Soft Skills

Zusammenfassung von Justin Jagieniak.

Career

Working remotely survival strategies (S.89-93)

In diesem Unterkapitel diskutiert der Autor über Strategien für den Fall, dass man von zuhause aus arbeitet. Er spricht davon, dass heutzutage eine Menge unabhängiger Software-Entwickler per Remote-Desktop oder über ein virtuelles Büro von zuhause aus arbeiten. Er geht dabei auf die Vor- und Nachteile der Heimarbeit ein. Als Nachteil sieht er zum Beispiel die Isolation, Einsamkeit und mangelnde Selbstmotivation. Er erwähnt, dass er anfangs die Vorteile dabei sah, aus seinem Bett morgens direkt an den Arbeitsplatz zu kommen. Sagt aber auch, dass er die entstandenen Herausforderungen unterschätzt hat.
Die erste Herausforderung für ihn ist das Zeitmanagement. Er behauptet, dass es bei der Arbeit zuhause eine Menge Arten von Ablenkung gibt, die einen daran hindern die Arbeit zu erledigen. Auch neigt man angeblich dazu, seine Arbeit später am Abend zu erledigen, wenn man sich die Zeit abseits vom Alltagsstress nehmen kann. Seiner Meinung nach endet das aber in einem Desaster. Er behauptet, wenn man von zuhause aus arbeiten will, muss man ein vernünftiges Konzept für das Zeitmanagement erstellen.
Eine weitere Herausforderung ist die Selbstmotivation. Er empfiehlt Leuten die Probleme mit Disziplin und Selbstkontrolle zu haben, nicht von zuhause aus zu arbeiten. Er meint, dass wenn man im Büro arbeitet, der Chef beobachten kann, wie man seine Arbeit erledigt. Wenn man allerdings von zuhause aus arbeitet, kann dies der Chef nicht. Das heißt, dass man für seine Disziplin und Motivation selbst verantwortlich ist. Er hält eine Planung und Routine der Zeiten bei der Heimarbeit für sehr wichtig, vorallem für die Phasen bei denen man nicht motiviert ist. Auch sollte man Versuchungen und Ablenkungsmöglichkeiten von seinem Arbeitsplatz entfernen. Wenn man sich absolut unmotiviert fühlt, schlägt er vor eine Zeituhr auf 15 Minuten zu stellen und sich in diesen 15 Minuten zu seiner Arbeit zwingt. Meist ist es der Fall, dass man nach jenen 15 Minuten mehr motiviert ist seine Arbeit fortzusetzen. Ein weiteres Hilfsmittel ist das Nutzen der Kommunikation mit Mitarbeitern per Google Hangouts oder Skype, um nicht sozial von der Arbeit abgehängt zu werden.

Fake it till you make it (S.94-97)

Hier spricht der Autor davon, dass man immer wieder auf Herausforderungen und Grenzen stößt, auf die man nicht vorbereitet ist. Er sagt, dass es einige Leute gibt, die der Herausforderung lieber ausweichen und andere die sie annehmen und kämpfen. Er behauptet, dass es nicht an ihrer Selbstsicherheit und Fähigkeit liegt, Erfolg zu haben, sondern dass sie alle in der Lage sind vorzutäuschen, dass sie es schaffen. Er meint, dass das Vortäuschen etwas zu Können bis man es schafft, etwas mit Selbstsicherheit zu tun hat. Man handelt nach einem großen Glauben an sich selbst alle Hürden überwinden zu können. Man muss dazu bereit sein ins große Unbekannte springen zu wollen, auch wenn man dann nicht weiß was man tun muss und Angst dabei empfindet. Er hält es nämlich für unmöglich als Softwareentwickler ein Experte für alles zu sein. So würden auch die meisten Jobinterviews Fähigkeiten erfordern, die man noch nicht besitzt. Er meint, dass das Schlüsselwort hier das "noch nicht" ist, dass es wichtig ist, sein Auge auf die Zukunft zu richten statt auf das, was man noch nicht kann. Daher ist es wichtig eine Aura von Selbstsicherheit auszustrahlen mit dem Wissen, dass man Herausforderungen in der Vergangenheit gemeistert hat und es keinen Grund gibt, dass man

sie nicht auch in der Zukunft meistern kann. Weiterhin soll man nach Meinung des Autors kein Lügner sein, sondern ehrlich mit den eigenen Fähigkeiten sein und zeigen, dass man in der Lage ist mit Hindernissen umzugehen.

Resumes are boring - Let's fix that (S.98-102)

Im nächsten Unterkapitel behauptet der Autor, dass der durchschnittliche Lebenslauf eines Softwareentwicklers ein fünfseitiges monströses Dokument ist, welches eine Schriftart und zwei Spalten hat, und mit grammatikalischen Fehlern, Tippfehlern und erbärmlich strukturierten Sätzen voll mit Phrasen, wie "anführend" und "auf Ergebnisse fokussiert" gefüllt ist. Er vergleicht dies mit einem professionellen Lebenslauf und kommt zum Ergebnis, dass man lieber einen professionellen Lebenslauf-Schreiber anheuern sollte. Dies begründet er damit, dass das Schreibenlernen eines professionellen Lebenslauf eine Verschwendung von Zeit und Talent ist und dies besser in professionelle Hände gehört. Bei der Beauftragung eines professionellen Schreibers ist seiner Meinung nach darauf zu achten, dass er technisches Know-how besitzen sollte und auch Beispiele aus dem technischen Bereich zeigen kann. Bei der Beauftragung selbst sollte man beachten, dass man so viele Informationen wie möglich von sich selbst gibt und versucht sich selbst in einen positiven Licht mit den Informationen darstellen zu lassen. Auch ist es wichtig einen Format zu wählen, das für den Leser einfach verständlich ist.
Weiterhin regt der Autor an, dass man seinen Lebenslauf, wenn es fertiggestellt ist, auch online auf einer Homepage und Portalen wie LinkedIn zur Verfügung stellen sollte. Auch dies würde häufig von einer professionellen Fachkraft übernommen.
Wenn man trotzdem seinen Lebenslauf lieber selbst schreiben will, rät der Autor folgendes zu beachten: Man soll seinen Lebenslauf online stellen, ihn in einzigartiger Weise präsentieren, vorherige Projekte im Lebenslauf präsentieren und er sollte frei von Tipp- und Sprachfehlern sein.

Don't get religious about technology (S.103-106)

Im letzten Unterkapitel warnt der Autor davor bei Technologien religiös zu werden. Damit meint er, dass Leute, die eine Technologie, Programmiersprache oder Software kennen, gerne an dieser festhalten ohne offen für etwas anderes zu sein. Er hält dieses Verhalten für destruktiv und limitierend. Wir würden einen Punkt erreichen, an dem wir aufhören zu "wachsen" und meinen auf alles eine Antwort gefunden zu haben. Er sagt von sich, dass er aufgehört hat religiös zu sein und in einem Punkt seiner Karriere verschiedene Betriebssysteme, Programmiersprachen und Texteditoren ausprobiert und gelernt hat, bevor er sich für seine beste Technologie entschieden hat.
Der Autor meint aus seiner Perspektive betrachtet, dass nicht alle Technologien großartig sind, sondern manche nur gut sind. Von Zeit zu Zeit kann sich dies auch ändern. Er sagt, dass es nicht nur die eine gute oder gar die beste Lösung für ein Problem gibt. Man entscheidet danach, was man am besten findet, aber das heißt nicht notwendigerweise, dass es das Beste ist. Der Autor erzählt, dass er in seiner früheren Zeit darüber debattierte, ob Microsoft oder Mac besser ist, C# besser ist als Java oder statische Programmiersprachen besser sind als dynamische. Er sagt, dass seine eigene Ansicht falsch war. Er hat in den letzten Jahren an einem Javaprojekt gearbeitet und gelernt für alles offen zu sein. Er versucht nun die Dinge auszuprobieren, bevor er eine Entscheidung trifft. Sein Punkt dabei ist, dass man in seinen Optionen keine Limits setzen sollte. Man sollte nicht seine Wahl der Technologie für die beste erklären und alles andere ignorieren. Es wird seiner Meinung nach am Ende einem nur weh tun. Andererseits sagt er, dass nur dann wenn man gewillt ist offen zu sein und man etwas für das Beste erklärt, andere einen folgen werden.