TRANSKRIPT | No / Low Code, Vibe Coding, AI: Wie sich die Zukunft der Softwareentwicklung ändert (#130 Alumni Talk, Podiumsdiskussion)

Bella Kitzwögerer: Im Technikum Podcast reden wir nicht nur über Technik – wir leben sie. Entdecken Sie mit uns zweiwöchentlich, wie Wissenschaft, Technologie und Innovation unsere Zukunft gestalten.

Rafael Rasinger: In dieser Episode sind wir heute bei der Veranstaltung „Alumni Talk“ im Festsaal der FH Technikum Wien vor Ort dabei. Wir sprechen über Softwareentwicklung, die sich in einem grundlegenden Wandel befindet. Low-Code- und No-Code-Plattformen ermöglichen es, komplexe Anwendungen ohne tiefgreifende Programmierkenntnisse zu realisieren. KI-gestützte Entwicklungswerkzeuge automatisieren Routineaufgaben, generieren Code oder sogar ganze Applikationen.

Florian Eckkrammer: Herzlich willkommen bei uns an der Fachhochschule Technikum Wien zur Alumni-Veranstaltung „Low Code / No Code“. Mein Name ist Florian Eckkrammer, ich bin einer der beiden Geschäftsführer der Fachhochschule. Es freut mich sehr, dass ich das heutige Thema eröffnen und ein paar einleitende Worte sagen darf. Als gelernter Wirtschaftsinformatiker ist das natürlich ein Thema, das mich besonders interessiert. Ich sehe hier sehr viele Wirtschaftsinformatikerinnen und Wirtschaftsinformatiker, und es freut mich sehr, dass Sie wieder zu uns gekommen sind. Natürlich begrüße ich auch alle anderen, die vielleicht etwas anderes studiert haben – auch wenn ich Sie vielleicht nicht persönlich kenne. Low Code und No Code sind für uns auf mehreren Ebenen spannend. Einerseits verändern sie ein Berufsfeld, für das wir ausbilden wollen. Andererseits müssen wir überlegen, wie wir unsere Ausbildung gestalten. Generell stellt sich die Frage: Wie soll die Ausbildung hier bei uns an der Fachhochschule Technikum Wien aussehen, wenn sich einerseits die Erwartungen der Studierenden verändert und andererseits auch die Erwartungen der Unternehmen? Firmen wünschen sich oft Bewerberinnen und Bewerber, die achtzehn Jahre alt sind, aber zwanzig Jahre Erfahrung haben und bereits ein Senior-Level erreichen. Das ist natürlich nichts Neues – das ist ungefähr so, als würden Ostern und Weihnachten zusammenfallen. Die KI und alles, was jetzt in der Softwareentwicklung passiert, verstärkt diese Entwicklung aber noch. Auch Studierende fragen sich: Warum muss ich das lernen? Das kann doch die KI schon machen. Für uns bedeutet das: Wir sollen keine Juniors mehr produzieren, sondern möglichst schnell Menschen ausbilden, die auf Senior-Level arbeiten können – also den Weg über den klassischen Junior teilweise überspringen. Wir überlegen daher intensiv, wie wir Studierende in sechs Bachelorsemestern – und gegebenenfalls weiteren vier Mastersemestern – so weit bringen können, dass sie danach gut in den Arbeitsmarkt einsteigen. An der Fachhochschule Technikum Wien haben wir einen besonderen Vorteil: Wir betreiben Berufsfeldforschung direkt im Haus. Günter Essl wird dazu gleich noch

Einblicke geben, bevor wir später in die Podiumsdiskussion gehen. Wir analysieren sehr genau den Arbeitsmarkt und versuchen, diese Erkenntnisse in unsere Curricula einfließen zu lassen. Gerade arbeiten wir an einem großen Projekt zur Zukunft der Lehre an der Fachhochschule Technikum Wien. Dabei überlegen wir studiengangsübergreifend – über alle Bachelor- und Masterstudiengänge hinweg –, welche Erwartungen unterschiedliche Stakeholder an uns haben: Studierende, Interessenten, Lektorinnen und Lektoren, aber natürlich auch Unternehmen. Die zentrale Frage lautet: Welche Kompetenzen sollen Studierende mitnehmen, wenn sie bei uns ihren Bachelor oder Master abschließen? Aktuell zeichnet sich eine klare Richtung ab: Wir möchten einen sehr starken projektbasierten Anteil im Studium haben – also Experimental Learning. Idealerweise arbeiten Studierende aus unterschiedlichen Disziplinen zusammen. Maschinenbauer, Elektrotechnikerinnen, Mechatroniker und Informatiker könnten gemeinsam an Projekten arbeiten. Dabei lernen sie Teamarbeit, so wie sie später auch im Arbeitsumfeld notwendig sind, und sie lernen vor allem durch das Tun. Natürlich stellt sich die Frage, ab welchem Zeitpunkt man Studierende so stark eigenständig arbeiten lassen kann. Ein gewisses Grundlagenwissen müssen wir weiterhin vermitteln. Danach möchten wir Studierende stärker begleiten und coachen. So können wir auch besser mit der Problematik umgehen, dass heute teilweise Abgaben durch KI generiert werden oder Low-Code-Tools eingesetzt werden, um Leistungen vorzutäuschen. Wenn Studierende stattdessen an echten Projekten arbeiten, können sie KI als Werkzeug sinnvoll einsetzen, um Aufgaben zu lösen. Gleichzeitig können wir beobachten, wie sie arbeiten, sie besser coachen und ihre Leistungen realistischer bewerten. Das ist im Moment eine Zukunftsidee, die wir gerade gemeinsam mit allen Mitarbeiterinnen und Mitarbeitern im Haus detailliert ausarbeiten. Ich bin sehr gespannt, wohin uns diese Entwicklung führen wird. Vielleicht können wir bei einem späteren Alumni-Talk darüber berichten. Bevor ich Sie aber noch länger auf die Folter spanne, würde ich sagen: Günther, komm doch zu uns. Du hast die ursprüngliche Idee zu Low Code und No Code in einem Think Tank bei uns eingebracht. Daher ist es, glaube ich, eine gute Idee, wenn du uns einen kurzen Überblick gibst. Ich wünsche uns allen einen spannenden Abend und viele neue Erkenntnisse. Bitte, Günter.

Günter Essl: Danke, Florian. Wir sind sehr froh, dass es dieses Alumni-Talk-Format gibt. Denn Sie sind die Praxiserfahrenen. Sie wissen genau, was sich gerade in Ihrem Unternehmen gerade verändert. Und genau davon lebt auch unsere Berufsfeldforschung – und letztlich auch unsere Curricula. Die Berufsfeldforschung versucht möglichst schnell aktuelle und zukünftige Entwicklungen zu erfassen. Einerseits quantitativ über Arbeitsmarkterhebungen, andererseits qualitativ durch Gespräche mit Unternehmen. Wir wollen verstehen: Wie läuft es momentan wirklich in der Praxis? An dieser Stelle möchte ich auch Rafael Rasinger danken, der mit innovativen Formaten den Austausch zwischen Hochschule und Praxis stärkt. Ein solches Format ist bei uns auch der sogenannte „Do Tank“. Vor etwa einem Jahr saßen wir zusammen und haben uns gefragt: Welche neuen Entwicklungen müssen wir stärker beobachten und in den internen Diskurs einbringen? Eine der zentralen Fragen war: Wie werden Low Code und No Code die Dynamik im Softwaremarkt verändern? Schon damals war diese Dynamik relativ groß – auch wenn vieles noch wie Zukunftsmusik klang. Heute ist es bereits Arbeitsrealität.

Low Code bietet Chancen zur Partizipation. Menschen, die weniger Erfahrung im Programmieren haben, können trotzdem Anwendungen erstellen und an Entwicklungsprozessen teilnehmen. Gleichzeitig löst das auch Verunsicherung aus – besonders bei Softwareentwicklerinnen und Softwareentwicklern. Ein wichtiger Punkt ist daher auch eine klare Begriffsklärung. Low Code bedeutet nicht automatisch, dass Programmieren vollständig delegiert werden kann. Das konzeptionelle Denken bleibt zentral. Genau davon lebt Programmierung. Damit stellt sich eine größere Frage: Wie weit wollen wir unser Denken an KI delegieren? Das betrifft nicht nur technische Fragen, sondern auch Fragen beruflicher Identität und sogar ethische Dimensionen. Ist die Auslagerung von Denken an KI überhaupt möglich? Und wenn ja – wie weit wollen wir gehen? Diese Fragen werden auch Gegenstand eines Forschungsprojekts sein, das wir gemeinsam mit Unternehmen durchführen wollen. Meine Kollegin Denise Bärnfeind und ich werden dazu qualitative Erhebungen durchführen, um besser zu verstehen, wie sich berufliche Anforderungen verändern – nicht nur in der Softwareentwicklung, sondern insgesamt. Wir werden daher sicherlich noch auf einige von Ihnen zukommen. Ein letzter Punkt: Spannend wird auch die Frage sein, ob künftig tatsächlich alles berechenbar ist – also ob alles in Algorithmen abgebildet werden kann. Und besonders wichtig für Studierende ist die Frage: Welche Kernkompetenzen bleiben langfristig relevant? Wie gehen wir künftig mit unseren eigenen Denkprozessen und der zunehmenden Technologisierung um? Damit eröffne ich die Bühne und freue mich auf die weiteren Präsentationen und die Paneldiskussion. Vielen Dank.

Rafael Rasinger: Danke, Günther. Vielen Dank. Herzlich willkommen zum Alumni Talk. Danke für die begriffliche Einordnung und auch für die Einblicke, was wir als Fachhochschule planen. Wir bleiben modern, entwickeln uns weiter und stellen uns den neuen Themen unserer Zeit. Herzlich willkommen also zum Alumni Talk „Low Code, No Code, Vibe Coding“ und vielen weiteren Entwicklungen. Mein Name ist Rafael Rasinger. Für alle, die mich noch nicht kennen: Ich bin an der FH Technikum Wien für Innovation zuständig, betreue unsere Scale-ups und bin auch Alumni-Beauftragter. Ich bin selbst Absolvent dieser Hochschule. Damals habe ich Elektronik studiert und betreue heute unter anderem Firmenpartnerschaften. Deshalb freue ich mich sehr, heute – mit meinem Alumni-Hut sozusagen – eine hochkarätige Runde begrüßen zu dürfen. Später wird Bernhard Wallisch durch die Paneldiskussion führen, gemeinsam mit unter anderem Albert Kroisleitner. Eine kurze Frage ins Publikum: Wer von Ihnen hat schon einmal von Peter Steinberger gehört?

(Publikum reagiert)

Gut, das war zu erwarten bei dieser Zielgruppe. In Zeiten wie diesen kann man sich der KI nicht verschließen. Die Frage ist vielmehr: Wie kann man mit KI sinnvoll arbeiten? Genau darum geht es heute Abend. Wir wollen verschiedene Perspektiven kennenlernen. Zunächst hören wir zwei Beiträge, bevor wir in die Paneldiskussion gehen. Und damit wechseln wir jetzt kurz ins Englische.

A very warm welcome to Alex Evans from Amazon Web Services. You will tell us how you use AI-assisted coding in your team and in your daily work. Welcome – great to have you here.

Alex Evans: Hi everyone. My name is Alex Evans. I'm a Cloud Infrastructure Architect in IT Professional Services. Thank you, Raphael, for inviting me to come here and speak. I'm part of the global automotive team, which means I focus on the automotive industry. My presentation is titled "From Billable Hours to Superpowers." For the last twelve years I've been working in professional services, mainly around customer success. My role involves working hand in hand with customers to build, design and deliver implementations for their solutions. Over the last two years there has been a tectonic shift in how we work. As one of the largest hyperscalers in the world, we face a lot of pressure to demonstrate strong capabilities in the area of AI. Let me quickly move to the slides so you can see them. A bit about myself: I'm Irish, I spend a lot of time doing sports in my free time, and I have three kids – one is nineteen years old. That means I don't actually have a lot of free time left for coding anymore.

My agenda today is fairly simple. I'll briefly go over my background and credentials, then look at the current business drivers, how things currently work in practice, what has changed recently, and the benefits of using AI agents. Finally, I'll share what I believe will be some of the big developments over the next one to two years. As I mentioned before, there is a lot of pressure from leadership within Amazon. We are a company with over one and a half million employees globally. AWS itself is only a fraction of that, but in professional services in particular there is strong pressure to demonstrate value. We charge significant amounts for working with customers, so we need to show clearly that we bring real value. At the same time, customer expectations are increasing rapidly. Because many tools are now freely available, and tasks can be completed ten or even one hundred times faster than a few years ago, expectations have changed dramatically. For me, this means that almost every week there are new internal guidelines, new frameworks or new recommendations. It's not only the tools that are changing – the rules about how we use them are changing as well.

To give you an anecdote: every year we present at our big internal IT conference. Two years ago we were told that at least 80 percent of all presentations had to be about AI. That shows how strong the push toward this topic is. Last year I had the opportunity to run a workshop on Kiro, which is our internal AI agent solution. Amazon is also famous for what we call "dogfooding" – meaning that we use our own products internally before releasing them publicly. AWS itself originally started as an internal Amazon product before it became widely available and eventually took over the market. Another anecdote: just two weeks ago we had an emergency security refresher session because of new AI tools coming out. Everyone had to be trained again on what we are allowed to use on our corporate laptops and what we are not allowed to use. In my daily work I now mostly use agent-first development. That means I'm often not writing code directly anymore. Instead, I guide AI agents and instruct them how to build the solutions. Another important point in my

talk is the value of using different tools. For example, you might use Kiro as a frontier autonomous agent, but you might also use AGP.

Quick question: who here knows the AGP protocol? AGP is an agent client protocol. It allows you to use an agent within different tools in your daily workflow without being locked into one specific platform.

In general, AI has taken over most of my working tools. Everything on the left side of this slide is now largely automated. This would be a typical technology stack for a customer: they might use Terraform, infrastructure-as-code tools, Kubernetes, and so on. My own background is more in serverless and event-driven design patterns, so I used to do a lot of serverless development. Much of that is now generated automatically. Unfortunately, the tasks in the middle – such as planning agile deliveries, meetings, and coordination across multiple communication channels – still require human involvement. One development approach that has become very powerful is spec-driven development. The advantage is that you can move most of the planning to the beginning. You break down the deliverables and structure how you want the agents to build the solution. One of the biggest current problems with agent-based coding systems is the context window. The more context you add, the less accurate the output becomes. Agents start skipping steps or producing incorrect results. With spec-driven development you can define requirements first. The system then generates a design, which you can review and iterate. After that, the tasks are automatically broken down and executed. These tasks can run in parallel, each in its own context session, which significantly improves reliability. Another important topic is the Model Context Protocol (MCP). Internally we were initially encouraged to use our own MCP builder, but it contained too many tools and overloaded the context, making it less useful. Six months later the situation improved significantly. MCP tools can now be loaded on demand, and there are many useful open-source tools available. For example, there is a system called Context7, which provides access to thousands of application documentations. This ensures that the AI always uses accurate documentation instead of relying only on training data from places like Stack Overflow. Another useful feature is hooks.

Hooks help control what the AI generates. With vibe coding you often generate large amounts of code without fully understanding how it was produced. Hooks allow you to run validation processes automatically. For example, when saving a file, the system can check whether a function already exists somewhere else in the code base and prevent duplication. There are many examples of how hooks can be used. Another important element is steering. Most tools now allow you to guide agents with configuration files that define which frameworks to use, coding styles, or architectural patterns. Hopefully, over the next year these formats will converge into a single standard that works across all agents. Looking ahead, my main predictions are the following: First, MCP will become much more widely adopted. People don’t want to be locked into one specific AI provider.

Second, language models are evolving daily. Ideally we will interact with them mostly through prompts or even voice interfaces. For example, I haven't yet tried Alexa Plus, but voice-based AI interaction may become much more common. Third, local coding models will become increasingly important. Hardware is improving and smaller models are becoming more efficient, so running local models for coding will soon be realistic. And finally, the next frontier will be fully autonomous agents. To give a quick anecdote: at a recent conference with over 100,000 participants, the main sponsor presented a solution that was essentially a fully autonomous agent that runs 24/7 and resolves issues automatically.

To summarize a few key takeaways: Space is changing extremely fast. You need to be ready to constantly adapt to new tools and new language models. You are never very far behind, because everything evolves so quickly. And finally: quality in, quality out. In enterprise environments it's crucial to provide structured prompts with the correct context and validation mechanisms to ensure reliable results. Thank you.

Rafael Rasinger: Thank you.

Alex Evans: Sorry, that was about twenty minutes of content compressed into a much shorter time.

Rafael Rasinger: You were great. Thank you very much for sharing your perspective. Just a quick question, Alex, for clarification.

Alex Evans: The context window is directly tied to the language model. Each model supports a maximum context size. The larger the model, the larger the context window it can handle. This is a common limitation across most models. Most of my experience is with Claude Code, partly because it's an AWS partner product and we are encouraged to use it internally – again, the dogfooding concept. Many of the mechanisms I described earlier exist specifically to break tasks into separate sessions so that the context window doesn't overflow.

Rafael Rasinger: The remaining ten percent of questions we can discuss later during the drinks. Thank you. And now we will have a quick startup pitch. Please welcome Alina Maxim, who will show us how AI can already be applied in the education sector.

Alina Taborsky: Hallo, ich bin Alina und das hier ist Levio. Wir haben eine psychologisch fundierte, KI-gestützte Lernplattform, die maßgeschneidertes Lernen ermöglicht. Die Idee entstand aus unseren eigenen Erfahrungen mit Lernen.

Maxim Dvoynishnikov: Ich war eigentlich immer unterfordert.

Alina Taborsky: Und ich war die meiste Zeit überfordert. Unterschiedliche Probleme – aber sie führen zur gleichen Frage: Was brauchen Menschen, damit Lernen ihren

individuellen Weg respektiert und genau dann Unterstützung bietet, wenn sie gebraucht wird?

Um diese Frage zu beantworten, haben wir ein kleines, aber sehr starkes Team aufgebaut. Ich übernehme die kreative Leitung sowie Strategie und Kommunikation.

Maxim Dvoynishnikov: Ich verantworte die technische Leitung und den Betrieb. Besonders wichtig ist uns, dass unsere Arbeit wissenschaftlich begleitet wird. Unser Advisory Board bringt Expertise aus Bildungsforschung und Bildungspsychologie ein, damit wir die Wirkung unseres Systems nicht nur versprechen, sondern auch sauber nachweisen können.

Alina Taborsky: Das Problem im Lernalltag ist sehr konkret: Oft merken wir erst am Ergebnis, dass etwas nicht funktioniert hat. Dann ist es meist zu spät – die Verständnislücke oder der Frust sind bereits entstanden. Lehrpersonen fehlt oft die Zeit, individuell zu differenzieren und zu analysieren, woran es tatsächlich liegt. Das ist belastend für einzelne Lernende – und teuer für die Gesellschaft. Die Folgen: Motivation sinkt, Potenziale bleiben ungenutzt und Talente bleiben verborgen.

Maxim Dvoynishnikov: Unsere Plattform verwandelt bestehende Lerninhalte in individuelle Lerneinheiten. Diese sollen Motivation steigern, Lernstress reduzieren und langfristig berufliche sowie soziale Chancen verbessern. Technisch basiert das auf einem zweistufigen Prozess. Im ersten Schritt werden hochgeladene Materialien in einen strukturierten Wissensgraphen überführt. Im zweiten Schritt wird dieser Wissensgraph genutzt, um adaptive Lerneinheiten zu erzeugen. Diese Einheiten können dynamisch an die Bedürfnisse der Lernenden angepasst werden. Dafür entwickeln wir ein eigenes Deep-Learning-Modell, das Lernsignale analysiert – zum Beispiel Navigationsmuster, Klickverhalten, Mausbewegungen oder freiwillige Selbsteinschätzungen.

Alina Taborsky: Das Ergebnis ist eine Plattform mit drei getrennten Zugängen. Im Lernportal begleitet Levio als Maskottchen dialogbasiert und proaktiv durch den Lernprozess. Lernende sehen dort auch ein Dashboard mit Fortschritten, Kompetenzentwicklung und einer Meta-Learning-Übersicht. Im Tutorportal können Lehrpersonen eigenes Material hochladen, strukturieren und an Kursteilnehmer verteilen. Im Autorenportal können Inhalte strukturiert und als lizenzierbare Kurse bereitgestellt werden.

Maxim Dvoynishnikov: Geschäftlich funktioniert die Plattform zweiseitig. Kursanbieter stellen Inhalte bereit. Organisationen oder Privatpersonen können Lizenzen erwerben. Anbieter zahlen für Aufbereitung und Wartung ihrer Kurse, und wir beteiligen uns über ein Revenue-Sharing-Modell an den Verkäufen.

Alina Taborsky: Gleichzeitig ist Levio auch die Lernumgebung selbst. Bildungseinrichtungen und Unternehmen zahlen monatlich pro Nutzerin oder Nutzer für Zugang, Verwaltung und Lernbegleitung. Wir entwickeln Levio, weil wir nicht länger

zusehen wollen, wie Menschen im Lernprozess verloren gehen. In jedem Menschen steckt mehr, als später in Noten, Zertifikaten oder Berichten sichtbar wird. Zu oft geht verloren, wie jemand denkt, welches Tempo jemand braucht und wie viel Mut es kostet, dranzubleiben. Gerade dort entscheidet sich, ob das Lernen wächst oder abbricht. Unser Ziel ist, dass jeder Mensch das Gefühl hat: Ich werde gesehen – so wie ich lerne.

Rafael Rasinger: Danke. Ja, und auch ihr seid nachher noch da, falls es Fragen oder Interesse gibt. Ich freue mich jetzt, als nächsten Programmpunkt die Paneldiskussion anzukündigen. Und zwar mit Bernhard Wallisch, unserem Panel-Moderator, Alumnus und Senior Lecturer an der FH Technikum Wien. Bernhard, übernimmst du gleich das zweite Mikrofon? Und du übernimmst auch die Vorstellung deiner Panel-Gäste. Viel Spaß und spannende Diskussion. Danke sehr.

Bernhard Wallisch: Und es freut mich wirklich, dass wir heute eine Paneldiskussion haben, bei der wir nicht nur Alumni, sondern auch Firmen begrüßen dürfen, die in diesem Umfeld arbeiten und auf das heutige Thema einen ganz eigenen Blick haben. Ich beginne mit Frau Dr. Svenja Schröder von der msg Plaut – ein großes Unternehmen im Bereich Consulting, Lösungsentwicklung und Softwareentwicklung. Ich will gar nicht zu viel vorwegnehmen und der Vorstellungsrunde nicht vorgreifen. Dann bitte ich Herrn Albert Kroisleitner auf die Bühne, Alumni aus unserem Haus, mit seiner Firma, die KI auch intensiv eingesetzt hat, um Produkte zu entwickeln. Ich bin schon gespannt auf Ihre Erfahrungen. Dann haben wir Herrn Dr. Christof Strasser von 9Devs. Unter anderem arbeitet er mit internationalen Softwareentwicklungsteams, vor allem auch mit Standort in Indien. Der internationale Aspekt dieser Entwicklung verspricht ebenfalls spannend zu sein. Und wir haben auch noch Speaker 10, Student im vierten Semester, also einen zukünftigen Alumni, der neben dem Studium bereits facheinschlägig arbeitet. Da bin ich auch schon sehr gespannt, was wir heute alles erfahren werden. Gut, namentlich durfte ich alle schon ankündigen. Jetzt möchte ich jedem von Ihnen die Möglichkeit geben, sich selbst, das eigene Unternehmen und vielleicht auch den Bezug zum Thema unserer heutigen Podiumsdiskussion – also die Frage, wie sich Softwareentwicklung verändern wird – in ein paar Worten vorstellen. Bitte sehr.

Svenja Schröder: Ja, mein Name ist Svenja Schröder. Ich freue mich sehr, hier zu sein. Vielen lieben Dank für die Einladung. Ich bin Head of Requirements Engineering bei der msg Plaut Cloud. Vielleicht zuerst kurz zum Unternehmen, ohne zu sehr auszuholen: Die msg Plaut ist eine Unternehmensberatung mit Sitz in Österreich. Wir haben rund zweihundert Mitarbeitende hier in Österreich, dazu eine große Muttergesellschaft in Deutschland, die msg systems, mit mehreren tausend Mitarbeiterinnen und Mitarbeitern. Wir sind also ein kleines, wendiges Schiff in einem großen Rahmen. Wir haben übrigens auch eine Partnerschaft mit dem FH Technikum, die wir rege nutzen und sehr schätzen. Warum passe ich gut in diese Runde? Ich glaube, wir haben es vom Kollegen davor schon gehört: Es wird immer wichtiger, Requirements sauber aufzuschreiben. Das merken wir auch bei uns im Alltag. Wir arbeiten stark mit Low-Code- und No-Code-Plattformen, zum Beispiel im Microsoft-Umfeld. Und es wird immer wichtiger, gerade wenn man KI nutzt,

genau zu definieren: Was ist eigentlich der Use Case? Was will man wirklich lösen? Jeder, der schon einmal mit KI gearbeitet hat, weiß das. Wir beschäftigen uns in unserer Abteilung daher sehr intensiv mit Fragen wie: Wie formuliert man Prompts richtig? Wie schneidet man Anforderungen so, dass am Ende ein echter Mehrwert herauskommt? Aber ich will nicht zu viel vorgreifen. Ich reiche das Mikrofon gerne weiter und freue mich auf die Diskussion.

Christof Strasser: Hallo, hallo. Ich bin Christof Strasser. Ich habe drei Hüte auf – und einen nicht. Ich bin nicht so wie ihr Techniker. Ich bin Jurist, ein bisschen Wirtschaftler, und mein Geld verdiene ich als Anwalt. Ich bin seit zwölf Jahren Startup-Anwalt in Österreich und habe in dieser Zeit viele Startups begleitet. Das Zweite, was ich gemacht habe: Vor acht Jahren habe ich begonnen, selbst Software zu entwickeln. Ich habe dabei die ganz bittere Schule durchlaufen – mit vier CTOs, bis ich schließlich bei meinem jetzigen Partner gelandet bin. Wir sind mittlerweile mit unserem größten Produkt online. Mein Partner ist Inder, hat acht Jahre in Europa als Senior Software-Architekt gearbeitet, und um ihn herum habe ich ein Team von Entwicklern in Chennai in Indien aufgebaut. Seit drei Jahren haben wir dort auch Rechtsanwälte – inzwischen vier –, weil wir sehr viel internationales Business machen. Was wir vor ungefähr eineinhalb Jahren zusätzlich begonnen haben, ist ein Employer-of-Record-Offshoring-Modell. Das heißt im Wesentlichen, dass wir Entwickler in Indien an westliche Unternehmen weiterreichen. Und überall liegt inzwischen die Schicht KI – sowohl in meinem Anwaltsbetrieb als auch in der Softwareentwicklung. Beides sind Knowledge Professions, und wir nutzen KI extrem stark. Ich selbst arbeite als Transaktionsanwalt wahrscheinlich vier bis fünf Stunden am Tag mit KI, und meine Developer ebenfalls. Ich glaube, dass wir dadurch eine Perspektive einbringen können, die vielleicht ganz interessant ist.

Arshia Reisi: Hallo, mein Name ist Arshia Reisi. Ich arbeite als Senior Consultant im Bereich Cyber Security und Penetration Testing bei KPMG. Wir sind eine internationale Beratungsfirma und bieten unter anderem Consulting im Bereich KI, Entwicklung und Cyber Security an und begleiten auch die Implementierung. Wie ich in meinem Alltag künstliche Intelligenz verwende, darüber werden wir später noch mehr sprechen. Aber im Endeffekt ist es ähnlich wie bei vielen anderen auch: Wir nutzen KI, um Tooling schneller zu entwickeln. Wir haben vorhin von AWS gehört – vom Schritt von „Billable Hours“ zu „Superpowers“. Und Billability ist natürlich auch in Consultingfirmen und generell in privatwirtschaftlichen Unternehmen ein sehr wichtiges Thema. Deshalb schauen wir natürlich auch, wie KI uns dabei helfen kann.

Albert Kroisleitner: Ja, danke. Hallo an alle, mein Name ist Albert – Albert Kroisleitner. Ich bin Vice President Product bei Fiskaly. Was ist Fiskaly? Wer von euch kennt Fiskaly? Doch einige – überraschend, aber gut.

Für alle anderen: Ihr hattet uns wahrscheinlich alle schon einmal in der Hand, wenn ihr einkaufen geht – beim Billa oder sonst irgendwo. Ihr bekommt einen Beleg, und auf diesem Beleg habt ihr in Österreich einen Code. Das heißt, dass diese Rechnung

fiskalisiert wurde. Genau das machen wir im Hintergrund. Und zwar nicht nur in Österreich, sondern auch in Deutschland, Spanien, Italien, Frankreich und vielen weiteren Ländern. Wir haben über 1,5 Milliarden Transaktionen pro Jahr – also da geht schon einiges drüber. Wir sind aber noch ein relativ junges Unternehmen, gegründet 2019 von drei sehr innovativen Leuten, die übrigens alle noch in der Firma sind – CEO, CTO und COO. Wir sind venture-capital-finanziert, das heißt, wir haben einen großen Investor aus den Nordics im Hintergrund, der uns beim Wachstum unterstützt. Mittlerweile sind wir ein Scale-up, kein Startup mehr. Was heißt das konkret? Wir sind im letzten Jahr um 100 Prozent gewachsen, haben aktuell rund 140 Mitarbeiterinnen und Mitarbeiter und etwa 30 Millionen Euro Umsatz. Und wir sind eine Rule-of-Fifty-Company, also stark wachsend und profitabel. Das ist für mich auch ein spannendes Stichwort in Bezug auf die Vorträge, die wir heute gehört haben. Unser Problem aktuell ist: Wir haben einen sehr guten Product-Market-Fit – aber wir sind zu langsam. Dabei sind wir eigentlich schon schnell. Aber eben trotzdem noch zu langsam. Wir müssen noch schneller werden, damit wir unsere Produkte möglichst rasch in weitere Länder ausrollen können. Europaweit gibt es 23 Länder mit fiskalischen Anforderungen, und jedes Land hat seine ganz eigenen Regeln. Hier schnell Lösungen anzubieten und schnell auszuführen, ist für uns extrem wichtig. Ich glaube aber, dass dieses Thema nicht nur uns betrifft. Überall sieht man Disruption. Branchen werden durch KI, Startups und Scale-ups verändert. Ich freue mich deshalb sehr auf die heutige Diskussion und bin gespannt, was ich selbst daraus mitnehmen kann. Mich würde auch interessieren, was ihr denkt und welche Fragen ihr im Zuge der Paneldiskussion einbringen wollt – damit es mehr Diskussion gibt und nicht nur Vortrag.

Bernhard Wallisch: Danke sehr. Das Stichwort Disruption möchte ich gleich aufgreifen. Ich bin Senior Lecturer an der FH Technikum Wien und selbst Absolvent der ersten Stunde. Ich habe 1999 hier abgeschlossen – das ist also schon eine ganze Weile her. Dazwischen war ich siebzehn Jahre direkt in der Softwareentwicklung tätig, in ganz unterschiedlichen Rollen – vom Entwickler über den Product Development Manager bis hin zum CEO. Jetzt bin ich wieder in der Lehre und es hat mich zurück ans Technikum gezogen. Und ich freue mich, dass wir gerade in dieser sehr interessanten Zeit – in der Disruption wirklich ein zentrales Thema ist – auch die Lehre aktiv mitgestalten können. Damit würde ich gern mit der ersten Frage an das Panel beginnen. Ich schlage vor, dass wir uns nach einer ersten Runde auch noch Zeit nehmen, auf Fragen aus dem Publikum einzugehen. Gut. Dann beginne ich mit einer bewusst provokanten Formulierung.

Ich darf bei Ihnen anfangen, Frau Dr. Schröder: Ist es in der heutigen Zeit noch sinnvoll, Informatik zu studieren?

Svenja Schröder: Nein. *Kleiner Spaß.* Natürlich ist es noch sinnvoll, Informatik zu studieren. Ich muss gestehen: Wir gehören zu den Firmen, die sich oft gern Seniors wünschen – oder Juniors mit zwanzig Jahren Erfahrung. Aber natürlich sehen wir auch, dass das nicht immer ein Vorteil ist. Ich könnte jetzt weit ausholen: Diverse Teams bringen mehr Erfahrung, und es bringt nichts, einfach nur irgendwelche Profile zusammenstoppeln. Aber ich glaube, Informatikerinnen und Informatiker von heute

müssen andere Skills lernen. Und ich glaube, dieses Wort Disruption begleitet uns auch bei den Kunden sehr stark. Denn die Kunden sagen heute oft: Wir würden gern KI machen. Und dann steht man da und denkt sich: Okay – was machen wir jetzt konkret? Wir setzen das natürlich für unsere Kunden um. Wir bieten Leistungen an und entwickeln Produkte. Aber das Ganze muss von Menschen begleitet werden, die sich wirklich auskennen. Man kann sich natürlich mit einem Tool hinsetzen, in einer virtuellen Maschine arbeiten und sagen: Hier sind meine Anforderungen, programmiere mir das runter. Aber man muss nachher trotzdem noch prüfen: Was ist dabei herausgekommen? Hat das die richtige Architektur? Kann ich das deployen? Erfüllt es die Compliance-Vorgaben? Ist es sicher? Und dafür braucht man echtes informatisches Know-how. Also nicht nur auf Code-Ebene, sondern auf dem Level von Architektur, Qualität, Sicherheit und Zukunftsfähigkeit. Dafür braucht es Informatikerinnen und Informatiker. Ich glaube, dass sie in Zukunft vielleicht weniger tief selbst programmieren müssen, aber sie müssen die Materie verstehen und wissen, wie man Systeme so aufsetzt, dass sie funktionieren, sicher sind und langfristig tragfähig bleiben. Das sind Skills, die man lernen muss. Also ja: Man braucht ein Informatikstudium. Aber man wird es wahrscheinlich an die aktuellen Gegebenheiten anpassen müssen.

Bernhard Wallisch: Ja, danke. Das lädt mich gleich weiter zu Dr. Strasser ein. Sie haben gesagt, dass Sie Entwicklungsteams in Indien aufgebaut haben. Wie wirkt sich diese Entwicklung auf Ihr Geschäftsmodell aus? Könnte jemand nicht sagen: Wenn die KI das ohnehin selbst macht – wozu brauche ich dann noch eine Outsourcing-Firma mit Softwareentwicklern?

Christof Strasser: Ich versuche meine Schwäche, kein Techniker zu sein, in eine Stärke umzuwandeln und das Ganze aus der Perspektive einer anderen Profession zu betrachten. Ich glaube, dass die Schere auseinandergeht. Juristen sind schon viel länger mit dem Problem konfrontiert, dass unsere Arbeit als Commodity angesehen wird – also als etwas, das man möglichst gut automatisieren kann. Und das stimmt in vielen Bereichen auch. Genau so ist es bei Software Development. Einfach nur „so dahin“ zu juristeln oder zu programmieren wird wahrscheinlich nicht mehr reichen. Du hast, glaube ich, zwei Möglichkeiten: Entweder du ergänzt das Ganze durch eine Doppelqualifikation. Dann bist du ein Entwickler, der Dinge kann, die andere heute oft nicht liefern – zum Beispiel Produktverständnis, ein Gespür für UX oder Marktverständnis. Wer nur sagt: Ich denke den Markt nicht mit, ich denke die Vermarktung nicht mit, ich mache mein Ticket fertig und gebe es weiter – das wird künftig nicht mehr reichen. Man muss multidisziplinärer denken. Bei Juristen ist es ähnlich: Nur noch im Gesetz nachzuschauen, was drinnen steht, wird nicht reichen. Du musst auch eine Bilanz lesen können und wirtschaftlich verstehen, was du tust. Die andere Möglichkeit ist, sich extrem tief zu spezialisieren und ein Superjurist oder ein Superprogrammierer zu werden. Ich sehe das bei meinem CTO. Er sagt ganz klar: Junior braucht er nicht mehr. Das ist bei uns in der Juristerei ähnlich. Wenn jemand für etwa vier Wochen braucht, was die KI in vierzig Minuten vorbereitet, dann wird es schwierig. Aber er hat durch diese Tools plötzlich drei zusätzliche Senior Developers an seiner Seite – und kann sie nutzen, weil er selbst

zwanzig Jahre Erfahrung hat. Die große Herausforderung für euch als Ausbildende ist aus meiner Sicht daher: Wie bringt man Menschen überhaupt in diese Jobs hinein, wenn alle sagen: Ich brauche dich noch nicht? Wie kommt man dann zu den zwanzig Jahren Erfahrung? Das war zwar nicht ganz die Frage, aber ich finde, das ist der eigentlich spannende Punkt.

Bernhard Wallisch: Ja, danke. Dann beleuchten wir das Thema einmal von der anderen Seite. Neben Ihrem Beruf bei KPMG, in dem Sie ja bereits KI einsetzen, haben Sie sich vor zwei Jahren trotzdem dazu entschieden, Informatik zu studieren. Warum haben Sie das gemacht? Was war Ihre Motivation? Und warum finden Sie, dass das heute immer noch wichtig ist? Was können Sie aus dem Studium für Ihren Beruf mitnehmen?

Arshia Reisi: Vor zwei Jahren muss man sagen: Agent Coding und alles, was damit zusammenhängt, war noch nicht so ein Hype wie heute. Ich habe nach der HTL zuerst ein Jahr gearbeitet und dann hier mit dem Studium angefangen. Erst danach kam dieser große Hype rund um Agent Coding – also die Vorstellung, dass man selbst keinen Code mehr schreiben muss, weil die KI ihn für uns schreibt. Aber man kann sehr gut erkennen, ob ein Programm komplett von einer AI geschrieben wurde oder ob jemand mit menschlicher Expertise noch einmal drüber geschaut und es verbessert hat. Das sieht man ziemlich schnell. Und genau deshalb studiere ich heute Informatik – und genau deshalb sollten es vielleicht auch andere in Zukunft noch tun. Es verändert sich natürlich vieles. Vielleicht schreiben wir in zehn Jahren tatsächlich weniger Zeilen Code selbst. Aber wir müssen auf diese Agenten aufpassen – oder generell auf dieses Gehirn, das an unserer Stelle denkt und schreibt. Wir müssen beurteilen können, ob das, was die KI macht, richtig ist, ob es Sinn ergibt und ob es so überhaupt umgesetzt werden sollte. Das merkt man oft schon bei der zweiten Iteration: Dann sieht man plötzlich, dass da zum Beispiel ein sehr kritischer Bug im Anmeldeprozess entstanden ist, den man beheben muss. Deshalb braucht es weiterhin menschliche Kontrolle. Und genau aus diesem Grund – damit ich mich in Zukunft besser mit der Materie auskenne und ein besserer Informatiker werde – habe ich mich für das Informatikstudium entschieden.

Bernhard Wallisch: Sehr interessant. Vielen Dank. Herr Kroisleitner, aus der Vorstellungsrunde habe ich mitgenommen, dass Sie aktuell auf der Suche nach Informatikerinnen und Informatikern sind. Wie sieht das konkret bei Ihnen im Unternehmen aus?

Albert Kroisleitner: Immer spannend, wenn ich bei dir sein darf – und wir sind ja alle per Du. Also von daher: Ich bin einfach Albert. Übrigens: Wir suchen aktuell tatsächlich Leute, die sich mit AI auskennen. Das Problem ist nur, dass es gar nicht so viele gibt, die sich wirklich gut damit auskennen, weil das Thema einfach noch sehr neu ist. Deshalb machen wir AI Internships. Wir gehen sehr konkret in das Thema hinein und lassen Leute einfach ausprobieren. Wir stellen relativ hohe Budgets nur für Tokens zur Verfügung – probiert Dinge aus, seid kreativ.

Mehr dazu vielleicht später. Wir haben auch noch ein paar Goodies mitgebracht – T-Shirts, Socken und so weiter. Was ich spannend fand bei den Vorträgen vorher: Ihr habt viel über Entwicklung gesprochen. Bei mir ist es ein bisschen anders, weil ich aus der Produktregel komme. Ich glaube, dass sich der gesamte Entwicklungsprozess massiv verändern wird. Also alles rund um Requirements, Specifications, Marktanforderungen – das wird viel schneller werden und viel näher zusammenrücken. Das klassische Modell – jemand schreibt ein Epic, jemand anders macht die Definition, ein Ingenieur überlegt sich etwas, der Architekt schaut drüber, dann geht es wieder zurück – dieses Ping-Pong-Spiel, bis alle das gleiche Verständnis haben. Das ist eigentlich nicht mehr da. Und das ist sogar gut so. Es wäre gut, wenn es in anderen Firmen auch nicht mehr so wäre, weil es einfach Speed aus dem System herausnimmt. Wir müssen uns überlegen: Wie bringen wir Speed ins System? Braucht man dafür noch Entwickler? Aus Produktsicht könnte man sagen: Ich schreibe einfach ins Tool, was ich brauche, und das Tool baut mir die Lösung.

Ganz so einfach ist es natürlich nicht. Ich kann relativ schnell Prototypen bauen. Aber irgendwann muss etwas produktionstauglich werden. Und wenn wir Kunden wie Hofer, Aldi, Edeka oder andere große Handelsketten haben und dort etwas nicht funktioniert, dann haben wir ein Problem – und die Kunden auch. Das muss einfach funktionieren. Es gibt also einen Sprung zwischen Prototyp und Production-Ready. Das bedeutet, dass sich das Jobprofil des Entwicklers verändert. Ein Entwickler muss verstehen:

- wie die Architektur aussieht
- welche Sicherheitsanforderungen es gibt
- worauf man in der IT-Security achten muss

Der Entwickler muss also verstehen, worum es geht, auch wenn ein Agent den Code schreibt. Er muss den Agenten richtig anleiten – sozusagen „in seiner Zelle programmieren“. Ein Beispiel: Wir hatten letztens eine Coding Session, zwei Tage lang – Donnerstag und Freitag, so eine Art Hackathon. Dabei ist ein Produkt entstanden, das tatsächlich funktioniert hat. Die erste Reaktion unseres CTO war: „Okay, ich schaue mir das mal an.“ Und sofort hat er gesehen:

- Passwörter sind nicht verschlüsselt
- hier ist eine Sicherheitslücke
- dort auch

Im Internet wäre das sofort ein Problem gewesen. Das heißt: Es braucht jemanden, der versteht, worum es geht, und diese Dinge fixen kann. Auf der anderen Seite: Wenn man heute einen Prototyp baut und danach das Engineering alles neu schreiben muss, verliert

man wieder Speed. Deshalb müssen wir überlegen, wie wir Dinge schneller in Produktion bringen können. Da können Tools wie Hooks oder Agent-Frameworks helfen. Aber trotzdem braucht man Entwickler, die verstehen, wie das Ganze funktioniert. Wir sind ein junges Unternehmen, aber selbst bei uns gibt es schon Spannungen: Auf der einen Seite Leute, die sagen: schneller, schneller, schneller. Auf der anderen Seite Leute, die sagen: Vorsicht, Qualität muss stimmen. Dieses Spannungsfeld haben wir ehrlich gesagt selbst noch nicht vollständig gelöst. Aber wenn du Entwickler bekommst, die mit diesen Tools umgehen können – und gleichzeitig Architektur verstehen – dann ist das extrem wertvoll. Wir haben vor kurzem einen Architekten bekommen, der viele Dinge im Silicon Valley ausprobiert hat. Der sagt dann einfach: „Okay, ich habe verstanden, was du willst. "Hier sind fünf Varianten." So jemanden brauchst du. Der klassische Entwickler, der einfach nur Code schreibt und langsam Requirements umsetzt – das wird es vielleicht noch geben. Aber in Bereichen wie unserem, wo Geschwindigkeit und Qualität gleichzeitig wichtig sind, wird sich dieses Profil stark verändern.

Bernhard Wallisch: Gut, danke. Die angesprochenen Tools werden wir später noch genauer anschauen. Bleiben wir vielleicht kurz bei der Rolle und den Kompetenzen des Informatikers. Wir haben gehört: Informatiker werden weiterhin gebraucht. Aber sie müssen sich vom reinen Coding weiterentwickeln. Ich würde gerne eine erste Runde Fragen aus dem Publikum zulassen. Gibt es jemanden, der eine Frage stellen möchte? Bitte.

Publikumsfrage: Vielleicht kurz zu meiner Person: Ich bin auch Alumni hier und arbeite bei einer Robotik-Firma in Wien, wo ich für Software verantwortlich bin. Außerdem habe ich noch einen zweiten Hut auf: Ich arbeite auch mit der Arbeiterkammer und beschäftige mich mit Fragen rund um die Arbeit der Zukunft. Ich habe bei uns im Unternehmen ebenfalls AI-Coding-Tools eingeführt und kann vieles unterstreichen, was hier gesagt wurde. Requirements Engineering ist extrem wichtig. Außerdem sehen wir, dass Agenten nicht alles alleine erledigen können – besonders dann nicht, wenn physische Systeme beteiligt sind. Wir testen zum Beispiel auf echten Robotern. Sobald Hardware ins Spiel kommt, braucht man Ingenieure mit Domänenwissen – sowohl in Robotik als auch in Software. Meine Frage geht eher in Richtung juristische Aspekte. Es gibt inzwischen viele neue Richtlinien der EU – etwa den AI Act oder die neue Maschinenrichtlinie. Wie kann man sicherstellen, dass man compliance-mäßig auf dem aktuellen Stand bleibt? Ein Beispiel ist die NIS2-Regulierung, die in Österreich noch nicht vollständig umgesetzt wurde. Was sind hier realistische Ansätze?

Christof Strasser: Wir beschäftigen uns tatsächlich viel damit. In meiner Kanzlei sind etwa 87 Prozent unserer Klienten Softwareunternehmen. Erstens glaube ich, dass hier noch sehr viel in Bewegung ist. Das ist noch lange nicht fertig reguliert – und wird sich noch stark verändern. Die EU ist auf legislativer Ebene gerade etwas in Panik darüber, was alles reguliert werden muss. Viele der bestehenden Regelungen sind noch relativ schwammig. Wir versuchen unseren Klienten daher vor allem, die Angst zu nehmen. Ähnlich war es mit der DSGVO, die 2018 gekommen ist. Viele Anwälte schreiben Bücher

darüber und machen Content-Marketing damit – aber in der Praxis ist das Thema oft weniger relevant, als man denkt. Von etwa 450 Startups, die wir betreut haben, gab es vielleicht ein oder zwei echte Datenschutzprobleme. Das waren sogar große Unternehmen mit vielen Endnutzerinnen und Endnutzern. Ich sage daher immer: Lasst euch von der Regulierung nicht zu sehr bremsen. Build first, clean up later. Natürlich muss man Compliance im Blick behalten – aber man sollte sich nicht von Angst lähmen lassen. Europa muss aufpassen, technologisch nicht komplett zurückzufallen. Das ist vielleicht eine ungewöhnliche Aussage von einem Anwalt. Aber ich würde sagen: Regulierung ernst nehmen – aber nicht überbewerten.

Bernhard Wallisch: Vielen Dank. Es gibt noch eine zweite Anmerkung dazu – bitte, Herr Essl.

Günter Essl: Aus der Perspektive der Berufsbildung habe ich eine Frage. Vielleicht liege ich falsch, aber ich habe den Eindruck, dass aktuell nicht mehr unbedingt die klassische Bachelor-Ausbildung gefragt ist, sondern eher Menschen mit Master-Niveau, die abstrakte Architekturen überblicken können. Unsere Bildungslogik ist traditionell so aufgebaut: Zuerst Grundlagen und Detailwissen im Bachelor, danach abstraktere Konzepte im Master. Aber vielleicht brauchen Unternehmen heute eher sofort diese Meta-Ebene. Denn viele typische Bachelor-Tätigkeiten – die früher der Einstieg in ein Unternehmen waren – könnten durch Automatisierung ersetzt werden. Die Architektur-Ebene bleibt dagegen bestehen. Gehen wir also möglicherweise mit unserer Ausbildungslogik in die falsche Richtung?

Svenja Schröder: Die Frage ist tatsächlich sehr komplex. Ich habe selbst an einer klassischen Universität studiert und dort auch mein Doktorat gemacht. Ich habe lange im sogenannten Elfenbeinturm gelebt. Irgendwann habe ich beschlossen, in die Wirtschaft zu wechseln – und das hat meine Perspektive komplett verändert. Was ich an Fachhochschulen sehr schätze, ist die starke Praxisorientierung. Absolventinnen und Absolventen kommen mit viel praktischer Erfahrung. Ich kann sie oft sofort einsetzen. Ich hatte mehrere Praktikanten und duale Studierende vom Technikum – und die waren wirklich großartig. Deshalb denke ich, man sollte überlegen: Was können wir Bachelorstudierenden schon früh mitgeben, damit sie im Unternehmen direkt arbeiten können?

Ein Beispiel: Prompting. Ich verlange von meinen Mitarbeitenden, dass sie exzellente Prompt-Engineers werden. Sie müssen verstehen, wie man Prompts formuliert, welche Tools man verwendet und wie man mit AI-Systemen arbeitet. Das könnte man eigentlich schon im ersten oder zweiten Semester lernen. Man könnte mit Tools experimentieren und praktische Projekte machen. Dann könnten Unternehmen sagen: „Diese Absolventen können mit AI-Agents arbeiten.“ Das wäre für uns extrem wertvoll.

Albert Kroisleitner: Ich kann das absolut unterstreichen. Wir haben gerade eine Produktmanager-Position besetzt. Wir geben Bewerbern immer kleine Tasks, um zu sehen, wie sie denken. Die Person, die den Job bekommen hat, hat nicht nur eine

Präsentation vorbereitet. Sie hat das Produkt analysiert, Requirements ausgearbeitet und zusätzlich eine kleine App gebaut, die genau das Problem löst. Das Ganze an zwei Abenden. So jemanden brauche ich. Das zeigt:

- schnelles Denken
- klare Requirements
- praktische Umsetzung

Und genau solche Fähigkeiten sollten Studierende früh lernen. Ich habe selbst einmal an einer Fachhochschule unterrichtet. Mir war wichtig, dass Studierende schon im ersten Semester Dinge ausprobieren. Zum Beispiel einfache Websites bauen – selbst wenn sie eigentlich Wirtschaft studieren. Dieser Moment, in dem jemand merkt: „Wow, ich kann das wirklich bauen." Das ist extrem wertvoll. Heute ist mit AI noch viel mehr möglich. Deshalb ist Learning by Doing aus meiner Sicht der entscheidende Punkt. Man muss früh anfangen, mit diesen Tools zu arbeiten.

Arshia Reisi: Bitte kurz: Es wäre aus meiner Sicht sehr interessant, wenn genau das, was Sie gesagt haben, stärker in die Lehre einfließt. Also, dass Leute, die bereits etwas gebaut oder „engineered" haben, zumindest ein Gespür dafür bekommen, was sie den Tools eigentlich sagen müssen. Bei den Projekten, die wir machen, kommen Professorinnen und Professoren oft mit einer Projektidee zu uns und sagen: „Schreibt bis zum fünften oder sechsten Semester ein Projekt dazu." Spannender wäre aus meiner Sicht aber, wenn Lehrende nicht mit einer fertigen Projektidee kommen, sondern mit einem konkreten Problem und sagen: „Löst das." Und dafür sollte man auch ausdrücklich KI nutzen dürfen – weil wir sie ohnehin verwenden werden. Es macht heutzutage nicht einmal mehr wirklich Sinn, alles per se von Hand zu schreiben. Sinnvoller wäre es vielleicht, wenn es in Projekten oder Vorlesungen zusätzliche Module zu AI Coding gäbe und Dozentinnen und Dozenten mit konkreten Problemen zu uns kommen und sagen: „Löst das mit AI." Oder man nimmt echte Industrieprobleme. Also man fragt Firmen: „Welche Probleme habt ihr gerade?" Und diese Probleme präsentiert man dann als Projekt. Dann könnte man sagen: Eine Gruppe löst das Problem mit AI, eine andere vielleicht manuell – und dann vergleicht man, was besser funktioniert hat, wie hoch der Zeitaufwand war und wie gut die Ergebnisse waren.

Bernhard Wallisch: Ja, da habe ich jetzt vor allem herausgehört: Probleme aus der Industrie, project-based learning – also genau eines der Themen, denen wir uns in der Lehre am Technikum sehr stark widmen. Es freut mich sehr über diese zweite Runde Fragen. Ich glaube, es gäbe sogar noch eine dritte Wortmeldung, aber ich würde Sie bitten, die weiteren Fragen später beim Buffet oder im Anschluss direkt an die Panelgäste zu richten. Ich würde jetzt gerne den zweiten Themenblock starten. Wir haben schon viel über Tooling gehört und darüber, was mit den unterschiedlichen Tools alles möglich ist.

Deshalb möchte ich jetzt konkret fragen: Was sind Ihre Erfahrungen, Learnings und Einschätzungen dazu? Was funktioniert mit welchen Tools? Wo liegen die Grenzen? Und welche Konzepte oder Ideen sollte man mitnehmen, wenn im nächsten Monat schon wieder die nächste Fülle an Tools auf uns zukommt?

Svenja Schröder: Ja, ich muss gestehen: Ich delegiere sehr viel an meine Mitarbeitenden. Deshalb nutze ich selbst gar nicht so viele Tools direkt. In der Beratung müssen wir natürlich immer schauen: Was ist beim Kunden überhaupt im Einsatz? Was ist Compliance-mäßig erlaubt? Übrigens – beim Thema Compliance muss ich ein bisschen widersprechen. Ich finde Compliance nämlich cool. Da steckt auch viel Beratungsleistung drinnen. Ich bin ein absoluter Compliance-Fan. Alle sollten Compliance lernen. Wirklich. Genauso wie Risikomanagement übrigens. Aber zurück zum Thema: Wir sind natürlich immer stark davon abhängig, was im jeweiligen Rahmen möglich ist. Wir können nicht einfach alle Tools nehmen und auf die Kunden loslassen. Wir sind kein Startup. Was aber trotzdem wichtig ist: dranbleiben. Und das ist etwas, das mir auch schon in der bisherigen Diskussion aufgefallen ist. Weil sich die Dinge so schnell ändern, muss man manchmal auch bewusst Dinge ausprobieren, die einem im ersten Moment ein bisschen absurd vorkommen – wo man sich denkt: „Das kann eigentlich nicht funktionieren." Denn genau diese Dinge, die heute noch komisch wirken, funktionieren oft in einem halben Jahr. Das heißt: Man muss ständig beobachten, wo gerade etwas entsteht, das vielleicht noch nicht ganz rund ist, aber Potenzial hat. Die Disruption ist im Moment wirklich massiv. Und ich glaube, was man machen muss, ist, sich aktiv Zeit zu nehmen und Dinge auszuprobieren.

Du hast vorhin Billability angesprochen – das ist im Beratungsgeschäft natürlich ein echtes Thema. Ich kann mich nicht jeden Tag zwei Stunden hinsetzen und mit Cloud Code herumprobieren. Ich muss auch für Kunden da sein. Aber ich glaube, man muss genau jetzt investieren, um nicht abgehängt zu werden. Ich schaue mir manche Dinge tatsächlich abends an, nach Feierabend, weil ich sonst gar nicht dazu komme. Die eigentliche Herausforderung ist also: Wie bleibt man dran? Wie nimmt man die größeren Konzepte mit, ohne jeden Tag zehn Stunden in neue Tools zu investieren? Denn man möchte ja auch noch schlafen, vielleicht Kinder haben, Sport machen, essen oder etwas trinken.

Ein letzter Punkt: Was ich auch sehr wichtig finde, ist Vernetzung. Wir versuchen intern gerade, eine Art Community of Practice aufzubauen. Also wirklich nach dem Prinzip: *show, don't tell.* Jemand sagt: „Hey, ich habe etwas ausprobiert, ich zeige euch das mal." Wir haben dafür einen AI Lunch eingeführt – kurze zwanzigminütige Sessions ohne großen Vorbereitungsaufwand. Einfach: „Ich habe das getestet, ich zeige euch jetzt mal, was dabei rausgekommen ist." Egal, ob es schon perfekt ist oder nicht. Ich glaube, man muss im Moment sehr nah dran bleiben, um anschlussfähig zu bleiben. Und jetzt gebe ich weiter.

Arshia Reisi: Was im Security-Bereich – also in meinem Beruf – super interessant ist: Firmen zahlen mich im Grunde dafür, dass ich ihr Unternehmen hacke, ohne dabei Alerts oder Sicherheitsmeldungen auszulösen. Und das ist heutzutage sehr, sehr schwer. Denn

Produkte von Anbietern wie CrowdStrike, Microsoft Sentinel und anderen arbeiten inzwischen selbst mit Deep Learning und Machine Learning im Hintergrund. Das heißt: Mit Techniken und Taktiken, die ich vor zwei Jahren noch verwendet habe, komme ich heute oft nicht mehr weiter. Was wir deshalb machen: Viele bekannte Tools haben Signaturen. Wenn diese Signatur erkannt wird, wird eine Sicherheitsmeldung ausgelöst. Also schreiben wir diese Tools teilweise um.

Das heißt konkret: Funktionsnamen werden verändert, Verhalten wird angepasst, Ausführungen werden verzögert – zum Beispiel mit eingebauten Wartezeiten. Dadurch kann man Verhalten und Signaturen relativ einfach verändern. Und damit kann man selbst große Security-Vendoren umgehen – obwohl Unternehmen Millionen für diese Produkte bezahlen. Das ist ein Bereich, in dem AI und Tooling sehr stark helfen. Ein zweiter Anwendungsfall, auf den ich bei uns besonders stolz bin, ist eine Triage-Pipeline für Forensic-Cases. Wenn ein PC forensisch untersucht wird, sind die Triage-Daten immer ähnlich aufgebaut. Früher mussten wir Timelines parsen, Informationen extrahieren, in Monitoring-Tools importieren und manuell auswerten. Was wir jetzt gebaut haben, ist eine interne Lösung – wir dürfen keine externe verwenden –, bei der die Informationen automatisiert verarbeitet und an ein internes AI-System übergeben werden. Als Analyst bekomme ich dann einen Bericht mit einem ersten Überblick darüber, was passiert ist. Ich sehe also sofort, worauf ich schauen muss. Der Rest ist ein Dashboard und die vertiefte manuelle Analyse. Also ja: Tooling, Cloud Code und ähnliche Ansätze sind extrem stark, wenn es um Automatisierung geht.

Albert Kroisleitner: Ja, natürlich gibt es da sehr viel. Wir hatten letztens intern bei Fiskaly auch Diskussionen darüber, welche Plattformen und Methoden sinnvoll sind. Ein Thema war zum Beispiel auch Jailbreaking – also wie man Systeme dazu bringen kann, Informationen zu liefern, die sie normalerweise nicht herausgeben würden. Das machen wir natürlich nicht produktiv, aber es ist wichtig, solche Mechanismen zu verstehen, um die eigenen Produkte sicherer zu machen. Wir verkaufen am Ende Compliance – das heißt, unser Produkt muss sicher sein.

Aber zurück zum Tooling: Ich glaube, das wurde heute schon mehrfach gesagt: Es gibt unglaublich viele Tools, und jeder hat irgendwo seine Präferenzen. Der eine arbeitet eher mit Automatisierungsplattformen wie n8n, Make.com oder ähnlichen Tools. Am Ende ist das ein bisschen wie bei Programmiersprachen oder Entwicklungsumgebungen: Jeder hat seine Umgebung, aber die Konzepte dahinter sollten dieselben bleiben. Und genau das ist aus meiner Sicht der zentrale Punkt: Nicht das einzelne Tool ist entscheidend, sondern das Verständnis der Konzepte im Hintergrund. Wir müssen überlegen, wie wir wegkommen von riesigen Mengen an Dokumentation und Requirements, die irgendwo abgelegt werden, wo dann irgendjemand sie wieder heraussuchen und interpretieren muss – während der wichtigste Akteur, nämlich der Agent, gar keinen Zugriff darauf hat. Es geht also darum, diese Informationen so bereitzustellen, dass Agents direkt damit arbeiten können. Das bedeutet auch, dass man Abläufe ändern muss, um Multi-Agent-Systems sinnvoll aufzubauen. Dafür gibt es wiederum viele Tools.

Aber entscheidend bleibt: Die Arbeitsweise muss angepasst werden, damit man den Vorteil dieser neuen Möglichkeiten wirklich nutzen kann. Egal, welches Modell im Hintergrund läuft – ich habe inzwischen oft die Erfahrung gemacht, dass ein Modell für den einen Anwendungsfall besser ist und ein anderes für einen anderen Kontext. Deshalb investieren wir als Firma auch stark in dieses Thema. Wir haben bewusst ein recht hohes Budget dafür. Und ganz ehrlich: Wenn wir dieses Budget nicht ausgeben würden, würden wir eher dafür kritisiert werden. Also: Geld ausgeben, lernen und ausprobieren. Wir machen Hackathons, wir haben Austauschformate, wir zeigen einander Projekte. Das ist extrem wichtig. Denn auch wenn viele Leute jung sind, heißt das nicht automatisch, dass sie auch risikofreudig genug sind, sich wirklich auf diese Themen einzulassen. Manche sind absolute Frontrunner, probieren Dinge aus, bauen etwas, verwerfen es wieder und machen weiter. Andere sind stärker auf Qualität bedacht. Ich glaube, am Ende braucht es beides. Aber entscheidend ist, ein Forum zu schaffen, in dem dieses Wissen weitergegeben wird.

Denn eines ist für mich ganz klar: Wenn ich in einem oder zwei Jahren noch genauso arbeite wie heute, dann habe ich draußen vermutlich keinen Job mehr. Und wenn unsere Firma so weitermachen würde wie bisher, hätten wir ebenfalls ein Problem. Dann würden auch wir irgendwann disrupted werden. Wir sehen heute schon Märkte, in denen kleine Startups sehr schnelle Services anbieten, die ähnlich gut funktionieren wie unsere. Vielleicht nicht in stark regulierten oder zertifizierungspflichtigen Bereichen – aber überall dort, wo Anforderungen klar definierbar sind, gilt: Du gibst das Problem ins Tool und bekommst eine Lösung zurück. Speed is king. Also raus in den Markt – und dann dort weiter verbessern. Natürlich: Wenn ein Produkt zertifiziert werden muss oder klare Compliance-Anforderungen erfüllen muss, dann muss es das auch. Aber alles, was darüber hinausgeht – zusätzliche Features, Verbesserungen – kann man sehr schnell nachschieben. Und genau deshalb sage ich: Schneller werden. Wenn ich heute über ein neues Produkt nachdenke, dann sollte es bei uns idealerweise in drei Monaten draußen sein. Einfach Gas geben und weitermachen. Und selbst in diesen drei Monaten berücksichtigen wir natürlich bereits Compliance und Zertifizierung. Aber trotzdem: Wir müssen schneller werden, weil andere schon draußen sind. Und das ist aus meiner Sicht der entscheidende Punkt: Egal welches Tool man verwendet – man muss die Konzepte verstehen und dann ins Tun kommen.

Bernhard Wallisch: Ja, danke sehr. Apropos schneller werden: Ich muss ganz kurz auf die Zeit schauen. Ich glaube, eine oder zwei Fragen aus dem Publikum gehen noch aus, bevor wir zur Abschlussrunde kommen.

Publikumsfrage: Hallo, schönen guten Abend. Ich bin ebenfalls Alumni. Ich würde dieses Thema „schneller werden“ gerne noch einmal aufgreifen, weil es natürlich auch bei uns im Unternehmen relevant ist – und ich glaube auch in der Lehre. Wir reden hier von Technologien, die noch nicht sehr alt sind und eine Geschwindigkeit an den Tag legen, die ich bisher in dieser Form bisher nicht erlebt habe. Ich habe im Internet schon vieles miterlebt, aber diese Geschwindigkeit ist wirklich enorm. Und gerade deshalb frage ich

mich: Wie will man das in die Lehre hinbekommen? Und wie will man es auch in Unternehmen umsetzen, wenn man Technologien wie Codex hat, die gerade erst im Februar herausgekommen sind und de facto einen Junior-Programmierer ersetzen? Das ist einfach so. Aus unternehmerischer Sicht verstehe ich das total: Ich gebe meine Sachen hinein, lasse sie erledigen und bin schneller am Markt. Da bin ich voll bei Ihnen. Aber gleichzeitig machen wir damit den Software-Entwicklungsbereich in gewisser Weise auch kaputt. Für die Menschen, die in diesem Bereich arbeiten, sehe ich momentan sehr schwierig, wie da noch etwas nachkommen soll.

Albert Kroisleitner: Wer von euch hat noch einen Kassettenrekorder zu Hause? Und wer streamt Musik? Eben. Man kann das natürlich gesellschaftlich noch größer aufziehen und fragen: Europa gegen USA gegen China – wer hat welche Geschwindigkeit, wo sitzt die Technologie, wer entwickelt gerade was? Wir haben vorher kurz über den österreichischen Kollegen gesprochen, der jetzt bei OpenAI angeheuert hat – vermutlich zu einem sehr guten Gehalt, hoffe ich zumindest für ihn. Dieser Brain Drain ist schon ein Problem. Aber auf der anderen Seite ist Fortschritt eben etwas, das passiert – und im Moment passiert es extrem schnell. Ich bin kein Hellseher, aber wenn ich mir anschaue, was gerade alles kommt, dann ist schon sehr klar, dass sich vieles verändern wird. Ich frage mich zum Beispiel: Brauchen wir in zwei Jahren überhaupt noch Apps? Vielleicht sage ich einfach meinem Chatbot: „Buche mir einen Flug von A nach B“, und das war’s. Dann brauche ich keine App mehr. Und das kann man noch viel weiterdenken. Diese Dinge werden kommen. Die Frage ist nur: Wo spielen wir da mit? Und wo spielt unsere Firma mit? Wenn ich zurückdenke – ich habe einmal Feinwerktechnik studiert, das ist schon sehr lange her – da haben wir noch Laufwerke für Kassettenrekorder konstruiert, mit Zahnrädern, Übersetzungen, technischer Optik und all diesen Dingen. War spannend – aber brauche ich das heute noch? Nein. Das Ding ist weg. Und ähnlich sehe ich das auch mit dem klassischen Programmiererbild. Das Wissen bleibt wichtig, um die Konzepte zu verstehen, da bin ich voll bei dir. Ich habe selbst programmieren gelernt, vor langer Zeit angefangen mit Visual Basic. Das mache ich heute natürlich nicht mehr. Heute lasse ich mir Dinge in Python oder anderen Sprachen generieren. Dann gehe ich drüber und prüfe grob, ob das ungefähr das macht, was ich will. Ich nehme mir also die Zeit, das Ergebnis zu kontrollieren. Ich sage: Das ist meine Zielsetzung – kann das Tool das umsetzen, ja oder nein? Und wenn nicht, dann brauche ich Leute, die sich besser auskennen als ich. Aber Dinge verändern sich. Und ich glaube nicht, dass man sich dagegen stellen sollte. Das wäre keine gute Idee, denn es wird trotzdem passieren. Die Frage ist eher: Wie nehmen wir dieses Tempo mit?

Und da sind wir in Europa aus meiner Sicht immer noch langsamer als das, was wir aus dem Silicon Valley sehen. Wenn Kolleginnen und Kollegen von dort zu uns kommen und zeigen, was aktuell schon möglich ist, dann sitzt man oft da und denkt sich: Wow. Genau solche Leute und solche Perspektiven müssen wir auch stärker in die Lehre holen, damit wir ein Gefühl dafür bekommen, was da draußen schon möglich ist – und damit wir versuchen, dieses Tempo zumindest teilweise mitzunehmen.

Svenja Schröder: Dass man das Gefühl hat, hier würde etwas kaputtgehen, spricht eigentlich genau für Disruption. Genau das ist ja der Punkt: Man denkt, alles geht kaputt – aber in Wirklichkeit entsteht etwas Neues. Vielleicht noch einmal aus der Perspektive des Projektgeschäfts: Informatikprojekte verändern sich gerade massiv. Man muss sich wirklich überlegen, wie man Teams aufgestellt, um in der aktuellen Wirtschaftslage konkurrenzfähig zu bleiben. Wenn wir heute ein Informatikprojekt anbieten, können wir uns das gar nicht leisten, nur mit Fachkräften aus Österreich zu arbeiten. Man muss sich sehr genau überlegen: Wer übernimmt welche Aufgabe? Welche Technik wird wofür eingesetzt? Und was ergibt am meisten Sinn? Man könnte zum Beispiel sagen: Ein sehr guter Product Owner, ein Requirements Engineer oder Produktmanager, dazu ein sehr guter Architekt – und dann mehrere wirklich gute Agents. Das wäre zum Beispiel ein Team, das schnell etwas auf die Beine stellen kann, das auch Bestand hat. Oder nehmen wir Nearshoring und Offshoring: Wir haben uns jahrelang gefragt, wie wir Nearshoring- oder Offshoring-Teams für österreichische Kunden im Behördenbereich sinnvoll einsetzen können. Und heute gibt es zum Beispiel DeepL. Da ist das auf einmal gar nicht mehr so eine große Frage. Man muss natürlich in guter Requirements-Engineering-Manier ein Glossar bereitstellen, damit Fachbegriffe richtig übersetzt werden – zum Beispiel ins Englische. Aber dann funktioniert das. Und das ist eigentlich noch gar nicht so lange her, dass ich mir darüber große Sorgen gemacht habe. Jetzt gebe ich weiter – wir haben nicht mehr so viel Zeit, damit Christof noch etwas über Indien sagen kann.

Christof Strasser: Ich knüpfe da nur direkt an: Ich lasse inzwischen deutschsprachige Verträge von Indern entwerfen, auf Englisch übersetzen, dann wieder auf Deutsch zurückführen, gehe einmal drüber – und damit ist etwas erledigt, wofür früher nur ein österreichisches Team zuständig gewesen wäre und das dort vielleicht drei Wochen später geliefert worden wäre.

Bernhard Wallisch: Ja, danke. Ich habe auch gehört: Unsere Zeit ist um. Es war eine spannende und sehr interessante Diskussion. Wir haben in Österreich begonnen – beim Alumni Talk – und sind dann sehr schnell international geworden, bis hin zur Kontinentsicht und nach Indien. Ich freue mich, dass Sie alle dabei waren. Es wird nachher beim Buffet noch Gelegenheit zur weiteren Diskussion geben. Damit bedanke ich mich bei allen und übergebe mich wieder an Raphael.

Rafael Rasinger: Danke, Bernhard Wallisch, danke. Es waren ja noch Fragen aus dem Publikum offen. Bitte nutzen Sie dafür nachher die Zeit bei den Getränken und beim Buffet, um diese bilateral zu stellen – so, wie Bernhard es schon gesagt hat. Mir bleibt jetzt vor allem, allen Speakern zu danken, die heute gekommen sind. Thank you, Alex. Thank you to all panelists for joining us. Danke, Günther.<Und natürlich noch ein paar Ankündigungen, wie es weitergeht:<Als Absolventinnen und Absolventen seid ihr herzlich willkommen in der Alumni Technikum Community. Bitte meldet euch für den Newsletter an. Dort bekommt ihr einmal im Monat Informationen darüber, wann die nächsten Alumni Talks und Startup-Formate stattfinden. Wir haben auch einen Startup-Newsletter für alle,

die sich für Gründen und für Entrepreneurship interessieren. Auch dazu eine herzliche Einladung, sich anzumelden.

Die nächste Ankündigung: Unser nächstes Event ist für Studierende wie auch für Firmen gleichermaßen interessant. Wir haben die Karrierelounge am 15. und 16. April. Susanne Siegel ist dafür die Ansprechpartnerin. Firmen können sich anmelden und Partner werden – beziehungsweise sich bis zum angegebenen Termin registrieren. Und natürlich sind auch Absolventinnen und Absolventen, die einen neuen Job suchen, ebenso wie frisch Graduierte herzlich eingeladen, dort Arbeitgeber und Firmenpartner der FH Technikum Wien kennenzulernen.

Ende Mai kommt dann der nächste Startup Monday zum Thema Expo Austria. Die Weltausstellung findet nächstes Jahr, also 2027, in Belgrad in Serbien statt, wo auch Österreich über drei Monate präsent sein wird. Dafür suchen wir Innovationen, neue Startups und Lösungen, die dort ausgestellt werden können.

Mir bleibt nur noch zu sagen: Danke fürs Kommen. Thank you for joining us. Und bis zum nächsten Alumni-Talk, Startup Monday oder zur nächsten Karrierelounge. Einen schönen Abend, gerne noch Erfrischungen hinten mitnehmen und die Zeit zum Austausch nutzen. Die Speaker sind noch ein bisschen da. Schönen Abend und bis zum nächsten Mal. Danke.