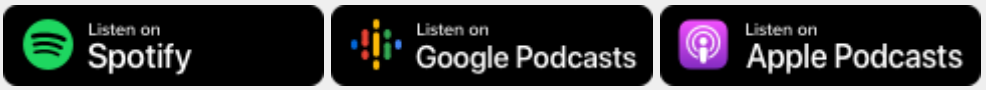
Pierwsze kroki w IT

PODCAST



Rekrutacja techniczna – przygotowanie się i przebieg

Gość: Radek Wojtysiak

Wszystkie polecane materiały i linki znajdziesz na stronie odcinka:

https://devmentor.pl/b/rekrutacja-techniczna-przygotowanie-sie-i-przebieg

Dziś moim gościem jest Radek Wojtysiak. Radek opowie nam o rekrutacji technicznej oraz o tym, jak się do niej przygotować.

Radku, dziękuję, że przyjąłeś moje zaproszenie na rozmowę!

To ja dziękuję, że mnie zaprosiłeś i że możemy porozmawiać.

Powiedz, Radku, coś więcej o sobie i co łączy Cię z branżą IT.

Moja historia jest już całkiem długa, bo liczy sobie parenaście lat. Gdy przygotowywałem się do naszego spotkania, robiłem taki rachunek sumienia, kim ja jestem i co tutaj w ogóle robię. Powiedziałbym, że jestem programistą trochę już mniej programującym, bo ukierunkowanym na ścieżkę menadżersko-liderską.

Od razu może powiem, gdzie obecnie pracuję, bo od całkiem niedawna (od 1 marca) jestem świeżo upieczonym menadżerem zespołów front-endowych w firmie The Software House i tam się realizuję.

Jeśli chodzi o to, co robię po godzinach: możecie mnie znaleźć przede wszystkim na LinkedInie. Tam staram się dzielić swoją wiedzą. Jest to dla mnie taki kluczowy projekt, produkt – prowadzę biuletyn „Kariera Developera". Możecie znaleźć tam bardzo dużo wartościowych treści związanych z tym, w jaki sposób można rozwijać się w branży IT, w tej branży bardzo dynamicznej, branży, które zaskakuje nas z dnia na dzień. Staram się przygotowywać takie materiały, żeby świadomie podchodzić do tego świata.

A oprócz tego lokalnie prowadzę razem z moją żoną miejsce rozwoju dzieci w kierunkach nauk ścisłych oraz robotyki. To miejsce nazywa się Przy klockach, w Kostrzynie Wielkopolskim. Mamy tam zajęcia chociażby z lego lub z Minecrafta (z programowania). Zachęcam, jeśli ktoś jest z Kostrzyna, to zapraszam, możemy się wtedy spotkać.

To jeszcze dopytam: jeśli chodzi o inicjatywę związaną z dziećmi, to w jakim wieku?

Jeśli chodzi o dzieciaki, to rozdzielamy. Na dzieciaki takie najmniejsze, czyli tak naprawdę od 0+. Tam są wtedy zajęcia bardzo zbliżone do sensoplastyki, do obcowania ze światem. Mamy zajęcia muzyczne, mamy zajęcia (to nowość zresztą) z językiem angielskim, gdzie pani ma dla tych dzieciaków np. jakieś wierszyki, piosenki, żeby one też mogły się rozwijać.

Jeśli chodzi o programowanie i nauki ścisłe, to mówimy tutaj o dzieciakach 5+, tak bym powiedział. Chociaż moja córka 4+ też całkiem daje sobie radę z tym wszystkim, więc raczej… Na naszej stronie jest 5+, ale jeżeli widzimy, że jakieś dziecko wie np. jak budować z tych klocków, wie, jak obsługiwać tablet, to nie skreślamy tego. Podchodzimy bardzo indywidualnie do każdego zainteresowanego.

OK, no dobra. Czyli jeśli jesteście z okolicy i chcecie nauczyć dzieci czegoś więcej, to zapraszamy.

A my wracamy do tematu tego odcinka, bo będziemy mówić o rekrutacji technicznej.

Co to właściwie jest rekrutacja techniczna? Czy to synonim live codingu, zadania do wykonania w domu czy może rozmowy bez siadania do komputera?

Rekrutacja techniczna to sposób weryfikacji naszych umiejętności technicznych. To jest przede wszystkim etap rekrutacji, etap całego procesu rekrutacyjnego. Nie jest to synonim, tak jak tutaj wymieniłeś, ponieważ to, co wymieniłeś, jak np. ten live coding czy zadanie do

wykonania w domu, to to jest opcja, którą możemy otrzymać w ramach tej rozmowy technicznej, tej weryfikacji.

Rekrutacje techniczne mają to do siebie, że niestety kojarzą nam się często z dużym stresem. Mają one charakter bardzo oceniający. Zawsze tak do tego podchodzę. Dlaczego? Ponieważ mamy rekrutera technicznego, który (można tak śmiało powiedzieć) trochę patrzy nam na ręce, patrzy na to, jak pracujemy, jaką jakość kodu reprezentujemy. To budzi stres.

Pomyślcie sobie, jak często w takiej zwykłej pracy – nawet niekomercyjnej – gdzie tworzycie sobie jakikolwiek projekt, ktoś patrzy, co robicie. Jest to więc pierwsza z trudności, ale w mojej ocenie rozmowa techniczna (już może tak o tym mówmy) jest istotna, ponieważ musimy chociaż trochę wiedzieć, kogo zatrudniamy.

Podsumowując: jest to bardzo ważny etap, niestety często bardzo stresujący. Istotny, bo pokazujemy, co potrafimy.

Ostatni z wymienionych przez Ciebie przykładów to rozmowa bez siadania do komputera. Ja bardzo lubię te rozmowy, ponieważ wymieniamy doświadczenia, wymieniamy to, co wiemy, i naszym odbiorcą jest ktoś, kto chce z nami pogadać. Zawsze to przyrównuję do takiego spotkania przy kawie. Najczęściej nie mamy kawy, ale jest to taka luźniejsza rozmowa. Opcji jest więc kilka.

To jeszcze bym dopytał, bo w zasadzie o tym nie wspomniałem. Co myślisz o testach? Mówiłeś o tym, że to rozmowa techniczna, ale spotkałem się też z tym, żę często pojawiają się po prostu testy. Ma to być niby alternatywa do rekrutacji technicznej (a w zasadzie

rozmowy), w związku z czym nie zawsze ta rozmowa się pojawia, bo rozmowa jest potem z osobą, która niekoniecznie jest techniczna.

Co sądzisz o takiej formie, jak sam stwierdziłeś, weryfikacji umiejętności?

Wiesz co, testy sprawdzające naszą wiedzę są fajne, ale wydaje mi się, że są one często niewystarczające, bo wiele firm ma przygotowany taki zestaw testowy i najczęściej jest tak, że on jest i nie zmienia się zbyt często.

Załóżmy, że nie dostaliśmy pracy, test nam nie poszedł, a po roku spróbujemy jeszcze raz. Jest niestety duże prawdopodobieństwo („niestety" w kontekście zatrudnienia kogoś), że pytania będą identyczne. Dlaczego? Bo koszt stworzenia nowego testu, jeżeli ten się sprawdza, jest całkiem wysoki. Musi nad tym usiąść co najmniej dwóch programistów na co najmniej godzinę i to przemyśleć. Może powinien sprawdzić to jeszcze jakiś rekruter. Żeby coś takiego przygotować, to jest kupa roboty.

Dlatego testy są fajne i ja podpisuję się pod nimi – żeby one były – ale niech to nie będzie jedyna weryfikacja. Ja np. często spotykałem się z czymś takim, że test miał miejsce, po czym była jeszcze rozmowa techniczna, czyli ten ostatni wariant, o którym wcześniej wspominaliśmy: żeby może jeszcze dopytać o nasz tok myślenia, o to, co robiliśmy, w jakich projektach pracowaliśmy. Wtedy jesteśmy w stanie uzyskać taki całkiem klarowny obraz, kim my jesteśmy jako programiści.

Na jakim etapie rekrutacji spodziewać się części technicznej?

Najczęściej jest to drugi etap rekrutacji. Mówię „najczęściej”, bo pewnie nie jest to reguła. Ale pierwszy etap to zawsze jest spotkanie z jakimś przedstawicielem działu HR. Tam najczęściej jest sprawdzane, czy nasz profil spełnia kryterium danej oferty, czy to, co robimy, co potrafimy, w czym się obracamy, wpisuje się w to, czego oni potrzebują.

Następnie, gdy już mamy taki „kontrakt” między nami programistami a zespołem HR, trzeba zweryfikować nasze umiejętności. Jest więc to najczęściej etap drugi, ale tak jak już wspomniałem wcześniej, może się zdarzyć, że będzie to część wieloetapowa składająca się z kilku mniejszych spotkań.

Wyobraźcie sobie, że są nawet firmy (nie będę tutaj podawał nazwy), gdzie takie spotkania są 5-6-etapowe, a nawet 7-etapowe. To już jest hardcore, ale firma ma w tym jakąś potrzebę. Nie nastawiałbym się więc, że zawsze jest to jedna część.

Patrząc z drugiej strony, rozumiem, że firma potrzebuje sprawdzić kandydata, ale jeżeli kandydat ma np. 4 takie oferty na stole i w sumie wychodzi, że ma 30 takich spotkań, to może być zmęczony i niekoniecznie mieć siłę na spotkanie któryś raz kolei, prawda?

Mhm. Dlatego trend pokazuje, że im mniej etapów, tym lepiej. Ostatnio rozmawiałem z moim znajomym (to chyba było nawet 3 dni temu), który mi pokazał, jak to wygląda u nich w firmie. Wyobraź sobie, że oni z 3 etapów skrócili rekrutację, jak on to powiedział, do 0,5 etapu.

Co to oznacza? Tak naprawdę już w pierwszej rozmowie HR zadaje kilka predefiniowanych pytań, Ty na nie odpowiadasz, oni to sobie spisują, mają już jakąś wiedzę na Twój temat, zadają parę pytań dodatkowych i

koniec. Mówi: ryzykują, ale teraz potrzeba programistów jest tak duża, a etapy są czasami tak długie, że trzeba zaryzykować. Jeśli ktoś się nie sprawdzi, to będziemy myśleli, jak tej osobie pomóc, albo – brutalnie mówiąc – jak tę osobę zmienić. Niestety tak też może być.

W zasadzie tak sobie teraz pomyślałem, że chyba szybciej jest wrzucić kogoś na projekt i od razu sprawdzić, jak to wygląda, niż tracić czas. Wydaje mi się, że rekrutacja – nawet jeżeli byłaby 7-etapowa – to i tak wszystkiego nie sprawdzi. I ile czasu straciłeś na samą rekrutację? Potem niestety myślisz sobie: *o, już tyle czasu straciliśmy, bo była taka długa rekrutacja, to co dalej?* A jeżeli jest to megaszybkie, to jeśli jest OK, to jest OK, a jeśli nie, to łatwiej się rozstać – i z perspektywy pracownika, i pracodawcy.

Dokładnie, to jest bardzo prosta kalkulacja. Jeżeli będziemy mieli jeden etap, to poświęcimy na niego godzinę dla może dwóch osób. A jak będziemy mieć 7 etapów, to trzeba to jeszcze przemnożyć przez liczbę osób rekrutujących. To jest zawsze pieniądz – jakkolwiek byśmy do tego nie podchodzili, ktoś musi wykonać pracę, a my musimy za tę pracę zapłacić.

My jako pracownicy też często tracimy czas na te rekrutacje, więc pamiętajmy o tym.

Dokładnie.

Co składa się na rekrutację techniczną? Czy ma jakieś stałe elementy?

To jest ciekawe pytanie. Zastanawiałem się nad tym, gdy dostałem pytania od Ciebie, i trudno mi było na to jednoznacznie odpowiedzieć, aczkolwiek wydaje mi się, że są powtarzalne formaty, o czym też już wspomniałeś. Mamy coś takiego jak live coding, są rozmowy techniczne, czyli takie *face to face*. Popularny jest też code review zadania, które wykonaliśmy w domu. Są też zadania logiczne, algorytmiczne, architektoniczne – na pewno więc, idąc na rozmowę techniczną, jeżeli o to nie zapytamy, nie mamy pewności, co dostaniemy.

Oczywiście warto zapytać przedstawiciela HR-ów – tę osobę pierwszego kontaktu – na co się przygotować. Ta osoba raczej powinna nam powiedzieć: *patrz, tutaj będziesz rozmawiał z tą i tą osobą i będziecie robić to i to*. Ale póki nasza ciekawość nas do tego nie skłoni, to nie nastawiałbym się, że będzie to coś konkretnego.

Wydaje mi się też, że trzeba być ostrożnym w kwestii przygotowywania się stricte na taką techniczną weryfikację. Bardzo dużo osób pyta bardziej doświadczonych programistów, w jaki sposób przygotować się na rozmowę rekrutacyjną techniczną. Ja zawsze odpowiadam, że się nie da. Dlaczego? Oczywiście możemy mieć jakieś książkowe pytania, bo oczywiście są książki, które dają nam gotowe pytania rekrutacyjne mające zweryfikować konkretną umiejętność, konkretny zakres wiedzy. Tylko że zadawanie takich pytań jest zupełnie bez sensu, bo co projekt, to inne zwyczaje; co projekt, to inne technologie, inne standardy. W mojej ocenie rozmowa techniczna powinna opierać się na tym, czy kandydat spełnia w jakimś stopniu standardy, które my reprezentujemy w naszej działalności, w naszej firmie.

Jak mamy na coś się przygotować, jeśli nie mamy zielonego pojęcia, jak dana firma pracuje? Rozmawiając z moimi kolegami i koleżankami w jakichś tam kuluarach, chociażby podczas różnych konferencji branżowych, często pytam o to, jak to u nich wygląda – bo ja też biorę udział w rekrutacjach jako rekruter techniczny i jestem ciekaw, jak to można poprowadzić. I chyba nie zdarzyło mi się, żeby pytania, które te osoby zadają, się pokrywały.

Te czasy chyba już trochę minęły, gdzie w przypadku chociażby języka JavaScript zawsze ktoś pytał o hosting. To było pytanie, na które wszyscy po kolei się przygotowywali z książką w ręku, „bo będą pytać o hosting". Teraz raczej się pyta, w jaki sposób my programiści podchodzimy do problemu, w jaki sposób byśmy rozwiązali ten problem z… no właśnie: z tą wiedzą, którą posiadamy.

Tutaj więc zdecydowanie należy być ostrożnym z tym przygotowaniem się, poświęceniem czasu. To, co powiedziałem: niektóre pytania mogą się powtórzyć, ale ich treść może być zupełnie inna. Wynik może bazować na podobnych metodach, na podobnych technologiach czy funkcjonalnościach danego języka programowania, ale rozwiązanie wyryte na blachę nie jest dobrym pomysłem.

To może bym podrzucił pewien pomysł. Nie wiem, na ile to by się sprawdzało, ale jeżeli byśmy wiedzieli, z kim mamy rekrutację, to można byłoby też zrobić research u tej osoby – może ta osoba prowadzi bloga, może ma kanał na YouTubie. Podejrzewam, że jeżeli ta osoba mówi o tego typu rzeczach, to pewnie jest duże prawdopodobieństwo, że też nas zapyta o tematy, na które sama się wypowiada. Może więc byłby to jakiś trop. Co myślisz?

Wydaje mi się, że można. Można zaryzykować. Tylko istotne jest też to, co powiedziałem wcześniej: często te rozmowy techniczne bazują na konkretnych projektach, które realizujemy w firmie, i sposób realizacji tego projektu jest jakby zbudowany ze światopoglądu wielu programistów (może tak bym to ładnie powiedział).

Pomysł jest ogólnie dobry, tylko może poszedłbym o krok dalej i sprawdził, czy dana firma nie ma bloga, gdzie artykuły piszą ich specjaliści. Tam mamy wtedy czarno na białym podaną informację, o jakich technologiach mówimy – bo nie będą pisali o technologiach, na których się nie znają, w których nie są ekspertami.

Odpowiedziałbym więc, że tak, warto. Zawsze warto robić research, ale dla mnie takim pierwszym miejscem wiedzy byłby blog firmowy, a później może faktycznie blog osoby weryfikującej. Ja zawsze zresztą sprawdzam osoby, które rekrutuję, ale wcześniej, gdy ja się gdzieś tam rekrutowałem, to pierwsze, co sprawdzałem, to kto będzie moim rekruterem technicznym. Bardzo łatwo to zrobić, wystarczy zajrzeć chociażby na LinkedIn. Tam praktycznie wszyscy mają swoje konta. Nie chcę mówić, że są aktywni, ale większość z tych osób te swoje konta gdzieś tam ma.

OK, no dobra, czyli mamy jakiś trop. To może kolejne pytanie.

Już trochę o tym wspomniałeś, ale może byśmy to podsumowali: czym różni się rekrutowanie przez osoby z firmy, do której aplikujemy, od rozmowy technicznej z rekruterem zewnętrznym?

Czy któraś z form rekrutacji jest według Ciebie korzystniejsza dla kandydata?

Wiesz co, wydaje mi się, że różnica jest całkiem spora, bo mając rekrutera technicznego z firmy, do której aplikujemy, mamy pewność, że ta osoba ma bardzo świeżą, rzetelną wiedzę. Taką, która faktycznie jest. A gdy myślę o rekruterze z firmy zewnętrznej, czyli gdy zlecamy to, żeby ktoś nam przyprowadził kandydata, no to pewnie: zakładam, że cały ten proces też jest skomplikowany i te osoby są przygotowane, ale to jest nadal przygotowanie w oparciu o szereg pytań. Czyli oni przygotowują się jakby z odpowiedziami narzuconymi z góry.

Dla mnie rekruter pochodzący z firmy, czyli ten wewnętrzny, to jest super sprawa, bo można mu zadać masę pytań związanych właśnie z technologią, z procesami, jakie występują w firmie. Można zapytać o to, z jakich narzędzi korzystają. Można nawet iść o krok dalej i zapytać o to, co dana osoba o czymś tam sądzi. Tutaj więc możliwości jest całkiem sporo.

No dobrze, to może powiedzmy o kolejnej rzeczy. O tym nie wspominaliśmy, a wydaje się, że może się coś takiego zdarzyć. Czy powinniśmy się przygotowywać do opowiedzenia o własnych projektach i omawiania ich fragmentów kodu? Czy na rozmowie technicznej tych tematów się nie porusza?

Zdecydowanie warto jest mieć przygotowane te fragmenty własnego kodu – na sto procent. Warto jest mieć wiedzę na temat tego, co robiliśmy wcześniej. To jest, jakby to powiedzieć, część, która powinna

wprowadzać tę osobę rekrutującą do zadania konkretnych pytań rekrutacyjnych lub chociaż skłaniać tę osobę do zastanowienia się, w jaki sposób tej całej naszej historii użyć, żeby zyskać wiedzę na temat tego, czego ta osoba rekrutująca potrzebuje.

Czyli po pierwsze: to, co my robimy dodatkowo, te wszystkie projekty to jest coś, co my wiemy, co my mamy, ale dobrze jest założyć, że na pewno nie jest to wystarczające – czyli że ktoś będzie zadawał pytania. Czyli tu jest też ryzyko, więc też uwaga: jeżeli ktoś tworzy projekt w oparciu o jakieś gotowe rozwiązania, gdzie szuka fragmentów kodu i je wykorzystuje, to dobrze jest wiedzieć, co te fragmenty robią. Bo jeżeli ktoś nas zapyta, co robi ta funkcja, a my przy tej osobie będziemy rozkminiać, co ta funkcja robi, to to jest w mojej ocenie bardzo źle odbierane.

Jeżeli więc się czymś chwalimy, miejmy pewność, że jest to nasza praca, a nie czyjaś. Oczywiście można używać fragmentów kodu, jakichś bibliotek zewnętrznych – no bo przecież wszyscy z tego korzystamy. Tylko róbmy to świadomie, bo rekruter będzie chciał wiedzieć, czy my wiemy, w jaki sposób się z tego korzysta, a nie wkleja.

Warto też tutaj powiedzieć, że te pytania to nie jest pewniak, więc nie starajmy się przygotowywać wielkich elaboratów na temat naszych projektów. Powiem uczciwie: mnie się zdarzyło raz albo dwa, że ktoś mnie zapytał o projekty dodatkowe. Raczej ludzie pytają się o to, w jaki sposób ja się rozwijam, gdzie szukam wiedzy, jak tę swoją wiedzę rozszerzam. Tak bym odpowiedział.

No dobrze, to w zasadzie mam pytanie, na które już trochę wcześniej odpowiedziałeś, ale spójrzmy na nie szerzej. Czyli nie tylko pod kątem technicznych tematów, ale może też mentalne.

Jak przygotować się do rozmowy technicznej, gdy aplikujemy do różnych firm, i jeszcze nie wiemy, czego się spodziewać? Co się najlepiej sprawdza? Może jakieś zadania na rozgrzewkę? Może mówienie na głos, żeby łatwiej sobie to wszystko uświadamiać? A może po prostu pair programming ze znajomym, żeby przyzwyczaić się do tego, że zaraz ktoś będzie nam na te ręce patrzył, do czego niekoniecznie musimy być przyzwyczajeni?

Moim zdaniem to jest kwestia bardzo indywidualna. Dlaczego? Bo co człowiek, to są różne potrzeby. Mówię tutaj o potrzebach stricte związanych z naszym dobrostanem: żebyśmy my dobrze się czuli podczas rozmowy technicznej.

Jeżeli miałbym mówić o sobie, to ja zdecydowanie na rozmowy techniczne się nie przygotowuję. Wcale nie chcę, żeby to zabrzmiało tak, że OK, mam doświadczenie, to nie potrzebuję się przygotowywać. Nie, to zupełnie nie ma nic wspólnego. Ja po prostu wychodzę z założenia, o czym już mówiłem, że jest to zbyt dynamiczny świat. Branża IT zmienia się z dnia na dzień. Jeżeli pomyślicie sobie o bibliotekach albo frameworkach, które wykorzystujecie w Waszych projektach, to pewnie jakiś procent z nich każdego dnia może mieć nową wersję (czy to wersję patch, czy minor, czy major, ale coś się tam zmienia). I teraz jeżeli ktoś będzie chciał mnie zrekrutować i zapytać o rzeczy bardzo szczegółowe, to i tak nie będę miał tej wiedzy – bo nie jestem w stanie jako człowiek,

pracując powiedzmy 8 godzin dziennie zawodowo, wszystkich tych nowości skonsumować.

Dlatego ja wychodzę z założenia, że przed rozmową rekrutacyjną lepiej przygotować się mentalnie. Tak się składa, że na swoim profilu pisałem artykuł na ten temat, więc jeżeli ktoś będzie chciał bardziej się zapoznać z moją myślą, to zapraszam też tam.

Co ważne: rekrutacja techniczna to bardzo często stres, gdzie rzeczy trywialne dla nas stają się wcale nie trywialne. Z własnego doświadczenia: miałem też taką sytuację, że będąc już tym wieloletnim programistą, zapytany o konstruktor (czyli o bardzo podstawową rzecz), nie potrafiłem opowiedzieć o definicji, bo byłem zestresowany.

Bardzo ważne jest przygotować się mentalnie, nie siedzieć do późnej godziny przy książce, weryfikując swoją wiedzę: czy na pewno wszystko wiem, czy na pewno mi się uda itd. Nawet jeżeli się nie uda, to nic wielkiego nie stanie. I tak wtedy bardzo dużo zyskuję, bo po pierwsze poznam pytania z tej firmy, po drugie będę miał doświadczenie z rekrutacją. To też jest już bardzo duży plus. Przede wszystkim jednak zadbam o siebie.

Pamiętajcie, że strefa Waszego komfortu też jest istotna z punktu widzenia osoby rekrutującej. Bo rekruterzy to też są ludzie i oni wiedzą, że my, kandydaci, się stresujemy. Miałem też taką sytuację, że ktoś wprost mi powiedział, czy weryfikacją może nie być live coding, bo osoba się stresuje i nic z tego dobrego nie wyjdzie, a ona ma pewność, że tę wiedzę posiada. Nie ma problemu. Prośmy, jeżeli czujemy dyskomfort. Reasumując: technicznie można próbować się przygotowywać. Też już to tutaj powiedziałem.

Są książki, które mają setki, jak nie tysiące pytań rekrutacyjnych w oparciu o setki, jak nie tysiące rozmów rekrutacyjnych z mnóstwa firm. Jednak podejdźcie do tego w taki sposób: czy macie pewność, że traficie akurat na te konkretne pytanie, których gdzieś tam się uczyliście? Jeżeli byście podeszli do tego tak, że poświęciliście 4 godziny albo nawet więcej na przygotowywanie się i nic z tego nie będzie, żadne z tych pytań nie będzie Wam zadane, to jest to trochę demotywujące, chociażby podczas takiej porażki, gdy nie uda nam się tej pracy dostać.

A tak, przecież Wy wiecie dokładnie, czego dana firma oczekuje. W ofercie, w odpowiedzi na którą składaliście CV, zapewne jest wyszczególnione, jakich technologii dana firma używa i czego oczekuje w kontekście zaangażowania danego pracownika. Zatem już po tym możecie stwierdzić, czy to jest coś, co jesteście w stanie robić.

Jeżeli więc macie wiedzę na temat, powiedzmy, języka programowania X, no to niech Was zweryfikują, czy faktycznie ta wiedza jest wystarczająca. Bo jeżeli brakuje Wam sporo, to nie nadrobicie tego w jeden wieczór. To podejście, zadbanie o swój własny dobrostan jest tutaj kluczowe, żeby ta rekrutacja przede wszystkim przeszła w zgodzie z tym, co my potrafimy, a nie, że stres zabierze nam wiedzę, którą posiadamy.

Czy masz jeszcze jakieś może porady dla osób, które mają odbyć taką rozmowę techniczną zdalnie?

Wydaje mi się, że rozmowa techniczna zdalna to jest plus dla osoby, która jest rekrutowana, ponieważ przede wszystkim siedzimy we

własnym domu, we własnym środowisku. Ktoś mi też powiedział, że to jest idealna przestrzeń, żeby przygotować sobie ściągi na ścianie, gdzieś tam na biurku, na laptopie, bo nikt tego nie zweryfikuje. No pewnie, można próbować i gwarantuję Wam, że rekruter Wam nie powie: *a teraz, proszę, przechodzimy przez pokój, w którym się znajdujesz, żeby sprawdzić, czy nie masz tutaj przygotowanych jakichś ściąg*. Z mojej perspektywy nie ma różnicy. To, co jest istotne: to jest miejsce, które zwiększy nasz komfort, zredukuje stres, o którym mówiłem przed chwilą.

Zaleciłbym też przede wszystkim nie podchodzić do tego jak do rozmowy z kolegą, bo to jest bardzo zły nawyk i będziemy zbyt lekceważąco podchodzić do całej rozmowy. Nie wyniknie z tego nic dobrego z punktu widzenia chociażby właśnie zespołu HR-owego.

Warto jest mieć takie nastawienie, jakbyśmy mieli się spotkać z człowiekiem na żywo. Czyli włączona kamera to podstawa. Nie wyobrażam sobie, żeby tak nie było. Tak więc wydaje mi się, że różnica między jednym a drugim wcale nie jest taka duża.

No dobrze, to przejdźmy do rozmowy technicznej, o której już tyle powiedzieliśmy. Jakich pytań możemy się na niej spodziewać?

Tutaj znowu trudno odpowiedzieć na to pytanie, jednak to też już padło. Są firmy, gdzie stawia się na takie dobrze znane pytania. Jest tych firm całkiem sporo. Dlaczego o tym mówię? Bo to też widać po postach chociażby na LinkedInie lub po ofertach pracy. Jak byście przejrzeli naście ofert często możecie zauważyć, że są tam podobne punkty

„czego oczekujemy”. Naprawdę bardzo podobne. Aż jestem czasami w szoku, czy oni od siebie nie kopiują, ale to już sugeruje, że idziemy w bardzo podobnym kierunku, jeśli chodzi o te oczekiwania.

Na pewno możemy spodziewać się pytań związanych z tym obszarem, na który aplikujemy – czyli jeżeli jesteśmy front-endowcami, a aplikujemy na back end, to nie oczekujmy tutaj front-endowych pytań, bo nie taki jest temat naszego spotkania.

Na pewno pytania związane z językiem programowania, który jest też pewnie wyszczególniony w ofercie. O to chodzi, żeby zweryfikować umiejętności w danej technologii, a nie innej. Zapewniam Was, że pojawią się też pytania ogólnoprogramistyczne. Dlaczego o tym mówię? Bo często jeżeli mamy na przykład dany framework z danego języka programowania… Zostańmy przy tym front endzie, powiedzmy, że mamy Reacta. Dużo osób uważa, że jak pójdzie na rozmowę techniczną, to będą pytania tylko z Reacta? No to niestety jest błąd. Oczywiście pytania z frameworka się pojawią, ale na bank pojawiają się też pytania z fundamentów języka programowania, czyli z JavaScriptu. Dlaczego? Bo jak korzystać z frameworka, jeżeli nie mamy pewności, że potrafimy korzystać z języka programowania? To by było bardzo dziwne. Dlatego zawsze ten fundament musi być znany. I wydaje mi się, że w 99% takie pytania o fundamenty się pojawiają.

Oczywiście jest też ryzyko, że jest bardziej ambitnie, czyli pojawią się pytania algorytmiczne, architektoniczne, sprawdzające sposób myślenia, sprawdzające sposób, w jaki radzimy sobie w stresie, jeżeli mamy np. mało czasu i musimy rozwiązać jakiś konkretny problem. Takich pytań też możemy się spodziewać. One są rzadziej, tak z mojej obserwacji.

Ale jeżeli firma poszukuje kogoś bardziej doświadczonego, to oczekiwałbym, że takie pytania mogą się pojawić.

Taką perełką z mojej perspektywy są pytania o to, czy dany kandydat testuje swój kod. To jest coś, co pojawia się bardzo często. Dlaczego? Ponieważ wcale nie jest standardem, że kod, który piszemy, jest pokryty testami. Kiedyś nawet miałem taką ankietę na LinkedInie, gdzie zapytałem o to wprost: czy piszesz testy jednostkowe. I byłem w szoku. Oczywiście dokładnych wartości nie pamiętam, ale mniej więcej wypadało to tak, że 60% osób testuje, a 40% osób nie. Mówię: te wartości mogą się nieco różnić, ale tak by się to rozkładało.

Często więc jest pytanie, czy wiesz coś na temat testów jednostkowych, testów end2endowych. Jeżeli wiesz, to co możesz nam więcej na ten temat powiedzieć: jak testować? Udowodnienie tego, że potrafi się testować, to w mojej ocenie jest duży plus. Ja jako ten rekruter techniczny, który potrzebuje osoby do tworzenia wysokiej jakości kodu, muszę mieć pewność, że ta osoba wie, w jaki sposób się zabezpieczyć przed potencjalnymi anomaliami. Dlatego pytania o testy ja osobiście zadaję zawsze. Podejrzewam, że wielu rekruterów również o to pyta.

Wydaje mi się, że to jest taki element, który pozwala wyróżnić się właśnie na tle innych kandydatów. To jest, powiedziałbym, dodatkowe pytanie, na które dobrze jest odpowiedzieć.

Dokładnie. Ja jeszcze tylko dopowiem, że to nie jest pytanie *must have*, w sensie – żebyście mnie też źle nie zrozumieli – jeżeli nie odpowiecie na pytanie o testy twierdząco, to wcale nie oznacza, że nie macie szans

dostać tej pracy. Jednak to jest dobra praktyka. My programiści powinniśmy testować i do tego też tutaj teraz nawołuję.

Tak jest. Testujemy kod, żeby się potem nie okazywało, że coś nie działa i musimy w sobotę czy w niedzielę pracować, żeby naprawić całą sytuację.

To przejdźmy może do kolejnych pytań. Czy pytania od kandydata w ogóle powinny się pojawić? Czy rekruter spodziewa się jakichś pytań? Jak to wygląda z Twojej perspektywy?

Wydaje mi się, że rekruter nie spodziewa się pytań. Raczej bierze pod uwagę, że takie pytania mogą się pojawić. Powiedziałbym więc, że na pewno jest przygotowany, że jeżeli pytanie padnie, to musi na nie odpowiedzieć.

Zadawanie pytań zdecydowanie dobrze świadczy o kandydacie, bo pokazuje, że kandydat jest zainteresowany daną firmą, jest zainteresowany tym, co ta firma robi i chce zweryfikować, czy jest to miejsce właśnie dla tego kandydata.

Jeśli chodzi o takie pytania, to kandydat przede wszystkim pyta o to, w jaki sposób pracuje taki zespół, do którego miałby trafić, oraz czy są jakieś procesy w firmie, z którymi powinien się zapoznać zanim zdecyduje się o zaakceptowaniu danej oferty.

Takie niuanse technologiczne to jest jedno, ale też są niuanse nietechnologiczne, np. jakie są benefity. To jest podstawa: czy kawa jest itd. Możemy się z tego śmiać, ale takie pytania po prostu są zadawane i

to są najnormalniejsze pytania. Ja na przykład nie widzę w tym nic złego.

Z mojej perspektywy, gdy ktoś mówi, że nie mam żadnych pytań, nawet takich najprostszych, to nie sugeruję, że ktoś bardzo, bardzo chce do nas dołączyć, tylko że może *jak mi się uda, to wtedy się będę zastanawiać, co tutaj robię*.

Ja sam jako kandydat (patrzę teraz przez pryzmat swojej przeszłości) nie zadaję dużo pytań. Wydaje mi się, że wiele pytań można zadać na różnych etapach rekrutacji. Osoba, która nas rekrutuje, zada to pytanie: czy my mamy pytania, ale w kolejnych etapach to samo pytanie będzie nam zadane, więc dawkowałbym te pytania. Jeżeli mamy rozmowę techniczną, to pytajmy o rzeczy techniczne. Jeżeli mamy rozmowę HR-ową, to pytajmy o rzeczy HR-owe. Jeżeli mamy rozmowę HR Leader, czyli taki kolejny etap, gdzie rozmawiamy z naszym potencjalnym przełożonym, to pytajmy, jak działa zespół, jak działają rzeczy związane z procesami w firmie.

To może dodałbym od siebie jeden tip. Wydaje mi się, że sposób zadania pytania jest bardzo istotny. Jeśli zapytamy: *jakie projekty robicie?* OK, ale to nie pokazuje, że jesteśmy zainteresowani. A jeżeli byśmy zapytali: *widziałem waszą ostatnią apkę, widziałem to rozwiązanie, starałem się coś takiego zrobić, ale miałem taki problem. Jak wy sobie z tym poradziliście?* To już wtedy całkiem inaczej wygląda, prawda?

Oczywiście. Jeśli po pytaniu „jakie projekty robicie?" ktoś nam poda chociażby nazwę projektu, to nie powie nam nic. Zdecydowanie więc podpisuję się pod tym, co powiedziałeś.

No dobrze, to lećmy dalej, bo mamy tutaj jedno z ciekawszych pytań. Może wydaje się proste, ale myślę, że jest ciekawe. Czy podczas rekrutacji technicznej można korzystać z Google'a?

Odpowiem tak trochę przewrotnie: ja tego nie wiem, ale mogę Wam podpowiedzieć, że na pewno wie to osoba, która Was rekrutuje. I tutaj pierwsze moje nawołanie jest do tego, żeby pytać. Nie kombinujcie sami. W sensie: nie interpretujcie tego w sposób „mogę / nie mogę".

Ja np. miałem taką sytuację (już lata temu): byłem na rozmowie rekrutacyjnej, dostałem zadanie front-endowe i zacząłem je robić. Nie zrobiłem go do końca, bo zapomniałem, jak nazywała się jakaś reguła w CSS-ie, po prostu nie pamiętałem tego. Nie korzystałem z internetu, na co przychodzi rekruter techniczny i pyta, jak mi poszło. Ja mówię, że tego zapomniałem, ale jak bym to sobie wygooglował, to pewnie bym wiedział. No i oni byli w ciężkim szoku: jak ja nie korzystałem z Google'a? Ja to wszystko pamiętałem? No akurat te konkretne zagadnienia pamiętałem.

Dlatego pytajcie, czy możecie. To nie jest nic złego, żeby zapytać, czy jeżeli wpiszecie sobie coś do Stack Overflow, to czy to dobrze. Biorąc pod uwagę, jak wygląda praca programistów, którzy w większości, jeżeli czegoś nie wiedzą, to googlują, to to jest w mojej ocenie bardzo naturalne podejście, że będziemy weryfikować naszą wiedzę. Przecież nie wiemy wszystkiego i zapewne wszystkiego nie będziemy wiedzieli przez ten czas, ten dynamizm (co już też tutaj padło).

Wydaje mi się jednak, że warto sobie uzmysłowić, że ta praca samodzielna wcale nie musi też być jakoś superekstra punktowana. W sensie: to, że my wiemy, jak zrobić coś zgodnie z danym schematem, wcale nie musi oznaczać, że robimy to dobrze. Nawet powiedziałbym więcej: jeżeli sobie coś googlujemy i szukamy rozwiązania, to nagle może się okazać, że to, co gdzieś tam wiemy, wcale już nie jest aktualne i znajdujemy jakieś nowe, fajniejsze rozwiązanie, bardziej optymalne lub sprawiające, że dana funkcjonalność będzie działała po prostu lepiej.

Zobaczcie: jeżeli nie korzystalibyśmy z tego Google'a, to wcale byśmy tej najlepszej wartości nie przekazali. Zapytajcie więc, czy możecie, a jeżeli możecie, to śmiało z tego korzystajcie, bo po to to jest i to, że potrzebujecie pomocy do rozwiązywania problemów, wcale nie będzie odbierane jako minus. Wszyscy programiści tak działamy – po prostu googlujemy.

Może to nawet będzie plus, bo szybciej znajdziemy dane rozwiązanie. Tak jak mówisz: optymalnie i lepsze z perspektywy biznesu.

Też może się tak zdarzyć, oczywiście. Ja więc nie widzę, żeby w tym było cokolwiek złego.

Jasne, to idźmy głębiej w ten temat. Czy Twoim zdaniem podczas takiej rekrutacji może nas spotkać coś podchwytliwego? Czyli np. mamy kod zadania, który nie działa albo nie działa tak jak trzeba, o czym niekoniecznie jest wspomniane w treści. Co rekruter może mieć wówczas na celu?

To, o co pytasz, to zapewne jest trochę takie code review. Ja bym to przyrównał właśnie do takiego code review, kiedy dostajemy fragment kodu, musimy się z nim zapoznać i nagle rekruter mówi nam: *uruchom to*, a to nie działa.

Wiesz co, wydaje mi się, że to jest bardzo fajne, żeby sprawdzić, czy ktoś po prostu po pierwsze umie czytać kod, a po drugie zna elementy danego języka programowania. Zacznijmy zatem od tego, że zdecydowanie może nas spotkać coś podchwytliwego, bo my mamy na takiej rozmowie myśleć, nie używać schematów, tylko myśleć, zastanawiać się, weryfikować i analizować.

Dlatego często spotyka się to, że dostajemy jakieś zagadnienie, kod nie działa, a my teraz, korzystając z naszej wiedzy i nie zmieniając logiki biznesowej, musimy wprowadzić jakieś drobniutkie usprawnienia. Mam na to nawet przykład: trzy lata temu miałem rozmowę rekrutacyjną. Znowu w języku JavaScript. Gdy mamy pętlę i w tej pętli deklarujemy zmienną, to możemy użyć albo albo *vat*, albo *let*, albo *const*. No i problem był taki, że było użyte złe słówko deklaracji. Powinno być inne (spłycając tę odpowiedź). Wystarczyło tak naprawdę zmienić trzy literki i już wszystko działało, i już oni wiedzieli, że my wiemy, w jaki sposób działa zakres zmiennych.

Podchwytliwe rzeczy mogą zatem mieć miejsce. Nie muszą. Warto być na to przygotowanym. Przede wszystkim (tego jeszcze nie powiedziałem) to jest też idealny sposób na sprawdzenie tego, czy my potrafimy pracować z czyimś kodem. Gdy się nad tym zastanowimy, to w takich projektach komercyjnych rzadko kiedy jesteśmy sami i sami tworzymy ten kod. Pewnie jest więcej osób, które tam działają. To jest

kluczowe, żebyśmy wiedzieli, co dana osoba napisała, gdzie zrobiła błąd, czy potrafimy wykonywać tę recenzję kodu (czyli to szeroko znane code review), bo będziemy mieli z tym do czynienia podczas pracy.

To zapytam Cię jeszcze o opinię. Mój podopieczny miał taką historię, że dostał zadanie rekrutacyjne. Powiedzmy, że było ono mocno rozbudowane, a na zrobienie go była godzina czy dwie. Było niemożliwe, żeby to wszystko wykonać. Potem okazało się, że to było zrobione specjalnie, żeby zobaczyć, czy ktoś będzie próbował i wykorzysta ten czas maksymalnie, czy po prostu od razu się podda. Chodziło o sprawdzenie, czy ktoś przynajmniej na początku będzie się starał cokolwiek zrobić, czy od razu odpuści. Co myślisz o takim zabiegu?

To jest bardzo ciekawy przykład i powiem szczerze: pierwszy raz słyszę o czymś takim. Ale jak tak teraz sobie myślę, tak w tym momencie, to ja analizowałbym to w dwóch kierunkach. Po pierwsze kandydat stwierdził, że tego kodu nie da się ukończyć w takim czasie i podał ku temu argument. Dla mnie świetna sprawa. Zauważył, zaargumentował, powiedział, że jest na to potrzebne więcej czasu. Oceniłbym to pozytywnie.

Z drugiej strony jest to, co powiedziałeś, czyli kandydat będzie próbował, kombinował, szukał, weryfikował itd. Tutaj też bym to ocenił pozytywnie. Zastanawiam się, czy jest na to jakaś negatywna odpowiedź. Może tylko taka, że nie da się i koniec.

W sytuacji, kiedy kandydat próbowałby rozwiązać to zadanie (zakładam, że komentowałaby to jakoś tak na żywo, żeby rekruter też wiedział, jakie myśli ma w głowie), to to też jest fajne. W sumie już o tym powiedziałem,

więc może nie będę tego powtarzał: ta osoba pokazuje, że ma rozum, taki mózg programistyczny – może to jest bardziej poprawnie powiedziane – i szuka tego rozwiązania. Szuka rozwiązania problemu, który jest zadany, a że się nie uda w danym czasie, no to ja osobiście pewnie bym powiedział: *jak bym posiedział nad tym dłużej, to pewnie by mi się udało*. Bardzo fajny przykład. Nie spotkałem się jeszcze z czymś takim.

Chcesz powiedzieć, że będziesz to wdrażał w swój wachlarz usług rekrutera, że tak powiem?

Musiałbym się głębiej nad tym zastanowić, powiem szczerze. No ale może, może to jest jakiś pomysł.

Musisz powiedzieć: *nie mogę powiedzieć*, bo zaraz wszyscy będą wiedzieć, jak to u Ciebie wygląda.

Było już pytanie o rzeczy podchwytliwe. Nigdy nie wiecie, co się wydarzy.

Tak jest, tak jest. No dobrze, to idźmy dalej. Ten mój przykład raczej pokazał, że same pozytywne rzeczy mogłyby z tego wyniknąć.

To ja chcę zapytać Ciebie o błędy najczęściej popełniane podczas rekrutacji technicznej. Jakie są i jak można ich uniknąć z Twojej perspektywy?

Gdy sobie o tym myślę, to mam takie dwa typowe błędy, które spotykam. Pierwszy z nich to jest nadinterpretacja lub niezrozumienie zadania. I to jest… aż jestem w szoku, że to powiem, ale to jest nagminne. Dlaczego?

Są ku temu dwa powody. Po pierwsze powód taki trochę dla mnie niezrozumiały, czyli mamy treść zadania po polsku, osoba sobie to czyta, ma jakieś tam dane wejściowe i jest napisane, jakie mają być dane wejściowe. Ta osoba robi to zadanie, robi, robi i wydaje się tej osobie, że wie co robi, ale to jakby się nie skleja.

Później jest zadawane pytanie pomocnicze, gdzie staramy się nakierować daną osobę na jakieś konkretne rozwiązania, a wtedy zdarza się, że ta osoba zadaje pytanie: *ale dlaczego w tym kierunku?* No i to już jest informacja, że chyba nie do końca się rozumiemy.

Drugi z kolei przykład jest wtedy, kiedy treść jest po angielsku. Znowu mamy wejście i wyjście. Ta osoba robi dokładnie to samo, czyli rozwiązuje zadanie, ale nagle zauważamy, że robi coś zupełnie innego. I niestety, co też jest dla mnie często zastanawiające, okazuje się, że kandydat nie zrozumiał technicznego angielskiego. Po prostu. Dlaczego to jest dla mnie takie zastanawiające? Ponieważ językiem programowania, takim językiem semantycznym, jest język angielski, gdzie dokumentacje i inne tego typu rzeczy dzieją się po angielsku. Oczekiwałbym więc, że takie podstawowe frazy, nawet techniczne, są znane. Niestety tak jest, ja to zauważam.

Oczywiście nie chcę też mówić, że to jakoś negatywnie rzutuje na rozmowę, bo jeżeli ktoś czegoś nie zrozumiał i to wyjdzie z dialogu między nami, to ja tej osobie podpowiem, że chodziło mi tutaj o to i o to. I nagle okazuje się, że ta osoba wie, co zrobić. Nie zawsze, ale często tak jest. Co tutaj można zrobić? Po prostu upewnić się, rozmawiać z rekruterem, zadać pytanie wprost: *czy chodzi Ci o to i o to? Bo nie*

jestem pewien. Odpowiem: tak/nie, nie musi tutaj nawet paść żadna podpowiedź.

Drugi z kolei błąd, który napotykam, to jest coś, co z angielskiego nazywa się *over engineering*, czyli mówiąc w skrócie: osoba stara się być tak perfekcyjna, że najprostsze zadanie pisze w wielu linijkach kodu, opisuje z komentarzami i z jakimś tam wydzielaniem do funkcji, do innych rzeczy. A okazuje się, że rozwiązanie można zamknąć w trzech linijkach. Niestety nie ma na to złotej rady, ponieważ dla mnie to jest zrozumiałe. Kandydaci chcą wypaść jak najlepiej, pokazać się z jak najlepszej strony: potrafimy korzystać z takiej metody, a tutaj potrafimy zrobić to, a może jeszcze dodamy klasę. Jest to zrozumiałe w przypadku rozmowy technicznej.

Zachęciłbym jednak do tego, żeby nie przesadzać. Przecież to, o co chodzi, to jest rozwiązanie danego problemu. Im prościej podejdziemy do tematu, czyli zaproponujemy najmniejszym kosztem naszej pracy rozwiązanie, które działa, tym lepiej. Oczywiście to nie musi być najbardziej optymalne rozwiązanie, ale później jest tzw. refactor, czyli my później ulepszamy nasze rozwiązanie. Wyszedłbym od tego, żeby nie budować tutaj nie wiadomo jakiej machiny do pytań rekrutacyjnych.

Warto zapamiętać, że pytania rekrutacyjne mają raczej formę prostą. Dlaczego? Bo nie ma dużo czasu, żeby zastanawiać się nad bardzo złożonymi algorytmami. Jeżeli dostaniecie zadanie do domu, wtedy możecie myśleć, możecie kombinować, możecie proponować rozwiązania, które mają w jakiś sposób pokazać, co potraficie. Na takiej rozmowie, która trwa godzinę, uważałbym, żeby starać się redukować czas poświęcony na rozwiązanie.

A czy jest coś, czego rekruter techniczny nie powinien robić? Czy jest coś, co może nam zapalić czerwoną lampkę i pokazać, że rekrutacja w tej firmie to może nie najlepszy pomysł?

Wydaje mi się, że taką naturalną rzeczą, która każdemu z nas pewnie przyjdzie na myśl, to jest taka ocena naszej pracy w czasie rzeczywistym. Załóżmy, że dostajemy zadanie do rozwiązania. Rozwiązujemy je sobie, mamy jakieś myśli w głowie, może piszemy sobie coś na brudno, coś co jeszcze w ogóle nie jest produkcyjne, a słyszymy jakieś komentarze typu wydaje *mi się, że to kiepski pomysł* albo *tak to się robiło pięć lat temu*.

Moim zdaniem to jest bardzo, bardzo negatywna postawa rekrutera i po pierwsze to może wpłynąć na naszą wydajność jako kandydata, może nas w ogóle zniechęcić do realizacji pomysłu, który mamy w głowie. Mówiłem już o tym: podczas takich rozmów jest dużo stresu. Jeżeli ktoś będzie kwestionował nasz pomysł, który w ogóle nie jest jeszcze zrealizowany, no to ten stres się tylko nasili.

Idąc dalej, są niestety podobne zabiegi. Ja na szczęście nigdy się z nimi nie spotkałem, ale kiedyś czytałem taki artykuł, gdzie był bardzo podobny temat. Ktoś opisywał rozmowy w jakiejś firmie i to, jak się czuł. Chodziło o te rzeczy, o których teraz mówię. Była jeszcze taka sytuacja, przynajmniej tam opisana, gdzie kandydat został porównany z innymi kandydatami. Czyli co to oznacza? Kandydat robił rozwiązanie danego problemu, a usłyszał: *hm, osoba przed Tobą wymyśliła to nieco lepiej*. To jest dokładnie ten sam case, dokładnie ta sama sytuacja, w której ja jako kandydat zamykam się w sobie i nie chcę już dalej brać w tym udziału.

Mam olbrzymi stres. Ja po prostu czuję się źle, nawet jak pomyślę sobie o takiej sytuacji – że ktoś mógłby powiedzieć mi, że inni nie mają z tym problemu, a ja mam – to jest wręcz bardzo dobijające.

Ostatnią, taką już trochę lżejszą rzeczą, która mi przychodzi na myśl, jest zadawanie zbyt skomplikowanych i zawiłych pytań albo zadań. Pytania i zadania są tworzone przez człowieka, który w różny sposób może dany problem przedstawić. Załóżmy, że chcemy sprawdzić, czy A jest równe B. Najlepiej zadać pytanie *czy A jest równe B?*, a nie *czy jeżeli coś tam coštam coštam coštam coštam coštam, to A jest równe B?*, a chodziło dokładnie o to samo, że mieliśmy np. 1 równe 1.

Na pewno więc zachęcałbym wszystkich rekruterów (jeżeli rekrutujecie, to i Was też), żeby upraszczać pytania. Dbajcie o to, żeby kandydaci wiedzieli, co mają zrobić, i żeby te odpowiedzi, które mają Wam oddać, wynikały z ich wiedzy w oparciu o bardzo łatwe zadane pytania. One powinny być klarowne, niezawiłe, jasne i transparentne dla każdego, tak naprawdę na równym poziomie. Każdy kandydat powinien zrozumieć to tak samo.

Myślę, że to warto dodać: pytania, o których wspominasz, mogą wyglądać w ten sposób, że każdemu kandydatowi można zadać to samo pytanie, ale na każdym stanowisku (poziomie) możemy oczekiwać innej odpowiedzi. To też pokazuje, w jaki sposób może wyglądać rekrutacja – jeżeli pytamy kogoś, no niech będzie: o ten hoisting czy coś innego, to w przypadku juniora oczekujemy pewnie dużo prostszej odpowiedzi niż od seniora, który powinien dogłębnie opowiedzieć, jak to działa, dlaczego istnieje i po co w

ogóle warto to robić. To, że pytanie jest proste, nie oznacza, że musi być na bardzo podstawowym poziomie. Zgodzisz się ze mną?

Zdecydowanie. Ja wychodzę z założenia, że dana oferta, na którą aplikujemy, ma dokładnie określone, do kogo jest kierowana. Jeżeli myślimy tu o młodszych programistach, to ja sobie nie wyobrażam, że będziemy oczekiwać od nich złożonych definicji. Więc te pytania, które zadajemy, mogą być właśnie, jak powiedziałeś, dokładnie takie same (o definicje), ale możemy oczekiwać różnych odpowiedzi od różnego poziomu. Nadal więc te pytania powinny być jak najprościej zadane. A jeżeli potrzebujemy, żeby to pytanie było głębsze, żeby zweryfikować wyższy poziom programisty, to starajmy się to po prostu dopowiedzieć. Jednak ten *core*, ten główny fragment pytania powinien być na dokładnie takim samym poziomie.

No dobrze, to ja bym może jeszcze wrócił do tego stresu, o którym wspominałeś na początku poprzedniej odpowiedzi, bo to jest trudny temat, to naprawdę ogranicza nasze możliwości wypowiedzenia się, poznawcze itd. Jak więc poradzić sobie ze stresem na rozmowie technicznej? Czego warto mieć świadomość, żeby być spokojniejszym?

Ja wyjdę tutaj z takim apelem do wszystkich: pamiętajcie, że rozmowa rekrutacyjna to jest dialog między dwoma ludźmi. Kandydat jako człowiek jest w jakimś stopniu na równi z rekruterem. Dlatego macie prawo mówić o swoich dyskomfortach, o tym (to co też Wam wcześniej powiedziałem), że np. nienawidzicie live codingu i wiecie, że się zestresujecie. Powiedzcie o tym. To jest bardzo ważne, żeby pokazać

siebie takim, jakim się jest, nie pozwolić na to, żeby stres nas przyćmił. Dlatego warto o tym mówić otwarcie: że podczas rozmów się stresujecie, że staracie się udzielać odpowiedzi w zgodzie z tym, co wiecie, ale czasami możecie tę wiedzę zgubić właśnie przez to, że się stresujecie.

Może się wydawać, że prosicie o jakąś taryfę ulgową. Nie. Wy prosicie o to, żeby ewentualnie zadawać pytania w inny sposób, nadal weryfikujący Waszą wiedzę. Pamiętajcie, że rozmowa rekrutacyjna to nie jest koniec świata i nawet gdy Wam się nie uda, to nic się nie dzieje. Oczywiście jeżeli ktoś nie ma pracy i ma jakąś trudniejszą sytuację, to coś się pewnie dzieje. Jednak pamiętajcie, że branża IT jest branżą bardzo rozwojową. Tutaj programistów szukają, programistów nadal brakuje, na rynku ofert są setki, jak nie tysiące. Możecie próbować w wielu miejscach, a teraz jeszcze w czasach pracy zdalnej, to już w ogóle. Warto wyjść z założenia, że jeżeli nie tutaj, to gdzie indziej.

Mówiłem już też o tym, że rozmowy, które są zakończone porażką, to jest dla Was czysta nauka. Nauka tego, że wiecie, jakie pytanie padło. Jeżeli zapytaliście o rozwiązanie (załóżmy, że nie wiedzieliście, w jaki sposób odpowiedzieć na pytanie) i ktoś Wam o tym powiedział, to już macie wiedzę na tacy. Starajcie się takie odpowiedzi wyłapywać, zapamiętywać, może doczytać sobie coś w domu – no i już jesteście bogatsi o nowe doświadczenia.

Podejście bardziej na luzie jest tutaj istotne, żeby nie być takim więźniem tego, że to jest firma, do której musimy się dostać tu i teraz. Ja nie mówię, że nie mamy mieć jakichś wzorców czy idealnych miejsc pracy, tylko że jeżeli nie uda nam się dzisiaj, to spróbujemy za rok, może

za dwa lata. Jest mnóstwo osób, które próbują kilkukrotnie dostać się do danej firmy. Rozmowa rekrutacyjna to nie jest koniec świata. Tu się powtórzę celowo, żeby to wybrzmiało. Pisałem o tym zresztą artykuł – już też o tym powiedziałem. Jeżeli ktoś będzie chciał sobie doczytać, to zachęcam do zajrzenia tam. Starałem się to tam opisać bardziej dokładnie – w jaki sposób można zadbać o to swoje przygotowanie mentalne.

To jak już jesteśmy przy przygotowaniu mentalnym i stresie, to niestety ten stres się pojawia i następstwem tego jest pustka w głowie. Nagle zapominamy wszystkiego, nawet najprostszych metod. Może podpowiesz nam jeszcze, co wtedy zrobić?

Pustka w głowie przede wszystkim nie jest niczym złym, bo to jest reakcja naszego organizmu na daną sytuację, chociażby na ten stres. Zachęcam, żeby przede wszystkim nie nawijać makaronu na uszy, jeżeli się czegoś nie wie, i (to jest też ważne) trzeba się zastanowić, czy to wynika ze stresu, czy mamy pewność, że tego nie wiemy. Wówczas po pierwsze śmiało odpowiedzieć: *nie wiem tego*, a nie opowiadać jakieś niestworzone rzeczy, krążyć wokół tematu, który może zupełnie tego nie dotyczyć.

Jeśli ta pustka w głowie wynika ze stresu, warto się po prostu do tego przyznać. Powiedzieć: *wiesz co, wydaje mi się, że to wiem, ale w tym momencie bardzo się stresowałem i nie mogę sobie tego przypomnieć*. Jeżeli zdecydowanie czujemy, że ten stres jest bardzo wysoki, to poprośmy o parę minut przerwy. Oczywiście nie mówimy tu o 10-15 minutach, ale o trzech minutach. Po prostu siedząc, zrób kilka wdechów,

pomyśl sobie o czymś innym – to zdecydowanie pomaga. Ktoś może powiedzieć: *co mi dadzą trzy minuty, jeżeli ja jestem zestresowany, cały spocony* itd. Proponuję spróbować i wtedy stwierdzić, czy to pomaga, czy nie.

Jest jeszcze taka opcja, że można poprosić o wskazówkę. To też nie jest nic złego. Można śmiało powiedzieć: *jestem zestresowany. Czuję, że to wiem. Czy możesz mnie nakierować nawet najprostszą wskazówką?* Wtedy nasz mózg najprawdopodobniej znowu zachwyci, przypomnimy coś sobie i będziemy w stanie dorzucić coś do zadanego pytania.

No dobrze, to chyba mamy już komplet jeśli chodzi o te tematy mentalne.

Chciałbym jeszcze na koniec zapytać Cię o trendy w rekrutacji technicznej. Czy widzisz, że coś się zmienia? Może odchodzi się od rzeczy, które dotychczas były standardem lub firmy zmieniają oczekiwania względem kandydatów? Jak to jest z Twojej perspektywy?

Z mojej perspektywy ten trend zdecydowanie się zmienia i to widać chociażby po wpisach, które tworzą osoby rekrutujące np. na LinkedInie. Dużo z nich otwarcie mówi o tym, że mamy czasy, gdzie kandydat jest weryfikowany pod kątem jego predyspozycji do rozwoju, a nie takiej twardej wiedzy technicznej. Dlaczego? Po pierwsze rynek cały czas potrzebuje programistów. Tacy zawodowi programiści, powiedzmy z wykształceniem, z doświadczeniem zawodowym wcale nie pojawiają się tak szybko, jak rynek by tego chciał. Dlatego firmy chcą próbować

znaleźć kandydatów, którzy mają szansę na to, żeby przy ich pomocy zyskać pewien wachlarz umiejętności i dzięki temu być dobrym pracownikiem oraz pojawić się na tym rynku.

Rekruterzy zdecydowanie patrzą na to, czy ja chcę się rozwijać, w jaki sposób ja się rozwijam, gdzie szukam wiedzy, czyli czy np. biorę udział w konferencjach branżowych, w jakichś lokalnych wydarzeniach, czy czytam jakieś branżowe książki, a może artykuły itp. To jest bardzo ważna cecha, bo jeżeli biorę sobie osobę, która nie robi nic sama z siebie, czyli nie chce wpłynąć jakoś na swój rozwój, to obawa jest taka, że mimo tej wiedzy twardej, którą ona posiada, nie ruszy do przodu. Wiemy o tym, że to jest branża dynamiczna, więc musi ruszyć do przodu, żeby nie dostarczać niekompatybilnych albo chociażby starych rozwiązań.

Z takich rzeczy, które jeszcze przychodzą mi na myśl, to na pewno (na szczęście też dla mnie) live coding odchodzi powoli do lamusa. Odchodzi się od tej metody właśnie dlatego, że nie jest to weryfikacja naszych umiejętności zgodna ze środowiskiem programistycznym. Już o tym wspominałem: nikt w pracy przy nas nie siedzi i nie patrzy, co my robimy. Warto jest weryfikować daną osobę pod kątem tego, jak ona pracuje w takim środowisku, w jakim my pracujemy jako ta osoba rekrutująca.

OK, no dobra, to w takim razie widać, że wszystko idzie w dobrą stronę, przynajmniej jeśli chodzi o live coding. Muszę przyznać, że nie raz już spotkałem się z informacją, że to nie działa – w sensie po stronie kandydata. Wszyscy się niepotrzebnie denerwują, stresują i w ten sposób możemy stracić naprawdę wartościowych

pracowników – tylko przez to, że ktoś im patrzy na ręce, a wiadomo, tak jak sam wspomniałeś, że tak nie pracujemy. Wtedy nie jesteśmy w stanie pokazać swoich 100% możliwości.

Oczywiście, że tak. Programiści to są osoby, które siedzą, popijają kawkę, a nie stresują się, patrząc w monitor, więc niech tak będzie.

Niech tak będzie.

Na koniec zapytam Cię o książkę, jaką mógłbyś polecić osobie, która chce przejść etap rekrutacji technicznej.

To jest trochę takie pytanie wstydu, bo nie mam tutaj konkretnej odpowiedzi. Tak jak powiedziałem, ja nawet nie chcę nawoływać do tego, żebyście czytali jakieś konkretne książki, które mają Was w jakiś sposób przygotować do rekrutacji technicznej. Dla tych bardziej ambitnych: oczywiście można przejrzeć sobie książki z pytaniami rekrutacyjnymi, nawet z danego języka. Z tego, co się orientuję, są takie, że jeżeli idziecie na rozmowę jako front-endowiec, to macie książkę z pytaniami rekrutacyjnymi na przykład z JavaScriptu. Ale to jest naprawdę dla takich turboambitnych, którzy może studiowali medycynę. Tutaj puszczam oczko, bo osoby na uniwersytetach medycznych są świetne w czytaniu takich dużych biblii.

Jeśli jednak chodzi o konkretną książkę, którą ja bym przeczytał, to znowu powiem trochę przewrotnie: przeczytajcie coś, co lubicie. Niech to Was zrelaksuje, wyluzuje. Jeżeli lubicie czytać fantastykę, to czytajcie fantastykę. Macie być osobami, które są zrelaksowane, które idąc na rozmowę rekrutacyjną, nie stresują się, czy na pewno wszystko

przeczytały, czy na pewno wszystko zweryfikowały i czy na pewno wszystko wiedzą. Może więc tak odpowiem bez konkretnego tytułu.

Dobrze, dobrze, nie ma problemu. Tak że zapraszam do czytania, żeby się po prostu zrelaksować i położyć wcześniej do łóżka, żeby się wyspać i być gotowym na rekrutację.

To już ostatnie pytanie na dzisiaj. Gdzie możemy Cię znaleźć w sieci?

Mówiłem już o tym, że działam na LinkedInie i to jest moje główne centrum dowodzenia. Tam możecie znaleźć wpisy, które pojawiają się prawie codziennie, ale także artykuły takiego, powiedziałbym, większego nakładu – czyli staram się, żeby te treści były dokładne, rzetelne i żeby dotyczyły takich rzeczy, które mogą starczyć na dłużej.

Od niedawna jestem też na Instagramie, jednak tam dopiero zaczynam, więc jeżeli ktokolwiek chciałby mnie znaleźć i się ze mną skontaktować, to zachęcałbym jednak do korzystania z LinkedIna, bo tam przez większość dnia jestem dostępny chociażby nawet z telefonu. Jak ktoś do mnie pisze, to zawsze mogę przeczytać.

Linkedin to też miejsce, gdzie prowadzę biuletyn, o którym wspomniałem: „Kariera Developera”. Tam pomagam programistom wskoczyć na kolejny poziom ich programistycznej kariery. To jest właśnie miejsce, gdzie publikuję swoje artykuły. Nie mam żadnej strony zewnętrznej, więc nie szukajcie pod RadekWojtysiak.pl, bo po prostu niestety nie mam na to czasu. Tylko i wyłącznie więc LinkedIn. Na ten moment to jest moje miejsce domowe.

Tak jest, tak że zapraszamy do Radka na LinkedIna, a jeżeli nie macie, to musicie założyć, bo programista bez LinkedIna nie istnieje (oczywiście trochę się śmieję).

To ja, Radku, chciałbym Tobie podziękować za naszą dzisiejszą rozmowę. Super rozmowa, super informacje i kawał dobrej roboty. Dzięki jeszcze raz i mam nadzieję, że to nie ostatni raz kiedy się słyszymy i widzimy.

Również mam taką nadzieję, że to jest pierwszy, ale nie ostatni raz. Także bardzo dziękuję za spotkanie. Fajnie, że mogłem dorzucić tutaj jakiś swój punkt widzenia na tę rekrutację i trzymam też kciuki za Was wszystkich, za słuchaczy tego podcastu, żebyście podchodzili do rozmów rekrutacyjnych bardziej na luzie. Traktujcie to jako weryfikację tego, co potraficie, a jeżeli Wam się nie uda, to traktujcie to jako naukę i szansę na rozwój. Jeżeli nie tutaj, to gdzieś indziej.

Jeszcze raz, Mateusz, bardzo dziękuję za rozmowę.

Dzięki, cześć.