Photo Credit

Paul: Content marketing is a good long term strategy but what I've done previously is I've used a lot of JV relationships. I used affiliates. An affiliate network helps send traffic to those products. Once it builds momentum then your brand starts to get known and you start to pick up a lot more traffic naturally.

James: James Schramko here. Thank you for listening to SuperFastBusiness.com. And I've got another interview. I'm really out about finding people, digging up the talent and also, some different topics.

And today's guest is someone who I was hanging out with at a recent conference in San Diego, sitting in the back of the room, discussing this enamoring topic of software and how

you set up software development. It's a fascinating topic so I brought onto the show an expert in this subject, Paul Clifford, welcome.

Paul: Hey James, thanks for inviting me in the show. I really appreciate it.

Disrupting The Marketplace

James: Now, you are doing something very disruptive in the marketplace. You even called your website "Disruptware." Something around that. Is that right? Disruptware.com

Paul: Yes. Absolutely. Essentially, my background has always been in software, especially in the SaaS market. SaaS stands for Software as Service, so essentially, a software delivered on the Web. What happened, my last SaaS business where I was actually an employee, we sold that company for like $38 million and since then I moved and started on my own and started doing online marketing and how to sell online and everything like that. And I started building some apps, some content marketing apps, I've got a keyword tool, some security software and I got another SaaS app in progress.

What I really found is that, in the marketing world, a lot of people are coming to me asking, how do I do this, how do I do that. There's lot of people launching their apps, which really, they built on sand. They fall over after a little while. My approach really is to build a software business. You got this opportunity now where you can really create software much faster, easier and cheaper than ever before which means it is also easy to go wrong, right? But using my expertise and my experience in this world and combining that with the marketing knowledge, I decided to put together a forum-based community and a mastermind where I will be there every day helping others of entrepreneurs who either want to start up and get into

software or they've already got something and they want a scale. And I put all that together and we called that Disruptware

James: Nice. I've seen some really good examples of one of my regular guests in this show.He's Clay Collins. And he built the LeadPages and that is probably one of the greatest success stories I know. It is now one of the top 1000 websites in the world now.

Paul: It's brilliant.

Targeting The Right Audience

James: As an app. As a software. As a service. It's so sensible and he really jumped from this marketing crowd where there's a lot of WordPress plugins into the online app space and he went up the category and something struck my mind while we are sitting in the conference there with about two and a half thousand attendees, you said to me, well, "this is a very small market for you." Who do you think you were talking to here?

Paul: In terms of the software?

James: Yes, who should be building software businesses and paying to the stuff you were talking about? Who's your target audience?

Paul: Right. So, there's two kind of levels for everyone who wants to get into software, and my network again has two strings to it. I've got the marketing world which I've seen grown up with over the last few years and there's all the people who just really want to get into it. They started creating desktop apps or plugins or things like that. I really want to know, how do they scale

that, how do they take that to the next level? And how do they get into SaaS? The other side is any entrepreneur or info marketer who wants to create a real business that isn't something that's just gonna fall over next year, something that they gonna keep working at. The great thing about creating a software business is that there are so many ways you're connected from it or not at all if you don't want it. You can create a lifestyle business or you can create something to sell. Either way, it's like a real business. OK? And all you gonna know really is, is it gonna work? Is your idea gonna work? And how can you build it?

James: Well, it sounds very simple there. I think we should dig in to that. I know there's courses on this, I had guests before from The Foundation and that was a pretty popular course certainly in the marketing space and I think the appeal was that you didn't have to have an idea, you didn't have to have money, you just have to follow the course. Can it be that simple?

The Minimum Viable Product

Paul: To a certain extent. What we have to remember is that the whole world of software has changed radically, especially for the last three or four years and a lot of this has come from a movement called the Lean Startup by a chap called Eric Ries. And what was happening basically is you know all this startups in the old days they required huge development teams, huge money and you never really knew if this is gonna work. When I was in my corporate days, I had a development team of 40 people and it took ages to get stuff done and the Lean Startup Movement is about getting the minimum possible product, even on paper or even on Powerpoint to a certain extent. Getting your idea to something that you can test with the customer or with the prospect and the sooner that you can create something, that you can communicate your idea to a customer, and workout, is this really gonna solve the problem? Then the more your whole business model becomes validated.

James: The minimum viable product.

Paul: Exactly. The minimum viable product, or MVP as it's called. Because what you have to remember is that 75 percent of startups fail and it sounds like a huge number and the key to understand is that the people who actually succeed in that remaining 25 percent, their Plan A, their original idea isn't actually their success. The idea is you create your Plan A and whatever is your original thinking is, it will evolve into Plan B, or your Plan C. Whatever that is, it's going to be the business you are going to end up with. For example, if you look at some big successes, they never started like that. Like Hotmail. Everyone knows Hotmail. But it never started as a mailing app. They started that, it was called, JavaSoft. What they wanted to do is create a database that can be accessed through the Web. Their challenge is actually connecting to the email accounts. After they went live with JavaSoft, they realized, "hang on, the email list is more of a pressing issue. Let's do that instead." And that's how Hotmail came about. Even Paypal never started as an online money transfer system. It actually was meant to be, it started as a cryptography software for Palm Pilots so they could actually, like, beam software to each other. They dumped the idea and took it to the Web and of course it exploded.

The point is that the people who have the great success, they don't have the great success from day 1. They start off with the Plan A. The trick is to understand when to switch to Plan B, or when to switch to Plan C. It's always evolving, and in the Lean Startup world, he calls this pivoting, or pivot. So you know the key thing is to build is something as quickly as you possibly can and as cheaply as you possibly can. Nowadays, it is much easier to do that than it has ever been before, because of the global economy and outsourcing, you you've got these freelancing sites or you can partner up with someone techie. You can build something up really, really quickly out of your own pocket and start testing that on your potential customers and validate whether it is something that is going to solve their problem. Do I make sense, right?

James: It does make sense. What you're saying is that there's going to be reiterations from where you start. So, you start with something and it's gonna evolve. I've heard that term build-measure-learn loop. So you build something, you measure the results, you learn the outcomes and then you can make decisions from that. I guess to some extent, that's pretty much how I've run my business from a strategic point of view. I've built out services, I put out information products, I measure all the stats and downloads to see what people want to buy and make decisions based on that and it's exactly the same for software. It sounds easy, you've got all these global outsourcing talent. You've got software designers who can design stuff. You don't have to bring too much to the party because your customers are going to validate your ideas. What would be the sequence of events that someone who's starting, if they have blank whiteboard in front of them, what are the steps?

Starting From Scratch

Paul: For the people who have no idea what they want to do, let's just cover that for a second. I see two fundamental approaches that you can take. The first one is what I call invent it by the wheel, you can take something that's already working, so it's already selling, from some other competitor. By knowing that it is already working and it's successful, you know there's a market there and there's an appetite for it. All you have to do is to really analyze how you can make it better, how you can make it quicker or how you can make it easier to use or perhaps, how you can take that to a different market and niche down really, really small into a specific market which you have knowledge in. I think there's a lot of people who want to be entrepreneurs who might be in the corporate world at their desk job and thinking I just don't have that great idea. When you really get head around it and I understand that it's very rare to get that great idea. I mean Google wasn't the first to search, right? They weren't the first but they took something that is already there in Yahoo and made it much, much better. Look at Apple and the iPod. There were mp3s around before the iPod came to be.

James: We were talking about that today with a friend of mine and I realized that half the people listening to my show don't even know what a Walkman was. That's embarrassing.

Paul: I remember those, the Walkman then the Discman. That's one way you look at it – invent it by the wheel. The other thing which is a little bit more unique, when you want to do something, when you recognize that there might be a gap in the market and you need to really find that pain within your prospects. To do that, and I suggest you do that if you know the market and if you know the industry you are in and you've got a good understanding. What you need to do is to ask these prospects and question whether the pain really exists and if the pain exists, whether it is monetizable. You need to ask how they are doing this thing now, how is it costing them, what's the labor costs in reality. When you start asking those kind of questions, don't ask the solutions, you just want to ask their problems. Then once you identified that and once you understand that it's a real monetizable problem, then you can start looking into the solution for it. If you start asking them for solutions, people will say the wrong thing and they will throw you off the track. Henry Ford was famous for saying that his customers don't know what they want because when he went out and ask the people what they want, they said faster horses.

James: We need to check if that's a true quote or not but I totally get the point of it. And Steve Jobs who was of a similar ilk, he was not really worried too much about what people wanted. I think he was insistent on not allowing for people to have a stylus and he just wanted one button on the iPhone. He could see forward but I guess it was kind of hard that he wasn't really big on market research in terms of bringing in groups to tell him all about what they wanted either. How did he he innovate and get ideas?.

Paul: But they were geniuses, they were very very clever people right? When you think of it broader, around people like you and I how can we actually go? I'm not saying you are not a genius.

James: Oh, you know I was just thinking about that, as a matter of fact. Obviously, we're implying that none other of our listeners are and probably some are. Just on that actually, one of the interesting facts about Henry Ford that I didn't realize until I watched the very long documentary about him was that he built this huge factory that could build a thousand cars a day and he hated it. He felt sick about what he built in hindsight. So he actually did make that mistake you're talking about earlier, putting in too much effort and building something too big until he discovered that it wasn't actually what he wanted in the end.

Paul: Right. Right. That's interesting. So I guess, the covering off the idea part, once you have that, the critical thing then is really to turn that idea. Get it out of your head into something, on paper, on screen or on your computer. So you can actually start evolving that into something workable and in the business, we call that a wireframe. So you start designing a wireframe. I use a tool called Balsamiq, online Balsamiq and it's a really fast way to mock up screens. What's really good about that is that it validates how this thing is going to work to yourself even before it goes to any developer. When you start drawing this thing, you start to realize whether the screen should have this or this button, or whatever. And the whole thing will start to come alive as you designed it on your screen. And with Balsamiq, what you can do is that you can link the screens together behind certain buttons. Let's say, you put a button on the screen then you can link that to another wireframe and after a little while, you end up with a clickable wireframe, you start seeing it come alive. Once you have that, something clickable starts to make sense not only to you, but also, that's something you can actually take to your prospects. You can actually take that to your market, it's clickable and it really helps again to communicate your product to your potential customers and work out with this and see if it's going to solve their problem.More importantly, are they going to pay money for it?

James: That's very cool. You can start showing people your concept before you have to pay someone to build it all.

Paul: Exactly.

James: I remembered James Dyson talking about this. I had him come along to my FastWebFormula event and share his million-dollar software secrets and he had also mentioned that you can build out stuff on Keynote or Powerpoint to show people a mock up of whatever the screen would look like as well but the link button sounds fantastic.

Paul: Yeah. Don't get me wrong. There are loads and loads of tools. I mean there's a whole industry around wireframing. I just like Balsamiq because it's very easy to use, it's very quick to build screens. I did like a demo, a walkthrough on my site, and so something that anyone can pick up and run with.

Next Step: Building The Software

So I guess the next step then is how you get this thing built. Let's assume you've gone through this process now. You got a clickable wireframe. What you need to think about is, are there any sort of complex processes behind certain buttons? Anything that you think is particularly complex is worth writing down, just bullet points and again, you can do this within the tool as well but it's important to have your complete vision of what your version 1 or your MVP is going to look like, all sketched out. Because the rule of thumb is the more time you spend in this process, getting it designed as right as you think you can, then the less money you'll spend in the development process.

James: Sounds like building a house. If you designed it properly, then they can build it first time, then you have to rework stuff when you realize the door can't open, that something is too short or you got the wrong material.

Paul: Exactly. The cost of all development comes at the backend. It comes with the bug, in the reworking code, and all that sort of stuff. The more you can get that thinking done at the front end, then it's going to save you a fortune down the line. So, I guess you've got, you gotta get this built. Generally, there are two approaches. You can either partner with someone, let's say you've got some really good technical guy, and doing that then you don't necessarily put your hand in your packet and pay out cash right away. Of course, there are disadvantages in partnering, and that is that what happens when they are not technically as good as what you thought they were. Secondly you've given them a lot of equity.

James: That's the biggest hidden problem I found with partnerships. It's not if they don't go well. Because you lose a little bit of energy or time. It's when the partnership goes wildly successful, the partnership can be terrible. I look up things like Facebook, it caused a little bit of conflict there when people want to take ownership and bump other people off the boat and all those sorts of stuff.

Paul: Yes exactly. If you want to do the partnerships approach, you really got to write it down even if it is just in the email. But at least, you write down what happens if it's successful, what happens if it fails. What's our end goal, are we gonna sell it? Are we just gonna live off the profit? What happens if I lose or you lose interest which does happen a lot, right?

James: Yes.

Paul: Can you buy each other out?

James: That's where you have to have a strong strategy. So, if you don't have the capital you could give a share to the developer. That's sort of what you are saying here.

Paul: Exactly. The other way to do it of course, is to outsource or hire a developer. You can do that through recommendation or you can do that through one of the services online. There's a whole load of them now. The one I particularly like is Freelancer.com. That particular service has like evolved from years ago when it was used to be called Rent A Coder. The marketplace in Freelancer.com is very strong in developers unlike Odesks and some of the others. And with that sort of pedigree, you tend to get much better applications coming through for your project. So let me take you through that then. So you got your wireframe, you got your design pretty much ready and what you want to do now is to get some bids on that. So the first rule of thumb is always look on a fixed price basis. OK? So you never want to pay a freelancer by hour. It's always on a fixed price. What you want to is you want to put your project on Freelancer.com and you start getting people apply with prices and bidding for your project. The key things to know is, obviously it's a big topic, but the key thing to know is make your shortlist based on their feedback. Their ratings. Again, it's a marketplace with reviews and things like that. So look through what they've done before. In your description of your project, specifically ask them for examples of similar work. So want essentially to have people bidding on that who've done very much the similar sort of thing before. Using similar sort of technologies. Never go for the highest price, but equally, never go for the lowest price. You'll start to see what feels right, it's going to be somewhere in the middle and you'll start to get some bits in the middle.

The next thing to do is really get them chatting. You need to have some sort of chat with them, with the individual and start asking them for examples they were talking through that. What problems did they have before and start building a rapport. You'll find once you've done two or three of these, you'll start to really understand whether the guy knows what he's talking about but also you can communicate with this developer on a daily or weekly basis.

The other little tip is when you can, is get them on Skype so then you're talking to them directly and you build, you know when they're online.You want to basically make sure that once this

project is started, they do not disappear off and everything. The only reason that some people disappear on these sites is because often they are embarrassed to admit that they've hit an issue. Especially in some countries, their culture is such that they would rather stay quiet than come to you with a problem and what you want really is you want to know about the problem as soon as possible because you know that you can work around anything. You can find a solution between the two of you and so it's really important to get that communication going. So am I making sense to you?

Quick Recap

James: Yes. Absolutely. Just a quick recap here, we've got an idea, whether we're making a better wheel or addressing a pain point. We've documented it. We've put it up on a wireframe. We've got someone to say, this is valuable, I will pay you for this. You've specced out the software brief, you're either getting a partner or put it up for bid on Freelancer. You want a fixed price. You've done a bit of an interview process with the shortlisted people. Not the highest or the lowest bidder but someone with a good rating. And you've got a good enough working rapport where you can slice through any objections.

Paul: Exactly. Basically, you agree on the timeframes, when you want this delivered. If it's quite a large project, it's good to break it into milestones. So you have like a version 1 or whatever, and this is what should be in this version.

James: So is that how you pay as well?

Paul: Exactly. You can pay on milestones as well so it's a win-win. They are not committing to months or weeks of work with nothing. So you're also seeing results throughout the process.

James: So everyone is moving at the same pace. I must admit I think I made one of these mistakes. I kind of went into a projected amount for a software, I ended up paying sort of hourly rates and it cost me a lot more than I expected. Plus that point where I'm so far into it that I had to just keep going to get any kind of return and it was unappealing.

Paul: Yes. I know. I know.

James: I'm sure you've been there and surely, other people too. We are gonna save a lot of mistakes by discussing this in detail.

Investing On People

Paul: The great thing is when you find someone really good, you can build long term relationships with them anyway. And then if necessary, you can move to hourly for the downline, because you got a trust, you know how fast they work and everything else. And working on an hourly basis is a little bit easier because you don't have to define everything upfront. For the first project, for the initial thing, get a milestone. Work out your timeline. Now obviously to you, you don't communicate this, but you need to possibly like double or triple whatever their estimate is because they are going to go very enthusiastically about something, and it's going to be more complicated than they first thought, it will drag on… .

James: It always seems to cost twice as much and take twice as long.

Paul: But if you fixed price it, you're protected from the financial perspective. But all you've got to do is to just tell yourself: OK, It's going to take two or three times as long as what they're telling me.

James: That is very funny. It seems a phenomenon, it's like weather forecast or cooking recipe. Oftentimes, they never seem accurate.

Paul: Yes exactly. Once you've awarded the freelancer the job then off they go and they start building your software. A couple of key points, first of all, make sure you are speaking to them at least every every week if not every other day. Just checking. How's everything going? Any problems? And get some feedback. It's critical that you do that. Don't leave it for like two weeks, three weeks, four weeks and then check in with the guy. That's when most of disasters happen. You need to be on that case, like every few days.

The second thing is you want to build. So you want a delivery of whatever it is they're building as soon as possible, whether it is complete or not. I suggest weekly. So every week, you need them to deliver you the code, even though there might be one feature in it. But you want it to be delivered to you every week as a project. There's two reasons for that. One is because… first of all, you got a code on your site, if anything goes wrong, you can find someone else to take over. The second thing is, you can start testing as soon as possible. The sooner you start testing it, the sooner you log bugs. And what you should be doing is ensure that they're fixing bugs during the project, not at the end.

James: Right. Does it tend to blowout the time to complete, especially when you keep trying to add features when you find something like "oh, we should have done this differently,", the feature creeps, I suppose that's what you call that.

Paul: Yes. I'll come to that in a second. But on the bug side, you'll find that it doesn't really. Because, remember that they're kind of under pressure in a way to commit. They're committed to a timeframe and all you're doing is logging bugs. When they estimated that timeframe, they are estimating the bug fixing anyway. The difference is that you are getting them to fix the bugs during the project rather leaving everything to the end and the reason for that is that when you

leave everything to the end, you end up with a completely unknown time scale. Right? You got like a hundred or so bugs. You don't know how long they're gonna take to fix.

James: You want it to happen in one of the phases rather than after completion.

Paul: Exactly but if you're fixing 20 bugs a week, then at the end of it, you probably have onlly 20 bugs left.

James: Yup.

Paul: It's a fact of life no matter what. And in terms of the other point you made which is about scope creep or feature creep? This is where you need to be strong on yourself. Because you're going to have tons of ideas. You're going to think, ah, why if we do that, are we really cool? That's where you're really got to put things aside, whatever you are using to design your project. Design it up. Get it already but don't give it to them until the end of the project.

James: Yup.

Paul: OK? Store everything aside. Once you get the final version, you might change your mind completely but it's the worst thing. Once you start throwing features and start working on the feature in the middle of the project, then all the time scales get blown out and more importantly, they then got a reason to come back to you with the delay.

James: Yes, 'cause you changed this, you changed that.

Paul: Exactly.

What To Do With The Finished Product?

James: OK so let's imagine that we've been really disciplined. We bug hunted, developers still on-board, everything is going well and we now have our delivered software. What do we do with that work? I guess we're going to sell it or go back to our customers now and market it?

Paul: Yup, essentially. You've already got now some prospects interested because you've already started talking to them. As soon as you have now this initial build, then you can now go back to them, say "Hey, you had this problem and you said it's a real issue. Well, here's my solution that we've already talked about." You can actually start asking them for money, potentially. Some people will give a customer free access. There are two schools of thought. Really, what you want to do is kind of get them to pay something. Because as soon as they pay something, then they are emotionally committed to it. Though they will invest more in using it and raising issues to you than if you give it to them for free.

James: Yup.

Paul: Yeah. And I literally just interviewed someone the other day on my blog about this. And that's exactly what he did. He wouldn't…his whole group, you know, he called like the grandfather club or a founder's club, you know, obviously gave him a special rate, but he wouldn't give it to them for free. The key thing is, they bought in. They are using it as part of their business. It comes back to the whole thing about freemium software. You know, you sell so much freemium software, a lot of people just won't use it.

James: Yes. I think, and that's, I've seen a lot of pricing schedules, and it seems to lean… Certainly, when I've done testing on my membership, it seems to lean in towards having a little

bit of stakeholding happening with the cost upfront. Even if you have less people, the people there are wanting to be there.

Paul: Right. Right. And also, don't just expect it to sell itself, you know, just on marketing. You've already built rapport with some of these people. There's no reason why you can't speak some more, like via the phone or meet or anything like that. We talk a lot about marketing and online sales and everything like that and that's obviously the big part of it. But some people get wrapped up so much in that side that they forget, and often you can phone these people and show them the software yourself. And it's a lot easier to close a customer face to face than if you're just doing everything online and building up your marketing funnels and everything else. Makes sense, right? I mean even the biggest, the big software tools, like KISSmetrics – a very, very successful SaaS app… You know, what they're really good at, is since you sign up for your free trial, the next day your phone rings. They're on you straightaway, just want to check if everything's alright, how can I set you up, can I help you with this, can I give you a quick demo through….You know, they're on your case. In the business it's called inside sales but basically, they establish that touch point straightaway.

James: Yeah, that's very good. It's a marketing sequence. The game is to move people from engaged user to paying engaged user. And I've just been through that with a service that I joined a week or two ago. I think it's called mention.net. And they're very ambitiously trying to move me through to paying customer. And also my eWay gateway have been sending me emails encouraging me to take up some anti-fraud software and all they ask me to do is just reply, "Yes, I'm interested," and the next thing you know, the phone is ringing. So they've got this link up between segmented customers and offline call to action.

Paul: Yeah, exactly.

The Biggest Mistakes During Software Development

James: So a lot of this I guess will depend on the market you're in. Let's see, so you've given us a great sort of run-through of the steps from start to finish to have a product ready. What do you think some of the biggest mistakes you've seen people making with software have been?

Paul: I think the first was a classic one, that you're not solving a problem. You're creating a product maybe out of emotion. It could be, hey, someone else has got this out, I got to do one too, and you create something and obviously it doesn't solve a problem at all and they just… you know, no one buys it..

James: They're just releasing a complete rip-off of a product that doesn't go any further or add any extra value to solving the problem.

Paul: Right. Exactly. I think the other thing… when I talk to people, the other big thing that goes wrong with outsourcing is pretty much what I've covered in terms of the communication piece. You know, that's… I think people would invest a lot of time in doing the design, the wireframe, put it on a freelance site, click the button, hire them, and then just expect something to end up on their plate four weeks later.

James: So not working it enough?

Paul: Yeah. You know, You have to kind of work the system. You can't just expect to put something out there, and just expect the finished result, four or eight weeks, whatever the timeframe is, back.

James: I think some of the courses out there sort of imply that it is that easy.

Paul: Yeah.

James: You know, send out a Powerpoint deck, next week you'll have a number one app on Apple.

Paul: Yeah. Disaster, right?

James: It seems to be. That's why I'm interested in it. I do have SaaS, but I don't think I've done as well as I could have with it, and I've ended up bundling it in with my membership forum, my community members. But it was based around solving my own pain. It goes to figure that a lot of people also have the same pain that I have and it solves their problem, too.

Paul: Scratching your own itch, basically.

James: Exactly. I have enough of a community of people who are in the same sort of boat as me. It did make sense at the time.

Paul: Yeah. I think, you know, the outsourcing bet is a classic one that goes wrong. I think getting your idea out of your head into a wireframe, just spending a bit of time with that. That's another thing. I think inherently people can be lazy. I'm lazy. I would much rather just phone a developer and say, hey, I've got this idea. Can you just go and build it? That's what we'd want to do and I think going through the process of designing a wireframe is like a little pain that you just have to do, because it helps you imagine it and evolve the product yourself and

communicate to yourself before you put it out there. It's a really good way of aligning the idea into something that's actually workable, an actual tool that's going to work.

So I think those are the two, I guess two or three main issues. And of course, making sure that the pain that you are trying to solve is monetizable. There's no good in solving a problem… Solving a problem for a lawyer is going to have a lot more value maybe than solving a problem for a school, OK? Even though solving a problem for school would be great, but in terms of monetary value and how much they're going to pay for it, that's really what you've got to really think about.

James: So really a lot of this, like all good marketing, should really focus on the customer. Be really clear on who your customer is. How you can reach them, what they're willing to pay for, what they already pay for is a great example, and you build up from there.

Paul: Yup. Exactly. Once you got this whole thing live, try not to get too emotionally involved and start working on the numbers. And I know you're great at this, James, you are very metric-driven. And you need to kind of define to yourself what success looks like at each step of the way. In three months, let's say for example, I want to have 50 customers. If I want 50 customers, then how many leads do I need? And if I have this many leads, how much traffic do I need and how am I going to convert that traffic into leads? Start mapping everything out on a spreadsheet so that you know whether you're on track or not. Because what happens is that you got this great app in your mind, you've got a great website and everything's there, but how do you know you are actually moving through and heading for sales, without a sort of a dashboard of numbers that you can actually tick off as you go through the process?

Paul's Biggest Success To Date

James: Nice. So your whole website is centered around helping people grow a business for themselves around the software development.

Paul: Right.

James: What's been the biggest success you've had to date with software for yourself, after you left the company?

Paul: I launched a product called Kudani, which is for content marketing and it's for bloggers, where they could actually curate content from newsfeeds. It has like a newsfeed tool within it. And it all runs from the desktop and that was hugely successful. I launched that last year. And that's done multiple six figures since. So that's great. I've got a lovely security software tool that sits on people's blogs and prevents them getting hacked. And even though that didn't have like, or doesn't have the same sort of revenue, it's the one that everyone loves. It's got the lowest refund rate ever, and it's very, very reliable, and it does a good job.

James: And what's your best marketing tactic to let people know about your software?

Paul: I think content marketing is a good long-term strategy. But what I've done previously is I've used a lot of JV relationships, and used affiliates. An affiliate network helps send traffic to those products. Once it builds up momentum then your brand starts to get known and you start to pick up a lot more traffic naturally. Essentially, it's a two-pronged approach. Use affiliate traffic for the launch and then the content marketing is good for the long-term stuff.

James: Nice. Alright. Well, I really appreciate you coming on to the show and sharing these great ideas. We've gone all the way from not even having an idea to having a software produced and then marketed, too. This is going to head over to Disruptware.com. Find out more about Paul and the software system. And thanks so much for coming on and sharing these ideas.

Paul: Yeah. I really appreciate being a guest on your show James. I follow it, and I love the stuff you're doing. Thanks a lot.

James: You're welcome.