have a domain expertise in certain business aspects, but they're not programmers. For a longtime, there was not much in between the level of technical expertise between programmer and spreadsheet person, but today there's a variety of tools in between the programming languages and the spreadsheets. Can you tell me about how you see low-code tools impacting the job of the corporate knowledge worker?

[00:04:25] BT: Well, in terms of the spreadsheet, spreadsheet person you're talking about, I guess there's more of when something happens in a spreadsheet, something else happens outside of that spreadsheet. For example, we've got like an expert marketplace, I make it that people come in through a spreadsheet, which is we use Airtable, and then when certain things are met or when I'll go in and click approve or something, then Zapier will fire, then emails will fire based off of that. I think things like that are very sort of basic spreadsheet. You can add quick automations too like that, which I would never have known.

I was a spreadsheet person before. I was a social media analyst. A few years ago. Well, several years ago now, and all of it was copy paste this data to this data then send it off to a team in the Philippines who did the analysis and automation stuff for us, but now it seems like it's sort of at your fingertips whenever you need it yourself.

[00:05:25] JM: What are some other workflows that a knowledge worker might be able to perform with low-code tools?

[00:05:33] BT: We've seen things like applications systems. So, Lambda School for example, they built a bunch of their backend stuff without code and a big part of their application process was built on top of Airtable, Typeform, Retool and some others and we see a lot of people who build lead generation flows that at some point use some spreadsheets. There are CRMs, there are applicant tracking systems. There're tons of these things where there's actual like big products built behind them, but you can just really make the simple version of, "Okay. Well, I just need like a Kanban style board," and you can make really simple version of those yourself.

[00:06:19] JM: Do you have a sense of how these tools will impact corporations in the limit? If you zoom out 5 or 10 years, what are the things that we do today in corporations on a technical level that will seem crazy once these low-code tools are proliferated?

[00:06:38] BT: Well, I think there's a big – I mean, there are huge amounts of repetitive work, monotonous work that people have to do every day, whether it's, "Okay, every time an email comes in with a resume attached, I then download that resume and then going at that to Dropbox myself," whereas you can have that automated without even touching anything.

There's things like the CRM and applicant tracking like I mentioned that I think automations just work off of, "Okay, if someone has moved from one column to the next, then have this happen, like an automatic projection email or app organizing the new call." There are just tons of things in automation that I think people do a lot of busy work. A lot of things, spreadsheet-based, copying data from one to another, pulling analysis and things. We can actually do a lot of that to sort of – Google Data Studio, for example. There's a ton and I haven't really been in the workforce for a long time actually. I just see a lot of these automations that people build on top of the no-code, low-code space and what they're sort of telling me.

[00:07:49] JM: Yeah, the type of people that you seem to cover more or focus more on is the low-code entrepreneur, and this is someone who in the past might have built a business on WordPress, but today they have a much wider variety of tools. Maybe they still use WordPress, but they are also using certain low-code tools to build their platforms. Who are the types of people that are building fully-fledged businesses without code? Can you tell me about some of the prototypical types of users who are low-code entrepreneurs?

[00:08:30] BT: Yeah. I think we have fallen into that type of user, because that's the type of user who's been waiting for the no-code space to be sort of coming out for them. I think there's a lot of productized service type creators out there who are leveraging automations and workflows that have them do a productize service app a larger scale where they probably couldn't have done it before or would have to hire multiple VAs or other people to deliver that service. But we see a ton of people looking at the sort of marketplace level too.

There's a few that we've covered that one was a marketplace for backyard homes, so modular homes. You can buy different sizes, shapes, costs, sort of look in just as you would imagine a marketplace would look like. It looks exactly the same. It was built in WebFlow. Yeah, tons of people doing like a surfboard rental marketplace that we've seen, things like that.

[00:09:33] JM: In your personal background, you've worked in compliance and finance. You've done some social media management. You've been a community manager and your work is the kind of thing that is not easily categorized. My sense is that you've been – You spent the 5 or 6 years of your career trying a lot of different things, but fundamentally, you've been in front of a computer understanding the internet from a unique point of view. It seems like there's a lot of people who fit that description. They spend some time in front of the internet. They get a certain perspective. Maybe they're not programmers, but they see an entrepreneurial opportunity. This seems to be in many cases the type of person that becomes one of these low-code entrepreneurs.

[00:10:28] BT: I think, for me, I always thought I wanted to be an entrepreneur, and I thought of it in terms of "Oh, yeah. One day I'll have a huge company with hundreds of employees and millions in venture capital investment." But actually through the process of going through those things, I was trying up lots of different ideas. But I knew that if I tried to meant to code and I was building some sort of marketplace app, it would take me months and months and months to get a very basic crappy version of a marketplace.

If I want to test this idea and if this is going to have any legs, I'd rather do it in a weekend. Is there a way I can sort of make the 80% version look and feel like an actual marketplace with some of the tools available to me? Then that meant that I could test a lot of things a lot quicker, but I didn't actually realize the thing I would end up doing was the actual teaching people around the no-code space.

[00:11:31] JM: The people that you teach, the tutorials that you make and the people who consume the low-code tutorials, are these purely people who have no experience programming or are there some programmers who have switched over to being low-code users?

[00:11:48] BT: Yeah, our quarter of our community is actually programmers. When it's not the programmers who are sort of being a bit sensitive around this space and just arguing back, I think there's a big growing community of programmers who are actually seeing the value of testing using no-code too because what programmers build a lot of the same infrastructure when they have to do a new project, much like you would have to in any case.

If no-code gets you there quicker and gets you to a validation point of paying customers, then why not do it in an hour rather than 20 hours or whatever it is? I think it's the just case of being productive and taking the quickest path to that validation and the programmers who see that in the community are some of our biggest advocates.

[00:12:41] JM: Do you have any sense of the rate of user growth of the low-code ecosystem?

[00:12:47] BT: Not by numbers specifically. I mean, Makerpad has become very popular. I was sort of doing it before it was trendy. I think a lot of people now are just sort of catching on with the no-code space. I think 2019 was really the year that it caught on. I remember WebFlow was a tool for designers primarily. Then when they raised their series A and had a little website refresh on their Hero, it was a sense of no-code quite or break the code barrier I think was the term.

Companies of that with big, big sort of presence in this space have definitely helped popularized the movement and more and more people sort of get it. But I'm a bit disappointed actually, the term is no-code. It's a bit unfortunate, but that's what's caught on. I mean, now it's probably too late to be able to change it. That's sort of what we're stuck with.

[00:13:42] JM: Why did it become popular last year? Was there some kind of inflection point that caused no-code tools to start – Or low-code tools, whatever your preferred description in? Was there some kind of inflection point that caused these things to start working?

[00:13:58] BT: I think it was just technically more possible than it was before. It was much more than build your website in Squarespace and more towards you can build an application without needing to know how to code. There are different various levels of that. Our site is built in WebFlow, Airtable, Zapier and MemberStack and we have things where people could have their own profile. They can mark tutorials as completed. They can sort of say what tools they use, and that gets automatically added to their profile. But it's all built with sort of no-code, and that wasn't really possible or as accessible to someone like me a couple of years ago. I think it's definitely helped the possibilities of definite grow and more and more tools are coming out to help people who aren't technical to build some of these things.

[00:14:56] JM: My experience creating software applications. When get to a significant size, like at least as big MakerPad has gotten in terms of the amount of content you have on there. The workflow often becomes a little complicated, becomes convoluted, but you have these tools and you have some best practices to keep it coherent, to keep an understanding of how things work together.

I have a sense for how that works in the software engineering world not so much in the low-code world. When you're piecing together a site that feels like a fully-fledged platform. If you go to MakerPad, it feels like a fully-fledged site that is engineered by a team of software engineers. When you look at your stack and you have AirTable here and WebFlow there and Zapier in some other place, does it feel like they're working together harmoniously, or do you feel like you're jumbling together something in a way that these tools are not talking to each other as fluidly as you would like?

[00:16:04] BT: I think they're getting a lot better at speaking to each other more fluidly. That's for sure. I think it also – It's not necessarily as simple as, "Okay, just connect these four in a certain way and it will work in the way that you think." MakerPad especially has evolved from what it originally was, and I think originally it was just I would send you an automatic email based on you go to Typeform to pay and then you'd get an email that would give you a password protected website in a page in Webflow and that will give you the password.

From there, we really sort of increased the capabilities of what we've done. Yeah, when we're setting up these systems, like for instance the profiles recently, there's definitely quite a lot of testing to sort of get you to a plug where it's fluid and log program and there's a trial and error. There's the, "Okay. Now it's sort of live or it's semi-life to a few users who we onboard as the first testers." They'll put it through its paces and then that will throw errors or bugs in our Zapier sort of task history. Go through and sort of make sure we fix each and every one of those.

The process is actually – I can't speak from experience. But I think the process is actually quite similar in how you create like your version. You test it with some users. You fix bugs and then you just try and add more things to that. It's quite similar, but there's – Yeah, they do work harmoniously once you get to a certain stage and you sort of know what you do and you've crossed a lot of these things off. But it's not quite as simple as point and click and then they're connected.

[SPONSOR MESSAGE]

[00:18:01] JM: I remember the days when I went to an office. Every day, so much of my time was spent in commute. Once I was at the office, I had to spend time going to meeting rooms and walking to lunch and there were so many ways in which office work takes away your ability to be productive. That's why remote work is awesome. Remote work is more productive. It allows you to work anywhere. It allows you to be with your cats. I'm looking at my cats right now. But there's a reason why people still work fulltime in offices. Remote work can be isolating. That's why remote workers join an organization like X-Team.

X-Team is a community for developers. When you join X-Team, you join a community that will support you while allowing you to remain independent, and X-Team will help you find work that you love for some of the top companies in the world. X-Team is trusted by companies like Twitter, Coinbase and Riot Games.
Go to x-team.com/sedaily to find out about X-Team and apply to join the company. If you use that link, X-Team that you came from listening to Software Engineering Daily, and that would mean that you listen to a podcast about software engineering in your spare time, which is a great sign, or maybe you're in office listening to Software Engineering Daily. If that's the case, maybe you should check out x-team.com/sedaily and apply to work remotely for X-Team.

At X-Team, you can work from anywhere and experience a futuristic culture. Actually, I don't even know if I should be saying you work for X-Team. It might be more like you work with X-Team, because you become part of the community rather than working for X-Team, and you work for different companies. You work for Twitter, or Coinbase, or some other top company that has an interesting engineering stack, except that you work remotely.

X-Team is a great option for someone who wants to work anywhere with top companies maintaining your independence, not tying yourself to an extremely long work engagement, which is the norm with these in-person companies, and you can check it out by going to x-team.com/sedaily.

Thanks to X-Team for being a sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

[00:20:36] JM: It's so interesting to hear you talk about this, because I think your site is one of the more – It's definitely on the more complex side of a low-code application that's been written.

There's a term in software engineering called technical debt. Basically the idea is as your application gets built overtime, you accumulate these little pieces of debt. Meaning maybe you're trying to go fast because you're usually building some kind of business and you can't just make everything polished and perfect. You have to grow the business. You have to work on the things that are going to grow the business immediately. Are there places where MakerPad has developed technical debt? Kind of things that you look at and you're like, "This is engineered poorly, or this is causing this bug to repeatedly come up." Does that word technical debt bring to mind any particular things that have developed overtime in MakerPad?

[00:21:35] BT: Yeah, definitely. I think it may even be more prevalent in what I'm building, because we can build things so quickly and new features and new capabilities so quickly that sometimes makes previous ones obsolete or they need a lot of updating. We've had cases on the site where there's like a members area. We had our sort of experts listed, and that was previously a WebFlow form, go to Zapier, to Airtable, and then back to WebFlow CMS item and anyone can get themselves listed.

But now with us rolling out profiles, it's now a much more complex way of us doing that. So then now that profile section is the previous — sorry, profile section is no longer needed. So we got to go back and make sure the CMS collections are deleted and they're always linked to certain other pages. You've got to go and remove all those. It does take a bit of time to go and reverse some of the mess you've made, wash your files sort of. I don't think – It's not just for programmers. So we fill up in too.

[00:22:46] JM: Software engineering historically had this taxonomy of tools where some of them are on the backend. Some of them are on the frontend. Is there a similar way of thinking about these tools in the low-code ecosystem? Are there frontend and backend low-code tools or do you just think of them as some other paradigm?

[00:23:11] BT: No. I view them as sort of frontend, backend style, but there are some crossovers. I would see WebFlow as its current state is more of the frontend. That's how you make the thing look how it looks. Then Airtable would be your sort of day-to-day. Zapier would be your connector. But then there's tools like Bubble, which is almost like an all-in-one solution and it allows you to do a lot of logic-based things, if/this/then that type of stuff. It gets more complex.

I think it depends on what you know and how you like to build and sort of the visual element of creating something. But there's tons of different types of tools for each, so the connectors. There's also like Parabola and a bunch of others. There are many tools for each of these pieces, but I think when I think of building something or I'm building a new thing, I often look at, "Okay. Will it need a front, a back, a connector database stuff? What do those things look like?"

They did a workshop in the co-code conf where I actually just went through sort of the stages of, "Okay. This is how you think about the front, the back, and this is sort of mobile app specific." If you apply that sort of mobile app and if you could do all-in-ones, you've got Adalo, and Boundless, and Bubble." There are lots of different options for each case.

[00:24:37] JM: Is Airtable the most frequently used backend setup?

[00:24:42] BT: I think so. It's got to be Airtable or just using Google Sheets, I think. But the views and everything else that Airtable sort of brings with it has really helped, but it tends more coming out in the sort of spreadsheet database section of there's Retool, there's Clay. There's a bunch of others which have much more complex formulas, buttons and things you can do there. We're really interested in playing with those.

[00:25:12] JM: We just did a show about Parabola. That tool is mostly used for it seems like Dataflows and this kind of back office jobs, like maybe you want every day or every once an hour go through your customer list and send all your customers who have made a purchase of $50 or more an email, you can do these kind of batch processing Dataflow jobs with Parabola. Retool, I haven't done as much covereage of, but I understand it's useful for building reusable backend tools that could be usable for like a data analyst or an operations analyst. What kinds of applications have you seen built with Retool and Parabola?

[00:26:01] BT: Retool, I think in this case was Lambda School again. We did it in a few of them. That's how I can reference them quite easily. But they used Retool for a lot of their profiles and people – Students going through their boot camp and adding sort of check-ins and progressing through that. I think they were using Retool as their database for all the students.

We've just started playing with Retool. We've just partnered with them and we're interested in diving and I know there's a bunch of different things there. Parabola, there's a VC called Josh from 5 Capital who did an analysis on his Chrome browsing history. He made this huge flow to see how many times a day or how many times a week he would open Twitter and his email and things like that. That was quite an interesting flow and quite a complex one.

We've seen – There's a company. I think it's called Rowpoint who built – Basically their referral product on Parabola and they got to sort of 5 million in annual recurring revenue just using Parabola as their actual product. Customers would come in and they just assume they were using software as you normally would, and I think on the back, it was just very complex Parabola flows with other datas. We did a story on that too, which was really interesting.

[00:27:27] JM: That anecdote there that somebody built a $5 million annual recurring revenue business, that was heavily inspired by this unique tool, Parabola. That kind of story is the sort of thing that makes me convinced that this ecosystem is quite signifi cant, because you think about the trajectory of the software entrepreneur and how it has look for maybe the last 5 or 6 years, typically you go to a coding boot camp or an online education system, Treehouse or something, or maybe if you went to a university, you studied computer science, then it's a little bit easier. But then you've had to invest four years in the computer science.

But in all these cases, you'd have to learn to program. Then you've had to learn business on top of that. Then you'd have to learn some domain expertise on top of that. This seems like really important, because if you just have some domain expertise and you have a business idea, it's much easier to take that idea and generate a business out of it.

[00:28:41] BT: Yeah, I think if can sort of argue with people until I'm blue in the face of, "Oh, yeah. Low-code, no-code is just for prototypes." When I hear these stories, like what Lambda School got a 30 million in venture funding with 3,000 concurrent students on a coding boot camp by using no-code tools as their backend.

That's been one of my favorite examples. But, yeah, we see companies who make 5 million a year who built on no-code and I just think that no-code enables people to test their ideas. I say test, because it is that until you validated it and until you're building it up, there's no reason why – If MakerPad was just a tutorial platform and we bot up to tens of thousands of users, students a year and we just focused on tutorials, there's no reason why we couldn't be a multimillion dollar recurring business also built on top of WebFlow, Airtable and Zapier.

It was just an obvious choice for me when I couldn't code and I tried a few times. I mean, it just wasn't the right thing. I just didn't click with it and I didn't get it in terms of writing those functions out and doing it and my brain just didn't work in that way. But where I could see this WebFlow thing or this Airtable thing and Zapier thing and linking them together, I sort of got it and it just enabled me to build those things so much quicker and test them so much quicker and asked for payment straightaway. If no one pays and a horrible feedback, then I'm also less emotionally invested in it, right? I'm not 12 months, 18 month down the rabbit hole of, "Okay. Now, I've got to make this thing work, because I've made 10,000 lines of codes doing this and it's like everything I've ever done." If you can build it in an hour, if everyone hates it and it's really bad and no one wants to use it, you can also throw it away a lot quicker, which I quite like the analogy of as well.

[00:30:52] JM: When you're teaching people to use low-code tools, do you see people hit the same kinds of roadblocks that they hit in learning to code? I think your story about trying to learn to code a couple of times and feeling frustrated and giving up, that is not an uncommon story. Even myself, I kind of left the world of writing software engineering because it kind of felt like I didn't have much leverage. Even though I could write an app end-to-end, it took so long and I just wanted to have more leverage in my actions and these kinds of tools definitely provide more leverage. But what kinds of roadblocks – Since you're in the business of teaching people these things, what kinds of roadblocks do people hit? Do they still hit the technical roadblocks that cause them to quit?

[00:31:47] BT: I think it's the exact same process, because it's like doing anything. If you're learning a language, not a technical language, as in like French or something, there's always something you not – You might think of, "That word, I just can't get right or I cannot figure out what the right tense is of this sentence." There's always – I think in learning in general, there's always a case of it's never always just going to be straight forward and work for you in the way that a textbook or a tutorial might show you. There's always something.

I think with learning around the no-code space, there's almost those quicker wins which help keep you on that path of, "Oh! Well, I managed to build this whole app in an hour and a half, but I just can't get this image URL thing to work, but I've got 90% of the way where I have gotten one month of coding before before I wanted to rip my hair out." It's like the different types of wins, I think, and I think that it's just a process of teaching and a process of learning that you bound to hit roadblocks. It's just a way of being able to solve those, and I think we've got a very helpful community much like – In software engineering, there's very helpful communities on the web. We're just looking at ways to nurture that and make sure that everyone can get to where they're actually trying to go rather than giving up and saying, "Oh, no. I'm going to go back to coding. This is not for me."

[00:33:17] JM: What kinds of scalability issues have you seen with low-code tools?

[00:33:22] BT: I've not seen much, to be honest, because I think as I'm in the process of teaching people, it's often a lot of younger ideas or projects that are being built. For us, we've got almost 1,000 paying users and thousands a week that come on the site and we've been fine. We have no hiccups. I keep on using Lambda, but they had 3,000 concurrent students. I've not heard of a big issue here. I think that the tools are obviously becoming better and better each week, each month, and we have to see that. So I don't want to pretend that there's no limit and there's no ceiling, because there definitely is, but I don't know where that ceiling is yet.

Also, it usually the case that when people are asking me, "Well, why should I use no-code over [inaudible 00:34:17] code? Aren't I going to hit – Wouldn't I hit a limit if I have 10,000 paying customers on my WebFlow site?

I think, well, if you haven't got 10 customers, I'm not sure that's the thing you should be worried about right now. Maybe you can get to a thousand customers a month, a thousand customers a day perhaps. Then if you're at that stage and you're hitting those limits, you validated it. You've probably got a team. You've got an actual business on your hands. Then it might be time to graduate on to coding it yourself or hiring people to code a platform for you. But we don't know how much better these no-code tools are going to get also. I know that people talk about sort of being attached to a platform.

I look at that in the way that the things that I have created, they can all be exported or downloaded to like the basic level of a spreadsheet or is like all our tutorials, the video is saved on our files, on YouTube, on somewhere. All the text goes with it. WebFlow, we can export the whole site if we wanted to. It may look like a mess, because I've made it, but we still got all up there. All our Airtable data, if Airtable went down, we could just move all that to a normal spreadsheet. Things at Zapier, obviously, we rely on. So then we'd have to look at different ways to build those integrations, but there are still many options out there of these connector tools that help you do those things.

[SPONSOR MESSAGE]

[00:36:03] JM: Logi Analytics believes that any product manager should be successful. Every developer should be proud of their creation and they believe that every application should provide users with value. Applications are the face of your business today and it's how people interact with you. Logi can help you provide your users the best experience possible. Logi's embedded analytics platform makes it possible to create, update and brand your analytics so that they seamlessly integrate with your application.

Visit logianalytics.com/sedaily and see what's possible with Logi today. You can use Logi to create dashboards that fit into your application and give your users the analytics that they're looking for. Go to logianalytics.com/sedaily.

[INTERVIEW CONTINUED]

[00:37:01] JM: A sophisticated evolution of the historical back and forth between closed and open because it's not the degree of closeness that you would have with Windows or I guess some people would say Facebook. You can't really export your Facebook system to some other system. The fact that all of these different tools to proprietary, but you're piecing together a multitude of the different tools. In each tool category, there is some interchangeability. There's a little bit less platform risk when it comes to each individual tool.

To give some more light on just the variety of tools, there's obviously these connectors, like Zapier, there's the frontends, like WebFlow. There are a bunch of different categories. One category we haven't really explored is the fully-fledged platform sites, like you have Shopify rebuilding an ecommerce site, but it really gives you a whole ecommerce backend. You also have – There are other big platforms. I saw one on your site called Sharetribe, which is an out-of-the-box platform for building a marketplace, which is kind of amazing just the idea that you could have a system for building a marketplace, entire platform of platform building.

Are there any other large platforms that people are using to build fully-fledged no code sites? Things like Shopify or Sharetribe? Are there any other large thick sites in this category?

[00:38:46] BT: I don't know in terms of like having the functionality out-of-the-box, but tools like Bubble and Boundless and Adelo have the ability to let you have sort of, "Okay, here's your site. Here are your user profiles. Here is your signup pages. This connects in this way. Once you sign up, then you can go and access these things. Once you pay, you can do this."

I've seen people build like on-demand Uber style clones with Bubble. I've seen just like fully functioning job boards with Boundless. One of our tutorials is a Patreon clone built with Adelo. So you can have your own community page. People could donate to your page and things like that. These have more fluidity in what you can build, but then you've got some of these core components that let you have that login functionality, which is one of those difficult things in no-code or has like payment functions or things like that.

[00:39:53] JM: Can you tell me about some low-code tool that is extremely niche but also successful? I want to give the listeners a perspective on just how many tools there are in this ecosystem.

[00:40:07] BT: I mean, I don't know if you could classify Glide as niche, but it focuses just on mobile apps and it's essentially a UI on top of Google Sheets. All you need to know is how Google Sheet works. They have some sort of referencing fields and things. But, essentially, if you create a Google Sheet with 5 columns, like name, location, image URL and price, you would have your very basic version of like an Airbnb style app using Glide. You just click the sheets. Say, “Yeah, we’ll use this sheet,” and then you can load it.

We’ve done a couple tutorials with them and we’ve got an Airbnb clone where you have one Google Sheet, but you have a host app and a sort of user app. You can sign up. You can book. You can review. You can search on a map. Then as a host, you can manage your locations. You can have new ones, accept bookings and things like that, which is I think pretty crazy just based off of Google Sheet. They can build an Instagram clone with it. We built a cameo clone where you can sort of pretend, interact with a celebrity of some sort. That’s quite niche and that it’s just a mobile app builder on top of Google Sheets. But the things you can build with it are actually quite extensive.

[00:41:32] JM: Your process of learning how to use all these tools, it was partially through creating your tutorials to show other people how to build businesses or how to build applications. Tell me your workfl ow for creating these tutorials about low-code applications.

[00:41:54] BT: Initially, it was an accident I’d say. I think I was trying to validate any idea that I had. It just so happened that I was using no-code to build and then people would always ask how did I do that or how did I put this thing together, because it feels like a proper app.

I was always that person who couldn’t code, the annoying person trying to find and develop to build something for me and I was just like the ideas person. I had a lot of ideas initially and I think that actually a lot of ideas, probably when you cut them down, they probably fall into a bunch of categories or they’re either a membership style site. They’re other a job board style site, they’re marketplace. The functionality may differ slightly, but they’re all quite similar. I think they do fall into buckets, especially the things that you probably would end up building with no-code at this current time.

There was always just a list of things we could build. We could look at someone launched something new on Product Hunt or you’d see something coming out of YC that you think, “Oh! Well, I could probably build something similar to that in a lot less time.” It was just like a challenge to myself to see what different things could I build, adhere a new tool and think, or try that, or I’d see Zapier has an intuition with Google Docs. I built like – I didn’t realize that Google

Docs actually allows you to have dynamic data in it. I've built a form on Airtable that you could automatically generate invoices just using a Google Doc and a form on Airtable. It was just something that came out of – People were asking us if we could get VAT receipts for European companies. People were always sharing ideas on me or asking me how to do something and there's a never ending list on our backlog now of idea and things we could do. It's now quite easy to cut those down and start making them.

[00:43:56] JM: Your platform is called MakerPad. What are the goals of MakerPad?

[00:44:02] BT: We just want to teach people how to build stuff without code and hopefully at one point we won't need to mention the no-code thing or the without code thing. It will just be, "This is how you build a marketplace app," and we're sort of running through that. We've got a bunch of different things that we do. Tutorials is definitely one of them.

We're just announced our boot camp which starts on February 3rd, which is our foundations of no-code. We've got three categories there, I think is the foundations, and then the other boot camp is going to be building a business. Then the next one is automating at work. We think that's sort of our three main types of users. We've got a B2B offering where we help companies train their team on no-code tools, because I know a lot of sort of seed series A startups are probably using Airtable, Zapier and a bunch of other tools to run their workflows and run their businesses. When they hire new people or they want to get better at Airtalbe or build new things and launch them and then power their non-technical team. We often do that too. There are a bunch of things and we are just mostly focused on the education piece of teaching people how to do these things without needing to code.

[00:45:19] JM: Who are the kinds of people that a company would hire to do low-code application building? Are they called operations people? Internal tools people? I'm just trying to understand what this category of person who actually builds and works on low-code tools inside of an organization is.

[00:45:41] BT: Yeah. I think we see people in all different areas. We've got customer support people who want to automatically assign tickets and do things like that. There are definitely the operational types, marketing people who want to test new landing pages and things like that.

There are sales who want to build better lead generation tools, content where you can automatically publish content and things like that. Even we've done some automations around podcast and so you could pull in the person's latest social feeds and sort of preschedule their stuff and automatically transcribe it and things like that. There're tons of areas where we know that we can help. So we are trying to sort of do a few tutorials in each of these buckets to show off those areas to those people.

[00:46:33] JM: Do you see MakerPad as a GitHub for low-code applications? Do you want it to be a place where people can download or fork or take inspiration from previous low-code, fully-fledged businesses or applications?

[00:46:49] BT: I mean, I do describe it as a GitHub for no-code and I know that probably your audience may not like that as much. But I think that's how we see it in terms of the platform where you go, you learn about stuff. You can post stuff and you really build up your own profile if you want to get into this no-code space. I think it's just a matter of time until there is sort of a real way to form someone's project. There are lots of templates and stuff at the moment and we're working with some of these companies on some of the early programs to be part of that so that we could have those things.

Yeah, I think it's a matter of time until that happens or there's even component library on MakerPad that you can say, "Okay. I want to use Webflow and Airtable and I want to be able to do logins and memberships," and then it will show you, "Okay. Download this piece, download this piece, and put them together like this."

[00:47:47] JM: You're a part of Earnest Capital, which is a funding platform for revenue-generating businesses. Is there a connection between these higher-level low-code tools and the ability for a software business to generate profits more quickly than in the past?

[00:48:08] BT: Well, I obviously think that's true and MakerPad was a side project for 9 months and I was actually working at Earnest at the time until I decided to go fulltime, and Earnest funded me as well as a few others. But I think, yeah, it gets back to that point of you can launch it in a day, in a weekend and get your customers straight out-of-the-box rather than spending that time sort of building frameworks and everything else around what might take you a longer time. I think it helps get to the validation of making the money and then you can go anywhere from there.

[00:48:46] JM: The model for Earnest Capital, if I understand correctly, you fund businesses that are – They have some profits and in return you get some share of the profits that the business would generate in the future.

[00:49:04] BT: Yeah. They call it a shared earner's agreement, and they invest and it's like the term of, "Okay, if we gave you 100K, you'd give us back 300K, for example, based over certain time period, and our percentage of your profit share is basically after 12 months minus your salary." It's very founder friendly and it was definitely one of the reasons I wanted them to sort of invest in me. It was a great opportunity for us and it's working out really well for us at the moment. Yeah, they tend to look at monthly recurring revenue businesses and invest in those.

[00:49:48] JM: As we begin to close off, I want to just get some perspective for your beliefs about the future. We have kind of a classic team structure in how people commonly think about building software teams, software businesses. Either you have an engineer and a non-engineer as the founders of a business. But it seems like that has potentially changed with the idea that you can just validate a business with low-code tools. Do you have any strong beliefs about the classical team structure and what you should look for if you're trying to start a company these days? What kind of cross-training you need – If you don't necessarily need to have software engineering skills on your team, what team structure should you look for?

[00:50:43] BT: I think you need the ability to story tell regardless of sort of where you are in that team. I think if with no-code you could easily go and be a solo founder. Not that I recommend other people do that too, but that's sort of what I am. But I was lucky and that I was starting the no-code story perform no-code just getting a bit more popular. I think we'll see more and more people launch in one man businesses really quickly and not to say that there'll be any more successful. But it's just the case of you'd be able to build these smaller businesses and get them out there.

I do think that eventually we will see job descriptions that are saying we rely on Airtable, Zapier and Webflow, so we want our marketing growth team to be able to use these and push them out. We don't want – We have this software platform, but we want our non-technical employees to also be able to ship stuff too and not have the backlog as their excuse for why they haven't done it and try and rely on these expensive engineering resources, which I think is the case. I think that should be right. There should be ways to ship if you're not an engineer.

Yeah. I do want to have that all changed, but I do think that these sorts of skills will become more common place and I do think there's going to be – No-code doesn't mean no more coders. It means probably a lot more coders, because there are going to be some new more things that people want to build. I think if there's level of taking away the boring stuff with no-code. That opens up more creative stuff for people to create.

Yeah. I think we all kind of live harmoniously. Hopefully that's the case. But yeah, I think it should be supportive of both. I do think actually that no-code, low-code is one of the best ways to actually learn or start learning or start realizing that you want to learn how to code.

[00:52:43] JM: Are there some gaps in the tooling? When you look out over the entire set of low-code tools that you see, are there any places where the low-code ecosystem feels insuffi cient?

[00:53:00] BT: I think login sign up functionality is diffi cult, because when you have to write something down, that means when this person is logged in, they can now see this button and not this button and they can access this page and not this page. These things, when they're sort of crossed over in one place, they become a bit muddy.

I think there's a ton of – There's one company recently, Standard Library, who we worked with, and they were more sort of technical and they're now becoming more of a cross-over platform. You can build like a Slack bot or you can build some tools without code, but it will actually show you the output code so that actually it helps you learn to do those things and makes you understand, "Oh, when I've done that API call to Stripe, that's actually what I'm seeing in what code looks like versus I would see if I was just using sort of a UI click drag and dropdown type tool.

[00:54:04] JM: Are there any other classical beliefs or dogmas about company building or software building that you think we should call into question today?

[00:54:14] BT: I don't know. I think there's a big sort of debate on whether giving people the power to build their own businesses or build their own things. Is that a good thing or a bad thing? Ultimately, I think it's a good thing, creativity and everything else. But I mean, you're always going to see some questionable products out there launching every day and doing small lot of things. But I think it's better for the whole ecosystem. I think using no-code is now given the people who could never build anything the option to that. So many more ideas, many different ideas could be put out there. It'll be interesting to see how that develops.

[00:54:55] JM: Ben, thanks for coming on the show. It's been great talking to you.

[00:54:57] BT: Yeah, thanks for having me. I really enjoyed it.

[END OF INTERVIEW]

[00:55:07] JM: You probably do not enjoy searching for a job. Engineers don't like sacrificing their time to do phone screens, and we don't like doing whiteboard problems and working on tedious take home projects. Everyone knows the software hiring process is not perfect. But what's the alternative? Triplebyte is the alternative.

Triplebyte is a platform for finding a great software job faster. Triplebyte works with 400+ tech companies, including Dropbox, Adobe, Coursera and Cruise Automation. Triplebyte improves the hiring process by saving you time and fast-tracking you to final interviews. At triplebyte.com/sedaily, you can start your process by taking a quiz, and after the quiz you get interviewed by Triplebyte if you pass that quiz. If you pass that interview, you make it straight to multiple onsite interviews. If you take a job, you get an additional $1,000 signing bonus from Triplebyte because you use the link triplebyte.com/sedaily.

That $1,000 is nice, but you might be making much more since those multiple onsite interviews would put you in a great position to potentially get multiple offers, and then you could figure out what your salary actually should be. Triplebyte does not look at candidate's backgrounds, like resumes and where they've worked and where they went to school. Triplebyte only cares about whether someone can code. So I'm a huge fan of that aspect of their model. This means that they work with lots of people from nontraditional and unusual backgrounds.

To get started, just go to triplebyte.com/sedaily and take a quiz to get started. There's very little risk and you might find yourself in a great position getting multiple onsite interviews from just one quiz and a Triplebyte interview. Go to triplebyte.com/sedaily to try it out.

Thank you to Triplebyte.

[END]