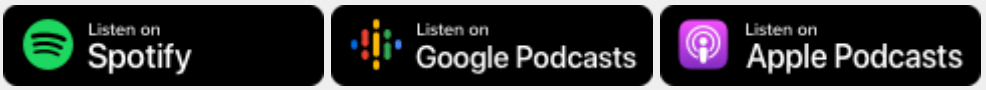
Pierwsze kroki w IT

PODCAST



Wywiad z programistą: przebranżowienie i sprawdzone wskazówki – część 1

Gość: Mateusz Jarzębowski-Bownik

Wszystkie polecane materiały i linki znajdziesz na stronie odcinka:

https://devmentor.pl/b/wywiad-z-programista-przebranzowienie-i-sprawdzone-wskazowki-czesc-1

Dziś moim gościem jest Mateusz Jarzębowski-Bownik. Mateusz opowie nam o swoim przebranżowieniu na programistę, podzieli się wskazówkami, jak skutecznie zmienić zawód. Mateuszu, dziękuję, że przyjąłeś moje zaproszenie na rozmowę.

DEVMENTOR.PL
YOUR CODE COACH

Dzięki za zaproszenie, Mateuszu. Trochę powtarzalnie brzmimy, bo się nazywamy akurat tak samo. Jestem przeszczęśliwy, że mnie zaprosiłeś do najlepszego podcastu w branży IT według serwisu itweek.pl. Mam nadzieję, że będzie to w miarę interesująca rozmowa – tak jak nasze rozmowy przed podcastem.

Bardzo dziękuję za ten wstęp.

Przejdźmy teraz do Twojej osoby. Powiedz nam, proszę, co łączy Cię z branżą.

Jasne, trzeba się przedstawić, żeby ludzie pomyśleli, że ja w ogóle wiem, o czym gadam. Historia jest mniej więcej taka, że od zawsze troszeczkę ciągnęło mnie do technologii. Nigdy jakoś tego szczególnie nie spełniłem, życie poprowadziło mnie bardziej w stronę ścieżki zarządczej. Byłem PM-em przez dziesięć lat – większość z tego czasu w marketingu, mniejszość w IT.

Wydaje mi się, że byłem w tej pracy dosyć dobry i rzeczywiście miałem dużo klientów, którzy na przykład mówili, że bardzo dobrze się za mną pracuje. Zrobiłem kilka rzeczy, z których jestem naprawdę niezwykle dumny i to wszystko zaowocowało tym, że chciałem w jakiś sposób poszerzać swoje kompetencje. W świecie PM-a to nie jest proste. Mamy opcję typu na przykład zrobienie MBA. Generalnie kursy zawodowe nie są jakoś szczególnie mocno cenione, jeżeli masz już dużo doświadczenia zawodowego, przynajmniej tyle, ile ja.

Zacząłem więc uczyć się programowania po to, żeby lepiej dogadać się ze swoimi developerami, żeby developerzy wiedzieli, że jeżeli coś mi

mówią, to ja generalnie będę to ogarniał i dzięki temu będę mógł stworzyć im lepsze warunki do dobrej pracy – to jest drugie najważniejsze zadanie PM-a, taka służalcza praca wobec developerów. Pierwszym najważniejszym zadaniem jest tłumaczenie biznesowi tego, co oni mówią, i odwrotnie. Chciałem to lepiej rozumieć i dlatego zacząłem się uczyć programowania.

Nigdy nie chciałem być developerem, ale mi się to po prostu strasznie spodobało. Czyniąc długą historię krótką, obecnie jestem fullstack developerem w ekosystemie, powiedzmy, JavaScriptu/TypeScriptu. Zacząłem się uczyć w 2020 roku (była wtedy pandemia covidu). Na naukę zostawało mi bardzo mało czasu, ponieważ miałem wtedy w domu niemowlaka.

Nie pamiętam dokładnie, kiedy zacząłem się uczyć, Helena pewnie miała 3-6 miesięcy. Bardzo chciałem się dowiedzieć, w jaki sposób sobie tę naukę przyspieszyć. W związku z tym, że jestem takim trochę nerdem, słucham strasznie dużo audiobooków, wykładów, podcastów, wywiadów ze znanymi albo z mniej znanymi naukowcami z dziedzin, które akurat w danym momencie mnie zainteresują. Wydaje mi się, że poza programowaniem posiadam dosyć dużą wiedzę właśnie z tematów kognitywnych związanych z efektywną nauką, które też będę się starał w jakiś sposób tutaj przemycić.

Chyba najważniejsze, żeby ten odcinek był taki bardziej, powiedzmy, narzędziowy – żeby dać ludziom w miarę gotowe rozwiązanie, a nie przedstawiać fundamenty. Chociaż tych fundamentów też postaram się troszkę przemycić.

No i mi się udało. Udało mi się przebranżowić…

Może nie mówmy, że się udało. To po prostu ciężka praca, bo *udało* to tak, jakbyśmy powiedzieli, że to był przypadek, a chyba nie o to chodzi.

Tak jest. Też słuchałem Twojego ostatniego odcinka z panem Rozieckim – to jest w ogóle odcinek, który według mnie każdy powinien przesłuchać. Wasza pierwsza rozmowa to jest naprawdę sztos. Każdy, kto chce się przebranżowić, powinien tego posłuchać. Przesłuchajcie obecny odcinek do końca, ale na liście ustawcie sobie ten jako kolejny.

I jako taki ostatek: byłem PM-em, jestem developerem, troszeczkę się znam na biologii – bardzo troszeczkę, ale widzę, że ludzie się znają na tym jeszcze mniej. Taka ostatnia wisienka na torcie, to jest coś, co robię w wolnym czasie: bardzo dużo się udzielam na grupach facebookowych dla początkujących developerów. W większości ludzie nie pytają nawet o to, w jaki sposób poprawić błędy w swoim kodzie, tylko pytają o rzeczy bardziej z doradztwa zawodowego: *Chcę się przebranżowić, od czego zacząć? Trochę utknąłem, dlaczego nie mogę znaleźć pracy?* itd.

To rzeczywiście jest coś takiego, co robię wyłącznie dla przyjemności. Bardzo dużo się na tych grupkach, na różnych czatach wypowiadam. Pozdrawiam stałych bywalców. Z tego powstała lista często powtarzających się pytań, na które postaramy się pewnie przemycić odpowiedź.

To może od razu powiedzmy, że dzisiaj odpowiemy na trzy najciekawsze pytania, a jeżeli ten odcinek będzie miał dobry odbiór – czyli najlepiej komentarze i łapki w górę – to nagramy osobny na temat najczęściej pojawiających się pytań. Zgadza się, Mateuszu?

Tak, i wtedy zrobimy sobie taką dłuższą listę, na przykład dziesięciu-piętnastu pytań, które najczęściej się pojawiają na tych grupkach. Ja naprawdę non stop odpowiadam na te same pytania. Non stop. Zacząłem już nawet robić takie kopiuj-wklejki, które trochę tylko zmieniam, żeby były bardziej dostosowane pod konkretną osobę, bo szczegóły się różnią, ale duże klocki zawsze są takie same.

Przykładowo: *Za którą technologię się wziąć, jak zacząć? Nie mogę dostać pracy, pomimo tego że robię XYZ* – i tego typu rzeczy. To są non stop bardzo powtarzalne pytania, więc jeżeli chcecie o tym odcinek, jeżeli Wam się spodoba, to dawajcie znać w komentarzach i będziemy z Mateuszem działać.

Tak jest.

A teraz zadam pierwsze, podstawowe pytanie. Zanim podjąłeś decyzję o przebranżowieniu, to – jak wspomniałeś – programowałeś, żeby poszerzyć swoje kompetencje. Po czym poznałeś, że programowanie to jest właśnie to, czym chcesz się zająć?

Kurczę, to jest zabawna historia, wiesz? Dlatego, że to nie był jeden moment, to stopniowo przychodziło do mojej świadomości. Pi razy oko uczyłem się dwa lata z lekkim haczykiem na przebranżowienie. To dosyć długi czas, ale przez pierwszy rok uczyłem się wyłącznie dla poszerzenia swoich kompetencji oraz dla zabawy.

Wyklikałem jeden śmieszny projekt, który był pseudoanalizą statystyczną: kierowcy których marek samochodów są największymi

dzbanami. Mój dobry przyjaciel pracuje w centrali BMW i chcieliśmy zbadać w jakiś możliwie najbardziej obiektywny sposób, czy naprawdę kierowcy BMW są tak straszni jak stereotypy mówią. Wyklikałem ten projekt i dotarło do mnie, że naprawdę strasznie mi się to spodobało. Nawet dzisiaj, jak pracuję, to wydaje mi się, że ludzie płacą mi za to, żebym ja czerpał przyjemność z pisania w VSCode.

Takim katalizatorem tego, żebym naprawdę ostatecznie zdał sobie z tego sprawę, była pewna dziwna historia. Dostałem ofertę właściwie pracy marzeń – zawodu, który jeszcze nie istniał. Robiłem w marketingu głównie e-sportowym i to była praca w e-sporcie. Wtedy naprawdę mocno zacząłem się zastanawiać nad tym, co ja chcę robić w życiu. Czy chcę dalej robić to samo, co robiłem, czy chcę iść jednak bardziej w stronę IT? Wtedy podjąłem już taką megaświadomą decyzję, bardzo dokładnie zaplanowaną – co chcę robić, kiedy, jak. I po prostu to zrobiłem. Wydaje mi się więc, że to spełnienie mojego marzenia było katalizatorem tego, żeby to marzenie kompletnie porzucić.

Ciekawe, raczej każdy zrobiłby na odwrót.

Jak to zrobić dobrze, żeby podjąć taką trudną, racjonalną decyzję? Jak do tego podejść, żeby było to przemyślane?

Wydaje mi się, że pytanie jest megaogólne i tak naprawdę każdy musi dojść do takiej decyzji samodzielnie. Jeżeli ta decyzja będzie motywowana wyłącznie zewnętrznymi czynnikami, to najprawdopodobniej nie podałasz, dlatego że nie zdajesz sobie sprawy z

takiej, powiedzmy, listy plusów i minusów w zawodzie albo plusów i minusów samego procesu przebranżowienia.

Najprawdopodobniej nie jesteś przygotowany na to, ile to jest pracy. To jest praca, która jest tytaniczna. Tego naprawdę jest bardzo dużo, więc pomaga, jeżeli to lubisz albo przynajmniej jeżeli jesteś osobą bardzo mocno zmotywowaną. To jest wymagający zawód umysłowy, który wykonujesz raczej solo. Wiem to po sobie, bo obecnie mam biuro w piwnicy, które uwielbiam i bardzo polecam.

Solo – chodzi o to, że siedzisz w tej piwnicy, ale pewnie pracujesz w teamie, tak?

Tak, ale ile się widzisz z teamem? Widzisz się czasami na pair programmingu, czyli raczej rzadko. To zależy oczywiście od firmy, ale raczej piszesz kod samemu. To nie jest rzecz, do której człowiek jako zwierzę został przygotowany genetycznie. Raczej dobrze, jeżeli się to lubi.

Powiedzmy, że jesteśmy z podobnego rocznika, więc w naszym przypadku może faktycznie tak to wygląda, ale na przykład nasze dzieci (patrząc po mojej córze), to nie wiem, czy nie będą przygotowane do takiej samodzielnej pracy przed komputerem. Moja córka, mając trzy lata, jest już z komputerem za pan brat. Wiadomo, że to mocno zależy od rodziców, ale takie dziecko od zawsze jest z komputerami i pewnie temu młodszemu pokoleniu będzie łatwiej.

Wydaje mi się, że są też osoby, które tak naturalnie bardziej ciągnie do natury. Na pewno u każdego są dwa komponenty: genetyczny, którego

nie może zmienić, i środowiskowy, czyli na przykład to, co ktoś Ci pokaże jako dziecku. Przykładowo jeżeli bardzo często siedzisz na dworze ze znajomymi, jesteś w harcerstwie, jeździsz na obozy, to w późniejszym wieku raczej będziesz ciągnął do tego typu rzeczy. Jeżeli natomiast lubiłeś gry komputerowe (czyli tak jak ja, bo z tego się potem wzięła oczywiście moja praca w e-sporcie) i na przykład masz rodziców, którzy dużo siedzą przy urządzeniach (na przykład ojciec programista, mama inżynier), to będziesz ciągnął bardziej w tę stronę.

Tak to zwykle jest z większością tego typu tematów – jest komponent genetyczny uruchamiany środowiskowo. Geny mamy wszyscy, ale to, które geny się aktywują i które, powiedzmy, dadzą o sobie znać w późniejszym wieku, zależy od Twojego środowiska.

Wiesz… Ty widzisz swoje dzieci, ale to nie jest zasada. Wydaje mi się, że ludzie jednak troszeczkę mogą mieć do tego predyspozycje albo nie. Mogą być ludzie, którzy lubią rozwiązywać ciągłe zagadki logiczne. Programowanie to jest ciągłe kminienie, szukanie informacji, próba ich zastosowania, metoda prób i błędów. Ja na przykład mam taki styl pisania, że piszę zanim pomyślę. Patrzę: działa, nie działa? OK, dobra, lecę dalej albo do tego wracam. Naprawdę bardzo to lubię.

Wróćmy jeszcze do tematu decyzji. Jak to zrobić, żeby podjąć tę decyzję o przebranżowieniu się?

Warto, żebyś zdawał sobie sprawę z tego, z czym to się wiąże. O czym mówiliśmy? O tym, że to jest tytaniczna praca, praca umysłowa, wykonywana solo. Najlepsi programiści albo tacy, którzy rzeczywiście zarabiają te mityczne piętnaście / dwadzieścia / dwadzieścia pięć K, to

są ludzie, którzy jednak ciągle się dokształcają. To też jest bardzo istotne, dlatego że pojawia się coraz więcej narzędzi.

Kolejna rzecz, na którą szczególną uwagę muszą zwrócić osoby z rodzinami (tak jak to było u mnie) – na początku pensja juniorska jest taka sama jak wszędzie w IT, przynajmniej moim subiektywnym zdaniem. Po prostu progresja pensji w IT jest duża, szybsza niż w ogromnej większości branż.

Warto też zdawać sobie sprawę z plusów tego zawodu. Te plusy są dla każdego inne. Choćby właśnie ta szybka progresja zarobków – że w miarę szybko się dojdzie do zarobków relatywnie wysokich w stosunku do innych (bo wysoki/niski to są pojęcia, które nic nie mówią, jeżeli nie masz jakiegoś punktu odniesienia).

Masz możliwość pracy zdalnej jak na przykład ja – pracuję teraz w większości z domu. Masz praktycznie nieskończone możliwości rozwoju w kierunku, który Cię zainteresuje. Mnie na przykład teraz mega zainteresowała architektura mikroserwisowa i w tym kierunku naprawdę chciałbym się w przyszłości rozwijać. Możesz być DevOpsem, sieciowcem, specjalistą cyberbezpieczeństwa i tak dalej. Rynek pracy dalej jest tutaj doskonały.

I jeszcze taka rzecz, która mi się najbardziej w tym podoba: ja naprawdę mam strasznie wielką satysfakcję z tego, co robię. Naprawdę lubię wykonywać rzeczy własnoręcznie. Wyobraźcie sobie, że przez dziesięć lat bycia PM-em ja nic nigdy nie zrobiłem. Ja wyłącznie starałem się motywować i kierować procesami w taki sposób, żeby ktoś coś zrobił. Teraz pierwszy raz w życiu pracuję w taki sposób, że sam własnoręcznie muszę coś wykonać. Dla mnie jest to źródło ogromnej satysfakcji.

Jeżeli dobrze zrozumiałem, to to przeważyło – satysfakcja była najważniejszym elementem, który zadecydował o tym, że zrezygnowałeś z pracy marzeń na rzecz programowania.

Satysfakcja jest wtedy, gdy coś Ci wyjdzie, ale wydaje mi się, że u mnie to była radość z pisania. Nie wiem nawet, czy to czyni mnie jakoś szczególnie wyjątkowym, bo taką radość można też sobie wytworzyć, o czym na pewno za chwileczkę powiemy. Wydaje mi się, że u mnie to bardziej była radość.

Widzę też z własnego doświadczenia, że ta radość powoduje, że jesteśmy w stanie przeskoczyć wszystkie problemy, które się pojawiają na naszej ścieżce nauki, przebranżowienia i tak dalej. Pewnie będziemy jeszcze o tym wspominać.

Tak, dokładnie. Spotykałem się też z bardzo różnymi mindsetami, na przykład: *nie umiem czegoś, więc tego nie róbmy*. W naszej firmie jest natomiast tak, że gdy ktoś przychodzi do działu IT i mówi: *zróbcie coś, czego nigdy wcześniej nie robiliście*, to my na to: *dobra, nauczymy się i zrobimy*. I jedziemy z tematem. Nie ma dla nas tematów, których byśmy nie ogarnęli.

Ciekawy jestem, czy często tak jest w branży IT. Zauważyłem, że zazwyczaj się mówi *można*, tylko wszystko zależy od tego, ile mamy czasu, zasobów i tak dalej. Nie ma takiego stwierdzenia, że *nie*. Nie wiem w ogóle, jak nazwać takie podejście.

Tak, ale to wynika z samej specyfiki zawodu. Zobacz: na czym polega taka codzienna praca programisty? Siedzisz, odpalasz edytor, masz wykonać jakieś zadanie, które zostało Ci odgórnie narzucone przez

biznes, i musisz dowiedzieć się, w jaki sposób to zadanie zrobić. Rzadko są sytuacje, że znasz na pamięć wszystkie słowa kluczowe, komendy, biblioteki, z których korzystasz, i tak dalej. Musisz sobie odświeżyć tę pamięć dokumentacjami, tutorialami, artykułami blogowymi na ten temat, Stack Overflowem. Musisz skorzystać z pomocy Copilota albo znaleźć informacje na temat zupełnie nowych rzeczy, z którymi wcześniej nie miałeś styczności.

Robisz na przykład integrację płatności z Przelewami 24, a nigdy wcześniej tego nie robiłeś. Musisz zrobić subskrypcyjne płatności blikiem albo zintegrować się z jakimś ERP-em, który na przykład ma słabą dokumentację. Codzienną pracą programisty jest dowiadywać się nowych rzeczy, robić sobie syntezę informacji i przelewać ją potem w kod, który działa. Też więc odnoszę takie wrażenie, że programiści raczej tacy są… Przynajmniej dobrzy programiści, bo na pewno są też tacy, którzy mówią, że się nie da, bo im się nie chce. Jestem pewien, że są takie osoby, chociaż ja takich osób spotkałem naprawdę bardzo niewiele i chyba nigdy z nimi nie pracowałem.

Znowu dygresja: zastanawiałem się, czy nie nauczyłeś się programowania właśnie dlatego, żeby się dowiedzieć, czy programiści mówią prawdę, gdy twierdzą, że *się nie da*.

Wiesz co, tak troszeczkę jest. Osoby nietechniczne, czyli Scrum Masterzy, Product Ownerzy (no, Product Ownerzy zwykle są w miarę techniczni), Project Managerowie czy... Te różne etykiety oznaczają czasami te same rzeczy, a czasami zupełnie inne w zależności od struktury firmy. Bardzo często biznes jest skazany na to, co mu

powiedzą osoby techniczne, ponieważ on tego najzwyczajniej w świecie nie rozumie.

Do pewnego momentu to oczywiście działa, ale potem masz na przykład takie sytuacje, jak ostatnio widziałem na jednej z grupek dla początkujących. Ktoś opublikował post, że musi się nauczyć programować, wymienił technologie, które ma w swoim projekcie, i powiedział, że jest właścicielem firmy. Widzi, że jego podwykonawcy trochę nie mówią mu prawdy. W jaki sposób czymś takim zarządzić?

Ja wtedy akurat starałem się mu to odradzać. Powiedziałem: słuchaj, to jest Twój czas, który możesz poświęcić na budowanie swojego produktu. Byłem PM-em przez dziesięć lat i wiem, co czujesz, widząc ludzi, którzy mówią Ci, że się czegoś nie da. Potem ich zagadujesz, troszeczkę stawiasz pod ścianą, a oni mówią, że jednak się da. Też miałem wiele takich sytuacji ze swoimi podwykonawcami, tylko akurat nie w IT, ale naprawdę bardzo dobrze znam ten proces i wiem, jak to wygląda, wiem, jak trudno coś komuś przetłumaczyć. Czasami pokonać mur stworzony z osobowości jest naprawdę najtrudniej, bo te mury są najwyższe.

Ja wiem, jak to zrobić, ale może są inne drogi. Może zatrudnij zewnętrznego audytora, może weź sobie kogoś do pomocy, kto Ci będzie na przykład robił konsultacje przez pięć-dziesięć godzin w miesiącu. To bardziej Ci się opłaci, bo jeżeli będziesz się uczył tych technologii, które wymieniłeś (nie pamiętam, co on tam miał w stacku, na pewno był JS/TS, może jakiś Python), to zanim ogarniesz to w stopniu wystarczającym, żeby móc rzeczywiście rozliczać developerów, to naprawdę przegapisz dużo okazji na rozwój własnego biznesu.

A może to jest nowa osoba w branży IT? Nauczy się programowania i potem będzie programistą, a nie będzie robić swojego biznesu jak do tej pory.

Może tak chciał, może po prostu chciał oszukać wszystkich, chciał się za to zabrać, bo widział, że to jest fajne, perspektywiczne zajęcie, gdzie się dużo zarabia, można pracować zdalnie właściwie z każdego miejsca na świecie, w każdej branży (bo możesz być programistą w rolnictwie, w rozwiązaniach chmurowych, w rządzie, w gazecie, w medycynie, możesz się zajmować wszystkim). Może tak, może po prostu wszystkich oszukał, a chciał sam to robić.

Może tak, może tak.

Fajne zajęcia, nie zdziwiłbym się.

Wracając do postów na Facebooku – wspomniałeś, że jesteś wielkim fanem, że pomagasz tam ludziom. Może wybierz trzy najczęściej pojawiające się pytania i spróbuj na nie odpowiedzieć, a jak odcinek się przyjmie, to zrobimy tę większą listę, o której wspominaliśmy.

Dobra, no to jedziemy. Jednym z częściej pojawiających się pytań – i to jest megaparadoksalne – jest: *jak zacząć?* To jest paradoksalne z tego względu, że – tak jak mówiliśmy wcześniej – praca programisty to jest praca samodzielna, non-stop wyszukujesz informacje, syntezujesz je. Ścieżek dla początkujących jest naprawdę mnóstwo. Zacząć uczyć się programowania jest najłatwiej, dlatego że na ten temat masz najwięcej materiałów i są to materiały w popularnych technologiach, megadobrze

ustrukturyzowane. Znajdziesz mnóstwo gotowych ścieżek rozwoju, z którymi zacząć jest naprawdę łatwo.

Masz mnóstwo informacji, które tylko leżą i czekają, żebyś po nie sięgnął, a Ty pytasz na grupie, jak zacząć, bo tego nie wiesz. I właśnie to jest moja pierwsza rada dla ludzi. Sam zrób research na temat tego, co dokładnie chcesz robić. Można to zrobić na kilka sposobów: przetestować kilka języków, zobaczyć w którym się dobrze pisze; spojrzeć na to, w których technologiach zarabia się najwięcej; przemyśleć, co chcemy pisać, jaki typ rzeczy – czy aplikacje mobilne, czy aplikacje webowe, czy na przykład stawiać stronki na WordPressie, bo wtedy dosyć łatwo można założyć swoją działalność (mimo że jest to raczej niskomarżowy biznes, to też niektórzy to bardzo lubią).

Sięgnij więc po te informacje, posiedź sobie w tym kilka dni, poczytaj dużo artykułów, pooglądaj filmy na YouTubie – bo to wszystko jest. Praca indywidualna przez pierwszych kilka miesięcy to jest coś, co polecam, ponieważ będziesz przez to bardzo dobrze przeprowadzony.

Jeżeli chcesz jakiś konkret, to tuż po wyborze technologii… I tutaj akurat troszeczkę zaspoiluję: moim zdaniem warto zabierać się za technologie, w których jest najzwyczajniej w świecie dużo pracy, ponieważ w takich technologiach szuka się juniorów. To też zresztą jest często pojawiające się pytanie. W takich językach, jak na przykład Scala, Go, Rust, są megawspaniałe projekty, bardzo atrakcyjnie finansowo, ale w nich nie szuka się juniorów. To nie jest dobry wybór dla juniora, ponieważ najprawdopodobniej po prostu nie znajdzie w tym pracy. Musisz znaleźć tę pierwszą pracę w technologii, która jest popularna: Java, PHP, TypeScript, C# czy tam .NET i tego typu rzeczy.

Wybierz sobie jakąś technologię, kup na przykład na platformie Udemy jeden kurs, który zawsze kosztuje pięćdziesiąt złotych, i rób ten kurs, pracuj indywidualnie przez kilka miesięcy. Jak już opanujesz podstawy, to czas na kolejny krok. I ten kolejny krok niech będzie naszym następnym pytaniem. Co zrobić, gdy już troszeczkę ogarniasz? Masz bardzo dużo wiedzy na dany temat, potrafisz na przykład postawić SPA w Next.js, potrafisz postawić apkę CRUD-ową, napisać jakąś prostą aplikację mobilną i nie wiesz, co robić dalej.

Teraz przechodzimy do drugiego pytania. Co robić dalej, gdy już trochę ogarniesz? Jak wykonać ten krok typu: *Zrobiłem pi razy oko 40-50% swojej pracy. Jak kontynuować?* Wtedy, moim zdaniem, najlepszym drugim krokiem jest wziąć sobie mentoring na przynajmniej kilka miesięcy. Nawet chyba nie *przynajmniej*, po prostu na kilka miesięcy, dlatego że dojdziesz do pewnego poziomu, gdzie będziesz miał tak zwany *analysis paralysis*. To takie coś, gdy troszeczkę dreptasz w miejscu, troszeczkę nie wiesz, gdzie iść dalej i jak iść dalej, dlatego że te ścieżki dla początkujących Ci się pokończyły. Trochę nie masz ochoty zajmować się takimi rzeczami jak na przykład testy czy GraphQL, bo to jest naprawdę mocno wymagające poznawczo i po prostu możesz być już tym wszystkim zmęczony. Jesteś w tym procesie od kilku miesięcy.

Wtedy właśnie rolą mentora jest to, żeby Ci pokazać, gdzie masz braki, czym masz się zająć. Mentor może podesłać Ci dokładne materiały (bo to jest jego praca) i indywidualnie pokaże Ci, na czym musisz się skupić. Przeprowadzi Cię przez ten proces i w tym momencie pozbędziesz się tego *analysis paralysis* – dlatego że ktoś Ci pokaże bardzo jasną drogę. W dodatku będziesz zmotywowany chociażby takim prostym faktem, że najzwyczajniej w świecie temu mentorowi płacisz. Jesteś odpowiedzialny

za pieniądze, które wydajesz. Chyba nie chcesz, żeby te pieniądze szły w błoto, prawda? To było więc drugie pytanie.

Może ja tylko wejdę w słowo, żeby nie było, że to jest odcinek sponsorowany. Mateusz ze mną nie współpracował. Żeby nie było tak, że opowiadamy o superlatywach mentoringu, bo Ty go u mnie kończyłeś.

Co więcej, ja współpracowałem z Twoją konkurencją.

Tak jest, zgadza się. Chciałem to zaznaczyć, możemy lecieć dalej.

Bardzo często pojawia się na przykład pytanie w stylu: *Dlaczego nie mogę dostać pracy? Przecież już tyle zrobiłem, już tyle potrafię*. Ledwo wczoraj widziałem jedną dziewczynę, która wcześniej pracowała w jakiejś firmie jako pani od PR-u, więc widać było, że bardzo dobrze potrafi pisać. Napisała cały poemat o tym, jakich to ona kursów nie ukończyła, jakie robi super rzeczy. Jak się w to wczytać, to okazało się, że ona rzeczywiście zrobiła bardzo dużo roboty, ale to jeszcze jest za mało jak na dzisiejszy rynek. Kilka osób dokładnie jej wymieniło, czego jej jeszcze brakuje i ona… Nie wiem, chyba po prostu nie wytrzymała konstruktywnej krytyki, bo pousuwała wszystkie posty ze wszystkich grup (wcześniej wrzuciła ten post na kilku różnych grupach). Więc tego nie róbcie. To jednak też uzmysławia, jak dużo pracy ludzie w to wkładają. Naprawdę ktoś uczy się rok / dwa lata i nie może tej pracy dostać.

I teraz: *dlaczego nie mogę dostać pracy?* Nie da się oczywiście odpowiedzieć na to pytanie dokładnie, ale da się zrobić sobie taką checklistę rzeczy, które mogą być przyczyną – bo Ty indywidualnie

najprawdopodobniej popełniasz któryś z tych błędów. Dlaczego mówię, że na pewno popełniasz jakiś błąd? Bo rynek nadal jest chłonny, nadal szuka juniorów. Jeżeli zrobicie jedną prostą rzecz… (*Wymyślił jedną prostą rzecz, jak dostać pracę w IT. Eksperci go nienawidzą*). Jeżeli zrobicie jedną prostą rzecz, to Wy tę pracę dostaniecie – po prostu nauczcie się lepiej od innych i potrafcie to zaprezentować. Tyle. To jest wszystko. Bądźcie lepsi niż 90% pola, co nie jest wcale takie trudne. Wiem, że liczba 90% jest troszeczkę przerażająca, ale biorąc pod uwagę, jak wiele osób chce wejść do branży bez elementarnych kompetencji w tej branży, to naprawdę nie jest to aż takie trudne zadanie.

Rozumiem jednak, że jesteście już po roku czy dwóch latach intensywnej pracy. Naprawdę jest Wam ciężko, bo nie możecie tej pracy dostać. Słuchajcie, przelećcie sobie taką checklistę. Może na przykład macie CV, które jest bardzo niedbale napisane. Może jest brzydkie? Ludzie niestety oceniają oczami. Może macie tam błędy językowe – dajcie to komuś do sprawdzenia, jakiemuś nerdowi z filologii polskiej, Waszemu znajomemu. Może rozpisujecie się na maksa na temat zajęć, którymi się zajmowaliście przed IT, co szczerze powiedziawszy, mało kogoś interesuje. Pracodawców raczej interesuje, co potraficie w obrębie stanowiska, na które będziecie aplikować. Napiszcie oczywiście, że pracowaliście gdzieś wcześniej, ale nie poświęcajcie temu aż tyle miejsca. To dosyć częsty błąd.

Może ktoś nie ma na przykład dobrych projektów na GitHubie. Nie wiem Mateuszu, co o tym sądzisz, ale moim zdaniem projekty to jest czynnik numer jeden, który odróżnia prejuniora, który dostanie pracę jako junior, od takiego, który nie dostanie. To nie musi być dużo projektów. Niech to

będzie nawet jeden projekt, ale duży, wyklikany z czystą historią branchowania, z jakąś strategią branchowania, z zachowanymi wszelkimi zasadami czystego kodu. Niech tam będzie dokumentacja – głupi pliczek w README. Niech w tej dokumentacji będzie chociaż jeden diagram (dlatego że tak jak wspominałem: ludzie oceniają oczami), niech będzie demówka online, żeby ktoś nietechniczny mógł to sobie przeklikać albo ktoś średniotechniczny mógł wysłać Postmanem zapytaniem do API.

Czy się zgodzisz, że projekty to jest taka rzecz numer jeden?

Zdecydowanie tak, chociaż wolę podchodzić do tego w troszkę inny sposób. Wolę, żeby było sześć projektów niż jeden duży, nawet mniejszych. Może być tak, że w tym jednym dużym coś się nie spodoba i zostaniemy odrzuceni, a jak tych projektów będzie kilka, to może ktoś zajrzy i zobaczy, że tu zrobiliśmy tak, tu zrobiliśmy inaczej, coś uwiedzie go w tym jednym, konkretnym projekcie i może będzie chciał z nami porozmawiać. Wolę więc opcję „dywersyfikacji".

Odnośnie jeszcze do tych projektów z GitHuba, to całkiem niedawno, bo dzisiaj rano, przejrzałem sobie GitHub osoby, która uczy się już przez rok czy nawet półtora. Pewnie jeszcze będziemy mówili o kwadracikach na GitHubie. Zobaczyłem, że tych kwadracików było dosłownie cztery przez półtora roku – na dodatek tylko po jednym dniu. Nie chodzi o to, żeby wypełniać te wszystkie kwadraciki, ale to też pokazuje, ile ktoś zrobił albo jak bardzo nad tym pracował. To od razu pokazuje, że pewnie jest tego za mało – i

faktycznie, gdy zajrzałem do tych repozytoriów, to niewiele się tam działo.

Ja bym powiedział, że co gorsza, to po prostu pokazuje, że ktoś nie potrafi pracować z Gitem, ktoś nie wykorzystuje Gita w tym celu, w którym on został stworzony. Zabawne, że o tym wspominasz, bo dokładnie to samo miała tamta dziewczyna. Miała chyba ze trzy czy cztery commity przez cały ostatni rok i na przykład ostatnim commitem, który był zrobiony kilka tygodni temu, było wgranie dwudziestu dwóch repozytoriów na jedno kopyto.

To pokazuje, że ktoś nie wykorzystuje Gita, nie potrafi z nim pracować, bo po prostu z nim nie pracował. A nawet jeżeli z nim pracował, no to co ma sobie pomyśleć rekruter? Rekruter nie ma na to dowodów, rekruter tego nie wie, musisz mu to pokazać. Dlatego to jest bardzo ważne.

À propos tych sześciu projektów, to jeszcze raz wracamy do tego, co oznacza duży/mały. Wydaje mi się, że jeżeli masz sześć projektów, to one nie są małe, tylko średnie. Może masz rację. Sześć średnich projektów z różnych technologii albo takich, które robią różne rzeczy, na przykład jeden z Reacta, jeden z Expressa, jeden z Nesta (mówię, oczywiście, o tych w swoim ekosystemie, dla Pythoniarza albo dla Javowca to może być zupełnie co innego), jeden back-endowy, jeden front-endowy – żeby pokazać, że coś potrafisz. Jeden na przykład taki, gdzie wykorzystujesz jakiegoś message brokera. Wtedy może rzeczywiście te sześć projektów ma większy sens. Ja natomiast dalej będę się upierał, że to raczej pewnie nie są małe projekty. Jakbym je zobaczył, to powiedziałbym więcej.

Jasne.

To co, lecimy dalej z listą?

Tak jest, ale chyba masz tu jeszcze parę punktów odnośnie do tego, dlaczego nie można dostać pracy.

Tak, dokładnie. Słuchajcie, zróbcie sobie nawet najprostszy audyt swoich social mediów. Czy macie dobrze ustawionego LinkedIna? Bo miałem takie sytuacje, że ktoś składa CV do firmy, w której akurat pomagam w rekrutacji, wchodzę na Facebooka kandydata i widzę: *Szlachta nie pracuje*. Ludzie, od razu lecicie na blackliste. Zróbcie sobie prosty audyt sociali, zobaczcie, czy nie macie w swoich profilówkach na przykład czegoś bardzo obraźliwego i tym podobnych rzeczy, bo takie sytuacje się zdarzają. Miałem dokładnie takie sytuacje. Nie chcecie sobie tego robić.

Bardzo częste jest to, że ktoś rzeczywiście dużo przepracował i naprawdę myśli, że potrafi napisać dużo rzeczy. OK, spoko, ale czy to jest rzeczywiście wysokiej jakości kod? Czy znasz zasady pisania czystego kodu? Czy znasz jakieś elementarne zasady architektoniczne? W jaki sposób umieszczać pliki w dobrych folderach? Czy znasz wszelkie skrótowce typu: SOLID, KISS, DRY? Czy zastosowałeś jakieś wzorce projektowe? Czy gdy mówimy na przykład o JavaScripcie, to piszesz zarówno obiektowo, jak i funkcyjnie i masz to pokazane w swoich repozytoriach?

To jest więc kolejna rzecz: możesz myśleć, że potrafisz pisać dobry kod, ale tak naprawdę nie skonsultowałeś się z żadnym zawodowcem albo mentorem (tutaj jeszcze raz wychodzi rola mentora), albo nawet kimś, kogo sobie po prostu znalazłeś i kto najzwyczajniej w świecie zrobił audyt Twojego kodu, ocenił i pomógł, bo ma dobre serduszko. Możesz

myśleć, że piszesz dobry kod, a on wcale taki nie jest. To może być przyczyna, a Ty o tym nie wiesz – bo przecież nie wiesz tego, czego nie wiesz.

Kolejna przyczyna: może wysyłasz mało CV, może wysyłasz CV na przykład wyłącznie pięć CV na miesiąc. Cytując Matthew McConaugheya (z *Wilka z Wall Street*), który skradł cały film jedną sceną: *You gotta pump those numbers up. Those are rookie numbers (...)*. To są małe liczby, musisz tego zrobić więcej. Podejrzewam, że akurat masz na ten temat coś do powiedzenia, ale Ci jeszcze nie dam wejść w słowo.

Jeszcze do tego wrócimy.

Dokładnie. Może na przykład piszesz w takich właśnie językach, jak wcześniej wymieniłem: Scala, Go, Rust. One są super, ja też w przyszłości chcę się nauczyć jakiegoś niszowego języka, dlatego że mnie to bardzo interesuje – żeby tworzyć te swoje mikroserwisy back-endowe jako bardziej performatywne. Ale to nie jest dobry język dla juniora.

Może chcesz zacząć od jakiejś bardzo niszowej subspecjalizacji, na przykład InfoSec albo DevOps, albo sieci – czegoś, w czym nie ma aż tyle pracy dla developera. Tak czy inaczej, ludzie, którzy przy tym pracują, bardzo często przychodzą do tego już będąc developerami. Zostanie developerem to jest naprawdę dobry start w IT, według mnie nawet najlepszy.

Może masz na przykład jakiś bardzo wąski stack technologiczny. Albo o, częsta rzecz w JavaScripcie: ludzie znają MERN Stack, ale nigdy nie

mieli do czynienia z relacyjnymi bazami danych, które jednak w środowiskach komercyjnych są częściej stosowane niż bazy nierelacyjne z bardzo wielu różnych powodów.

Może na przykład w ogóle nie znasz testowania. Może chcesz pisać back-endowo w JS-ie – tak jak ja to robię – i nie znasz Nesta, który jest najpopularniejszym tego typu frameworkiem. Dlatego przykładowo nie możesz znaleźć pracy – bo nie masz dużego projektu z Nesta.

Może chcesz pracować tylko zdalnie. Często ludzie mają taki wymóg. Tutaj niestety trochę muszę sprowadzić ich na ziemię. Praca zdalna w IT oczywiście istnieje, ale raczej nie wtedy, gdy przychodzisz do firmy i jesteś totalnym świeżakiem. Raczej praca zdalna jest zarezerwowana od poziomu bardzo słaby mid wzwyż. Wówczas już rzeczywiście raczej nie powinno być z tym problemu, wcześniej będzie raczej ciężko.

Może nie chcesz zejść z oczekiwań finansowych. Może szukasz wyłącznie takiej pracy albo przyjąłbyś wyłącznie taką pracę, w której od początku swojego zatrudnienia dostaniesz dziesięć tysięcy na rękę. Jako junior? No, niestety. Wydaje mi się, że też o tym wcześniej rozmawialiśmy: pierwsza pensja w IT jest zupełnie normalna, po prostu progresja jest szybsza.

Może słabo się sprzedajecie na rozmowach kwalifikacyjnych. Ja na przykład uważam, że jestem w miarę wygadany, ale uwierzcie mi – też udało mi się zawalić jedną rozmowę rekrutacyjną wyłącznie tym, że bardzo źle do niej podszedłem. Nawet więc taka osoba jak ja potrafi coś zawalić, a przecież przez dziesięć lat pracowałem z klientami, sprzedawałem i wykonywałem projekty za siedem cyfr… (nie, dobra, za

prawie siedem cyfr, bo chyba do siedmiu cyfr nigdy nie doszedłem) i raczej – wydaje mi się – potrafię rozmawiać z ludźmi.

Może potrzebujecie na przykład wykonać sobie jakieś testowe rozmowy rekrutacyjne z waszym znajomym, może z waszą dziewczyną. Niech ona będzie rekruterką i się z tego przygotuje, a wy bądźcie prejuniorem, który chce dostać pracą jako junior. Przećwiczcie sobie to, te wszystko rzeczy naprawdę są do wyćwiczenia.

Ewentualnie pomyślcie sobie, co dla was gra, co dla was nie gra. Dla mnie na przykład bardzo zagrało to, że znałem cykl życia wytwarzania oprogramowania. Mnie było więc na pewno na tych rozmowach rekrutacyjnych troszeczkę łatwiej, bo wiedziałem na przykład, o co zapytać. Widać było, że w miarę ogarniam, jak się wytwarza oprogramowanie. Poszukajcie takich zalet u siebie.

Może na przykład jesteście w stanie mocniej zejść z oczekiwań finansowych, ale wtedy przy rozmowie możecie zaznaczyć, że: *spoko, popracuję u Was na przykład trzy miesiące, sześć miesięcy za taką pensję, ale potem na pewno będziemy rozmawiać o dużej podwyżce*. Poszukajcie u siebie jakichś zalet.

Podsumowując, wydaje mi się, że odpowiedź na pytanie: *dlaczego nie mogę dostać pracy?* jest niemożliwa do udzielenia, dlatego że u każdego to jest indywidualnie. Ale jak widzicie – nawet wyłącznie po tych czynnikach, które wymieniłem – tych przyczyn może być dużo. Jest bardzo dużo różnych składowych. Nawet brak w jednej takiej składowej może sprawić, że nie znajdziecie pracy przez dobrych kilka miesięcy i zawalicie sobie bardzo dużo szans. Naprawdę warto więc pytać,

zasięgać informacji od kogoś bardziej doświadczonego, kto pokaże Wam, gdzie macie te braki.

Na pewno jest jakaś przyczyna, ponieważ – i tutaj naprawdę będę się upierał – jeżeli nauczycie się lepiej od innych i to zaprezentujecie w sposób przystępny dla rekrutera nietechnicznego oraz technicznego, to tę pracę dostaniecie. Po prostu nie ma innej opcji.

Może zwróciłbym uwagę na dwie rzeczy. Po pierwsze każdy z tych elementów nie zajmuje miesiąca, tylko raczej parę dni, dlatego lepiej się zastanowić, czy nie warto ich wdrożyć. Po drugie: nie oczekujmy zmian, jeżeli my nie wprowadzimy żadnych zmian. Jeśli widzimy, że wysłaliśmy dwadzieścia CV i nie ma odzewu, to już mamy sygnał, że coś powinniśmy zmienić. Jeżeli rozesłaliśmy sto CV, nic nie robimy i myślimy, że dostaniemy pracę, to pewnie tak nie będzie – mamy już sto razy potwierdzone, że jej nie dostaliśmy.

Trzeba więc – pewnie iteracyjnie jak w programowaniu – coś zmieniać, poprawiać. Dzięki temu będzie coraz lepiej i łatwiej znaleźć tę pracę. Pamiętajmy o tym, żeby zmieniać, poprawiać, analizować. To chyba jest coś, na co szczególnie zwróciłbym uwagę.

Najgłupszy dowcip ze środowiska chmurowego, jaki kiedykolwiek słyszałem: grałeś kiedyś w Fargate'a?

Nie.

Fargate to jest usługa AWS-a, która po prostu nazywa się jak Far Cry. W jednej z części Far Cry'a był taki antybohater (w ogóle Far Cry ma super antybohaterów, naprawdę świetnie napisane historie), który często

głównemu bohaterowi (czyli Tobie, graczu) zadaje pytanie: *czy znasz definicję szaleństwa?* Szaleństwem jest, jeżeli robisz cały czas to samo i oczekujesz innych wyników. Święta racja, nic dodać, nic ująć.

A czy myślisz, że tak może być, że podczas poszukiwania pracy nie są wdrażane zmiany i poprawki, przez co cały czas jest jak jest?

Sto procent pewności. Jeżeli walisz głową w mur, a to nie daje efektów, to ja rozumiem – możesz po prostu przez jakiś czas mieć najzwyczajniej w świecie niefart. Na przykład moje poszukiwania pracy… Mam to dokładnie spisane: w przeciągu dwóch miesięcy i pięciu dni wysłałem sto dwa CV. Dwa pierwsze zaproszenia na rozmowy rekrutacyjne miałem bardzo szybko. Gdybym nie zawalił jednej z tych rozmów rekrutacyjnych, to być może szukałbym pracy dosłownie przez tydzień albo dwa. Tę jedną rozmowę zawaliłem technicznie, bo nie miałem dostatecznie dużo skillów. Potem patrzyłem na LinkedInie na ludzi, którzy pracowali w tej firmie, i myślałem sobie: *kurczę, ale on był dobry* albo *ale ona była dobra, zrobili coś, co mi się technicznie nie udało*.

Potem bardzo długo nic, ale po miesiącu wpadło kilka ofert, więc tutaj wychodzi słaba intuicja statystyczna, którą ludzie mają. Musisz patrzeć sobie na długi okres – co się w nim dzieje – a nie wyłącznie: *Jezu, no, dlaczego codziennie nie wpada mi jakieś zaproszenie?*, bo to będzie tak, że będzie Ci wpadać jedno zaproszenie na rozmowę na miesiąc, a w kolejnym miesiącu możecie Ci wpaść dziesięć. Patrz więc na ten długi okres.

Jeżeli natomiast w bardzo długim przedziale czasu, np. powyżej trzech miesięcy albo powyżej sześciu miesięcy, w ogóle nie dostajesz

zaproszeń na rozmowę, to coś zrobiłeś/zrobiłaś nie tak. Walisz głową w mur. Idź po pomoc, skorzystaj z niej.

Trochę się rozpędziliśmy z tymi pytaniami, więc wróćmy jeszcze do początku.

Co trzeba wziąć pod uwagę przed podjęciem decyzji o przebranżowieniu? Często sprawa się komplikuje, jeżeli mamy inne obowiązki: wymagająca praca, rodzina i tak dalej.

Wydaje mi się, że rodzina to jest naprawdę bardzo ważny czynnik. Ja się przebranżawiałem… O właśnie, tego nie mówiliśmy. Ja ostatecznie zmieniłem zawód, mając trzydzieści pięć lat, małe dziecko, żonę, bardzo dobrze płatną pracę i karierę, która stała przede mną otworem, a jednak zdecydowałem się na to przebranżowienie. Jeżeli masz rodzinę, to jestem zdania, że nigdy nie zmieniasz pracy sam – chyba że naprawdę po prostu nie uczestniczysz w życiu rodzinnym (trudno mi oceniać takich ludzi, nie wiem, jak można tak robić).

Jeżeli jednak chcesz dobrze wychowywać swoje dziecko lub dzieci, być dobrym mężem i jeszcze się oprócz tego przebranżowić, musisz wcześniej się dogadać z rodziną, bardzo konkretnie powiedzieć: *Słuchajcie, mam przynajmniej rok tytanicznej pracy do wykonania, będzie mi potrzebny czas na naukę*. Umówmy się na tyle godzin w tygodniu w takie dni. Bądźmy oczywiście w tym w miarę elastyczni – jeżeli coś się stanie, będzie potrzebna moja obecność, no to jasne.

Mnie by się nigdy nie udało, gdyby nie moja żona – do niej należy połowa zasług przy moim przebranżowieniu. Wzięła ona na siebie przez bity rok czasu szacuję… 75% opieki nad naszą córką.

Może warto powiedzieć, że całkiem niedawno urodziły Ci się bliźniaki, a mimo to byłeś w stanie przygotować się do tej rozmowy. Mówię to dlatego, żeby pokazać, że można, że zawsze łatwiej sobie wymyślić wymówkę, niż po prostu się postarać i zorganizować.

To nie mogłoby być bardziej prawdziwe. Bez rodziny Ci się nie uda, ale zawsze są jakieś sposoby. Zadajmy sobie takie teoretyczne pytanie: *Czy rzeczywiście są rzeczy, których nie przeskoczysz? Czy na przykład możesz mieć tak złe warunki, że nie dasz rady?* Wydaje mi się, że wszystko zależy od tego, jak mocno jesteś zmotywowany, jak wiele jesteś w stanie na to poświęcić, ale takie sytuacje prawie nie zachodzą.

Ja może podam taki trudy przykład. Nie znam całej historii, ale miałem przyjemność uczyć osobę, która była nieuleczalnie chora. Nie zostało jej dużo czasu, a i tak postanowiła nauczyć się programować, żeby móc pracować w tym zawodzie. To już jest taki, myślę, ciężki przypadek. Wiesz, pewnie chciałbyś wykorzystać ten czas inaczej, a akurat ta osoba była bardzo zmotywowana, żeby nauczyć się programować.

No właśnie, to jest też taka zabawna sprawa. Ludzie po trzydziestce, nawet ludzie po czterdziestce, którzy się przebranżawiają, z jednej strony rzeczywiście mają dużo mniej luzu w organizacji swojego czasu, bo po prostu mają bardzo dużo obowiązków, ale jednak takie osoby są najczęściej turbozmotywowane. Jeżeli idziesz sobie na przykład patrzeć, jak ludzie grają w siatkę, to widzisz, jak taka osoba po

trzydziestce/czterdziestce się do tego przykłada. Jakoś ci ludzie potrafią się lepiej zorganizować. Wiem mniej więcej, dlaczego to jest, ale za bardzo wchodzilibyśmy w szczegóły.

Wracając do naszej głównej myśli: słuchajcie, najprawdopodobniej nie jesteście przygotowani na to, jaki to jest wysiłek. To, jak to robić, musicie ustalić ze swoją rodziną bardzo konkretnie. Jeżeli jednak nie macie rodziny, to świetnie. Wtedy umówcie się ze swoją dziewczyną (a nawet jeżeli jej nie macie, to umówcie się sami ze sobą), zróbcie sobie bardzo konkretny plan, co robić – kiedy, w jakim czasie, w jaki sposób. Wygooglujcie sobie metodę planowania SMART, która zakłada, że zaplanujecie określony czas, miejsce i tak dalej. Nie określajcie sobie może tylko czasu, w którym na pewno znajdziecie pracę, dlatego że (o tym jeszcze powiemy) to nie do końca zależy od Was.

Zadbajcie o takie rzeczy jak na przykład zdrowe ciało – dzięki temu będziecie w stanie dłużej utrzymać skupienie, będziecie w stanie skuteczniej się uczyć. Na pewno jeszcze wiele razy powiemy o tym w tym podcaście: zadbajcie sobie o takie duże trzy klocki, czyli sen, ćwiczenia i odżywianie. Poeksperymentujcie z jakimiś dietami, które są dobrze do Was dostosowane.

Zacznijmy jeszcze od innej rzeczy: *ja chcę nauczyć się programować, nie mam czasu ćwiczyć, będę tylko siedział przed komputerem*. Słuchaj, zadaj sobie pytanie, czy naprawdę nie masz czasu ćwiczyć. Stracisz mnóstwo czasu, gdy będziesz zmęczony, wiedza nie będzie Ci wchodzić do głowy, będziesz gorzej spać – właśnie przez ten brak ćwiczeń. Ćwiczenia to jest Twoja inwestycja czasowa w to, żeby skuteczniej się uczyć.

Myślisz, że nie masz czasu spać? Wolisz spać po pięć-sześć godzin, a dłużej się uczyć? Akurat na temat snu bardzo polecam wpaść na mojego GitHuba, bo tam jest dosyć zwięzła checklista rzeczy odpowiadających za Twoją higienę snu – pokrótce opisane, dlaczego to jest ważne, jaki to ma na nas wpływ itd. Jeżeli chodzi o to, w jaki sposób fizjologicznie się uczymy, to jest to dosyć prosty proces. W trakcie danej czynności oznaczacie sobie neurony (neuronów sobie raczej nie tworzymy, neurogeneza to jest taki śliski temat), oznaczacie sobie nowe połączenia między tymi neuronami. To jest taka synaptyczna neuroplastyczność. W trakcie snu to, co się działo w ciągu dnia, odtwarzacie sobie z prędkością kilkudziesięciokrotną. Powtarzam: robicie coś w ciągu dnia, oznaczacie sobie neurony, macie jakieś doświadczenie, które zostaje zapamiętane przez Wasz układ nerwowy, a w trakcie snu odtwarzacie to z prędkością kilkudziesięciokrotnie większą niż za dnia. Przez to uczycie się w trakcie snu (i tutaj widełki w zależności od tego, jaki to jest typ, czy jesteście zdrowi itd.) od dziesięciu do stu razy szybciej niż na jawie. Czy naprawdę nie masz czasu spać ośmiu godzin? Masz czas spać osiem godzin, dlatego że to mocno przyspiesza Twoją naukę.

A jeżeli chodzi o diety, to widziałem, że miałeś u siebie w podcaście Kamila Lelonka. Kamil Lelonek jest na przykład dużym fanem diet niskowęglowodanowych. Nie dajcie sobie wcisnąć, że na przykład jedna dieta jest dla Was dobra. Na każdą dietę da się znaleźć badania, które mówią, że ona jest najlepsza, a wszystkie inne są do kitu. Problem z takim podejściem jest taki, że każdy organizm jest inny i im bardziej zagłębiam się w temat, tym bardziej dochodzi do mnie, że nie ma złych diet, nie ma dobrych diet, po prostu musicie spróbować, co dla Was działa. Dla mnie na przykład działa dieta niskowęglowodanowa w

pierwszej połowie dnia, co pozwala mi nie być sennym popołudniami. Spróbujcie sobie czegoś takiego.

Może wtrącę tylko, żeby po prostu iść jakąś ścieżką. To jest chyba najważniejsze: nawet jeżeli nie będzie ona dla nas idealna, to i tak będzie działać lepiej, niż jakbyśmy nic nie zrobili. Warto więc podjąć próbę i to wdrożyć.

Dokładnie. Miejcie też świadomość, że po wszystkim, jak już się przebranżowicie, to satysfakcja jest naprawdę ogromna. Słuchajcie, to jest druga najtrudniejsza rzecz, którą zrobiłem w życiu i jedyna taka rzecz, którą zrobiłem świadomie; która została od początku do końca zaplanowana i zakończona sukcesem; która trwała tyle czasu i w którą włożyłem tyle pracy.

Nic nie przebije tej satysfakcji, jaką teraz mam z tego, że jestem developerem. To jest naprawdę najwspanialsze uczucie na świecie. Jedyna rzecz, którą – uważam – zrobiłem w życiu lepiej (ale głównie dzięki żonie) to jest to, jak dobrze wychowaliśmy najstarszą córkę. Bliźniaczki mają dopiero cztery tygodnie, więc jeszcze zobaczymy. Mam nadzieję, że uda nam się powtórzyć ten sukces.

I słuchajcie, być może najważniejsza rzecz… Wiem, że odpowiedź na to pytanie jest nieco długa, ale mam nadzieję, że nie masz nic przeciwko. Czy możemy sobie porozmawiać troszeczkę o tym, jak działają nawyki?

Jasne, zdecydowanie tak. Myślę, że już parokrotnie na ten temat w podcaście rozmawiałem, ale chętnie jeszcze raz do tego wrócę, bo to jest chyba często słowo klucz.

OK, słuchajcie, podejdę do tego troszeczkę od strony biologicznej, ale bez terminologii biologicznej, żeby Was nie zanudzać i żeby ten podcast nie trwał dwóch godzin. Chociaż i tak chyba idziemy na dwie godziny…

Tak czy inaczej, dobra wiadomość jest taka: układ odpowiedzialny za Waszą motywację jest naprawdę prosty do zrozumienia. To jest jeden z prostszych układów, które działają w mózgu, w stosunku do tego, ile rzeczy on robi i jak bardzo zarządza Waszym życiem. Mówię oczywiście o układzie dopaminergicznym.

Jak to działa? Wyobraźcie sobie huśtawkę – taką z dwoma siedzeniami na przeciwległych końcach. Huśtawka może się przechylać albo po lewej stronie do góry i wówczas z prawej w dół, albo po prawej do góry i po lewej w dół. Załóżmy, że jeden koniec tej huśtawki to przyjemność, a drugi – nieprzyjemność. W momencie, kiedy uruchamia się Wasz układ dopaminergiczny, czujecie przyjemność. Problem w tym, że huśtawka zawsze wróci do przeciwległego punktu po stronie nieprzyjemności. Zawsze jest to odbicie, ponieważ tak działa grawitacja i tak działa Wasz układ dopaminergiczny. Dlaczego? Chodzi o wyczerpanie się chemikaliów w Waszych synapsach. Mniejsza z tym.

To, co jest najważniejsze: to zawsze wróci. Moglibyście sobie teraz powiedzieć: *OK, słyszałem o tym. Układ dopaminergiczny. Przyjemność, tak. Jem czekoladę, czuję się przyjemnie. To jest układ odpowiedzialny za przyjemność*. Nie. To nie jest układ odpowiedzialny za przyjemność. Przyjemność to jest tylko środek do celu. A to, za co ten układ jest odpowiedzialny, to motywacja.

Jeżeli chcecie zadbać o swoją motywację do nauki, to zadbajcie o zdrowy układ dopaminergiczny. Co prawda inne rzeczy też na to

wpływają, ale to jest taki duży numer jeden. Jeżeli to ogarniecie, to tak naprawdę macie 90% tego, co Wam potrzeba.

Kiedy czujecie się najbardziej przyjemnie? W momencie, kiedy zachodzi coś takiego jak błąd przewidywania nagrody. Mam pytanie do Ciebie, Mateuszu. Czy miałeś kiedyś tak, że na przykład nie uprawiałeś sportu… Ty chyba jesteś sportowy, tak mi się wydaje, gdy na Ciebie patrzę.

Tak trochę.

Czy na przykład miałeś tak, że uprawiasz jakiś sport, stajesz sobie pewnego dnia przed lustrem i w ogóle się tego nie spodziewając, nagle widzisz: *kurczę, sylwetka mi się poprawiła, super*. Miałeś kiedyś taki moment?

Nie.

Nigdy nie miałeś.

Nie, nie miałem.

Dobra, ale wielu ludzi tak miało. Wtedy zachodzi coś takiego, że od razu chce Ci się wracać na siłownię. Ten błąd przewidywania nagrody będzie dla programistów bardzo ważny. W momencie, kiedy będziecie mieli nawyk codziennego siadania do pisania kodu, do uczenia się, do próbowania różnych rzeczy, to będziecie mieli ten błąd przewidywania nagrody. Pewnego pięknego dnia usiądziecie do swojego terminala, VSCode'a albo innego IDE, które stosujecie, i będziecie widzieli, że potraficie zrobić coś, czego nie potrafiliście zrobić wczoraj. Wtedy zachodzi błąd przewidywania nagrody.

Jak o czymś pomyślicie… A, chyba tego nie powiedziałem, a to jest najważniejsze. Już w momencie, kiedy wyłącznie o czymś pomyślicie (jeszcze zanim w ogóle coś zrobicie), Wasz układ dopaminergiczny strzela. On się uruchamia, ma potencjał czynnościowy, zachodzi czynność elektryczna pomiędzy neuronami i układ dopaminergiczny, mówiąc kolokwialnie, strzela, a Wy czujecie przyjemność.

Najprzyjemniej jest wtedy, kiedy jest błąd przewidywania nagrody, czyli kiedy nagroda jest większa niż to, czego się spodziewaliście. Nagroda oczywiście może też być niższa i wtedy Wasz układ dopaminergiczny na przykład strzela w wolniejszym tempie. Powiedzmy, że chcecie iść na randkę ze swoją dziewczyną. Będzie najlepszy wieczór ever, zaplanowaliście coś wspaniałego, ale tuż przed randką dostajecie SMS-a: *sorry, gościu, rzucam Cię*. I wtedy naprawdę czujecie się bardzo źle, ponieważ zachodzi błąd przewidywania nagrody z efektem netto ujemnym.

Jeszcze ostatnia ważna rzecz, taka fundamentalna: układ dopaminergiczny strzela z jakąś tam częstotliwością, która jest dla każdego jakby domyślna. Wtedy czujecie się po prostu OK. Macie motywację do robienia rzeczy, które są wysoce dopaminergiczne (oczywiście w dużym cudzysłowie), i macie mniejszą motywację do takich rzeczy jak na przykład sprzątanie mieszkania, ale też nie jest jakoś strasznie trudno się za to zabrać.

Układ dopaminergiczny przy błędzie przewidywania nagrody bardzo mocno wzrasta, a przy błędzie przewidywania nagrody, które jest netto na minus, może zwolnić do zera. Może w ogóle nie strzelać. To jest

bardzo patologiczna sytuacja. Naprawdę nie chcecie się doprowadzać do takiego stanu, a się da, o czym jeszcze za sekundę powiem.

I teraz tak: w jaki sposób zadbać o to, żeby być zmotywowanym? Żeby ten *baseline* dopaminy, taki punkt zero, był w miarę wysoki, żeby Wam się chciało robić takie rzeczy? Słuchajcie, dopamina to jest substancja chemiczna. Zadbajcie o to, żeby było jej najzwyczajniej w świecie dużo. Zadbajcie o zdrowy sen – i to jest temat, do którego na pewno jeszcze będę wracał w tym nagraniu, dlatego że to jest tak istotne dla Waszego dobrostanu. Dopamina się uzupełnia w trakcie snu.

Zadbajcie o ćwiczenia. Zadbajcie o jakąś formę automedytacji – do tego jest bardzo dużo aplikacji. W moim miniporadniku na temat snu jest na przykład wymieniona aplikacja Reveri, ale tak naprawdę można skorzystać sobie z Waking Upa, z Headspace'u itd. To pomoże Wam również dużo lepiej się skupić, gdy już usiądziecie do nauki. To też pomaga troszeczkę lepiej przenosić uwagę, co jest chyba kognitywnie najbardziej wymagającą rzeczą, którą można zrobić.

Zadbajcie sobie o dietę bogatą w L-tyrozynę, który jest niebezpośrednim prekursorem dopaminy. A jeżeli już w ogóle jesteście przekocurami, to zadbajcie o jakiś zimny prysznic. Ewentualnie możecie sobie poczytać o suplementacji L-tyrozyny, ale słuchajcie, skonsultujcie to z lekarzem. To jest megaśliski temat. Raczej nie chcecie brać żadnych tabletek tylko dlatego, że ktoś z podcastu o tym powiedział. Wiedzcie, że jest taka opcja, pogooglujcie sobie, jeżeli jesteście naprawdę tym zainteresowani. To jest, powiedzmy, taki pierwszy sposób.

Drugi sposób to jest właśnie wytworzenie sobie nawyku. Jeżeli uczysz się przez dwie minuty dziennie, bo nie masz czasu w ciągu dnia, to

chociaż odtwórz sobie na przykład kolejny film na playliście z Udemy – nawet przez chwilę, ale zrób ten nawyk, żeby on był codziennie. W ten sposób przyzwyczajasz swój układ nerwowy do tego sygnału. Jesteś z nim lepiej zaznajomiony. On w końcu da efekty, bo cały czas się starasz. Nie ma bata, będzie ten moment błędu przewidywania nagrody na plusik dla Ciebie i wtedy jeszcze lepiej się zmotywujesz. Będzie Ci po prostu z czasem łatwiej i to da efekty – to będzie takie kółko, które się toczy.

To ja może od razu powiem tutaj o swoim nawyku. Myślę, że będzie to dobry przykład. Jak coś nie będzie w nim pasować, to powiedz, to będę poprawiał. Mam tak, że wstaję rano, jak ogarnę córę, to siadam do komputera, robię wszystkie code review z dnia poprzedniego i dopiero wtedy jem śniadanie. Jeszcze przed śniadaniem robię rozgrzeweczkę, a potem dopiero śniadanko.

Dzięki temu, że mam nawyk siadania do tego code review i wiem, że potem będą ćwiczonka, będzie śniadanko, uzyskuję ten pozytywny efekt. Chce mi się więc wstać, chcę zrobić code review, bo wiem, że niedługo po tym będzie nagroda, na którą czekam. Czy myślisz, że tutaj można powiedzieć o nawyku, o którym Ty mówiłeś?

I tak, i nie z tego względu, że układ dopaminergiczny możesz modelować swoim przekonaniami. Jeżeli wiesz, że coś jest dla Ciebie dobre, to wtedy jest to dla Ciebie przyjemniejsze, ten układ się uruchamia. Ponadto układ dopaminergiczny się uruchamia w momencie, kiedy przyswajamy nową wiedzę, co też jest dla nas bardzo opłacalne z tego względu, że dzięki temu łatwiej się zmotywować do nauki. Chodzi o to, żeby troszeczkę wzmocnić ten sygnał w stosunku do innych rzeczy, które robisz. Tak w sumie bym to skomentował.

Dobra, czyli mamy wejście w nawyk po to, aby przyzwyczaić swój układ nerwowy do tego sygnału, aby to było takie samonapędzające się koło, które się toczy. Ewentualnie możecie sobie do nawyków dorzucić coś, co się nazywa nieprzewidzianą nagrodą. Przykładowo uczycie się przez dwa tygodnie bardzo pilnie, nigdzie nie byliście ze znajomymi, bo musieliście im odmówić, musieliście – tak jak ja musiałem kilka razy – przegapić jakiś weekendowy wyjazd z rodziną, dlatego że akurat klikałem jakiś projekt ze swojego mentoringu.

Macie z tego bardzo dużą satysfakcję, ale już czujecie się troszeczkę zmęczeni. Idźcie sobie gdzieś ze znajomymi, idźcie sobie do kina, zróbcie coś dla siebie. Dajcie sobie taką nagrodę, która jest nieregularna, ale nie do przewidzenia wcześniej przez wasz układ dopaminergiczny, dlatego że właśnie wtedy wychodzi ten błąd przewidzenia nagrody. Robicie coś, co jest mocno dopaminergiczne, czyli na przykład idziecie na drinka, jecie pół tabliczki czekolady, ale nie robicie tego często, robicie to w różnych okresach czasu w zależności od tego, jak chcecie. Nie bójcie się tego typu rzeczy, to naprawdę pomaga.

Mówiliśmy już o nawykach i o tym, żeby mieć dużo tej substancji chemicznej – to wynika z waszego lifestyle'u. Trzeci sposób (to może będzie najbardziej kontrowersyjne, ale też niezwykle istotne): unikajcie rzeczy, które Wam ten układ kompletnie rozregulują. I teraz wjeżdżamy na dosyć ciężkie tematy. Jeżeli jesteście uzależnieni od czegoś – od alkoholu, od narkotyków, od posiadania jednocześnie trzech dziewczyn, które nie wiedzą o sobie nawzajem – to są to zachowania, które bardzo mocno pobudzają Wasz układ dopaminergiczny. Wróćmy do huśtawki –

ten układ Wam z powrotem spadnie i wtedy naprawdę będziecie mieli bardzo duży problem.

Mówiąc o układzie dopaminergicznym, nie możemy nie powiedzieć o terapii uzależnień. Układ dopaminergiczny zarządza Waszym organizmem też w takim sensie, że przez jego złe działanie się uzależniacie. W biologii bardzo dobrą definicją uzależnienia jest: systematyczne zawężanie rzeczy, które sprawiają Wam przyjemność. Jeżeli będziecie na przykład uzależnieni od alkoholu, to będzie Wam trudniej czerpać przyjemność ze słabszego stymulantu, którym jest nabywanie nowej wiedzy w trakcie nauki programowania.

Nie mówię, żeby na przykład w ogóle nie pić alkoholu, bo życie to jest rachunek zysków i strat, musicie też robić rzeczy, które sprawiają Wam przyjemność – to też jest bardzo zdrowe. Musicie spotykać się z innymi ludźmi, a mamy takie społeczeństwo, jakie mamy. Ja też nigdy nie wycinam alkoholu do zera, więc na pewno nie będę Wam mówił, żebyście to zrobili. Naprawdę jednak unikajcie rzeczy, które bardzo mocno powodują górki dopaminergiczne.

Innym przykładem pompowania takiej górki dopaminergicznej jest: *będę się uczył programowania codziennie, ale za każdym razem po nauce napiję się zimnego piwka albo zjem tabliczkę czekolady*. Spoko, rzeczywiście będziesz miał tę górkę, zakoduje Ci się ten sygnał, że nauka była bardzo przyjemna, ponieważ wiązała się z wyższą nagrodą, ale w ten sposób budujesz olbrzymy na glinianych nogach. Ten olbrzym nie wytrzyma na tych glinianych nogach. One mu się załamią i wtedy spadnie Twój poziom zero – taki domyślny poziom dopaminy. Wówczas będzie Ci trudniej się zmotywować.

Takie rzeczy naprawdę potrafią powodować u ludzi ciężką depresję, więc podsumowując: nawyki, styl życia i unikanie uzależnień to byłyby takie moje, powiedzmy, top trzy. I na tym skończmy tę odpowiedź, bo i tak się trochę rozgadałem.

Jasne. Uf, kto powiedział, że przebranżowienie jest taką prostą sprawą.

Ale wiesz, to jest tak, że ja Wam daję jakieś podstawy z neurobiologii, które są megaelementarne. Wy możecie to wszystko sobie sprawdzić. Zauważcie, że ja nie mam wykształcenia medycznego, nawet nie jestem biologiem. Ja po prostu w czasie covidu zacząłem uczyć się programowania. Jak był covid, to ludzie nagle zyskali tyle czasu! Przestaliśmy jeździć do biur, siedzieliśmy wyłącznie z rodziną.

Nie zdajecie sobie sprawy, ile poza uczeniem się programowania i opieką nad niemowlakiem miałem jeszcze wolnego czasu, który wykorzystywałem na oglądanie sobie YouTube'a z wykładami, czytanie badań na Google Scholar, na PubMedzie, na czytanie książek popularnonaukowych, które opisywały te tematy. Z każdej rzeczy starałem się wyciągnąć coś, co pomagało mi uczyć się szybciej, bardziej efektywnie od innych, dlatego że wiedziałem, że jako ojciec i mąż jestem do tyłu w stosunku do ludzi, którzy nie mają takich zobowiązań.

Była to pierwsza z dwóch części rozmowy o przebranżowieniu na programistę. W następnym odcinku poruszymy m.in. temat sposobów nauki programowania, budowania portfolio i przygotowania się do poszukiwania pracy w IT.