

KONTAKT

contact@andreaswagner.io
+372 5870 8341
andreaswagner.io

ECKDATEN

Tallinn, Estland · EET (UTC+2)
Remote, vor Ort nach Bedarf
Verfügbar für neue Projekte
Freiberuflich über Whyservices OÜ oder Teilanstellung, EU-weit

KERNKOMPETENZEN

Ruby on Rails täglich seit 2017	●●●●○
Ruby Anwendungen, Skripte, Runbooks	●●●●○
PostgreSQL	●●●○○
Sidekiq & Redis	●●●○○
MySQL	●●●○○
Docker	●●●●○
CI/CD-Pipelines GitHub Actions, GitLab CI, Woodpecker	●●●●○
Linux-Server	●●●●○
Ansible	●●●●○
Git	●●●○○

●●●●○ Fortgeschritten ●●●○○ Sicher

ARBEITSWEISE

Rails-Konventionen · Schrittweise Upgrades · CI/CD · Infrastructure as Code · Schriftliche Systemanalyse

BRANCHEN

Fintech · Hosting & IT-Dienstleistung · Museen & Kultur · Agenturen & KMU

SPRACHEN

Deutsch (Muttersprache)
Englisch (C1)

AUSBILDUNG

M.A. European Urban Cultures
VUB Brüssel, Tilburg, MMU Manchester, EKA Tallinn, 2011
B.A. Regionalstudien Asien/Afrika
Humboldt-Universität zu Berlin, 2010

Andreas Wagner

Ruby-on-Rails-Entwickler

Ich übernehme Rails-Anwendungen, deren ursprüngliche Entwickler nicht mehr verfügbar sind. Ich halte sie im Produktivbetrieb stabil und mache Änderungen wieder risikoarm.

9 Jahre
freiberufliche Rails-Entwicklung, seit 2017

13 Jahre
Betrieb produktiver Systeme, von Fintech bis Museum

4
Rails-Anwendungen unter Wartung

KURZPROFIL

Ruby-on-Rails-Entwickler mit neun Jahren Praxis. Umfangreiche Erfahrung in Systemadministration. Ich arbeite an bestehenden und neuen Anwendungen. Ich bevorzuge schlichtes, konventionelles Rails gegenüber cleverem Code, denn Konvention steht auch bei Stil über Konfiguration. Weil ich die Server selbst betreibe, weiß ich, was eine Änderung in Produktion bedeutet: Deploys, Hintergrundjobs, Backups. Ich arbeite remote aus Tallinn, direkt mit Entscheidern und Fachbereichen.

PROJEKTERFAHRUNG

Übernahme und Modernisierung der Wettbewerbsplattform Defolio.com

2017 – heute

Freiberuflich über Whyservices OÜ · Tallinn, remote

Branche: Marketing · Rolle: Rails-Entwickler & Maintainer · Team: allein, mit dem Product Owner des Kunden

Die Anwendung war produktiv im Einsatz, aber Datenbank-Queries wurden mit jedem Wettbewerb länger. Framework-Upgrades wurden eingespielt, aber Syntax und Code-Stil drifteten immer weiter auseinander. Es gab keine funktionalen Tests. Framework und Betriebssystem waren inhärent miteinander verwoben. Ich erstellte zuerst eine Tiefenanalyse: Code und Infrastruktur lesen, danach ein schriftlicher Bericht mit Risiken nach Schwere sortiert. Bevor ich etwas änderte, sicherte ich die geschäftskritischen Abläufe mit Tests ab. Dann vollendete ich die Ruby- und Rails-Upgrades (Ersatz für veraltete und nicht funktionale Syntax) und ersetzte dabei Eigenbau-Lösungen durch Rails-Konventionen. Deploys laufen danach über eine CI-Pipeline mit Staging und Rollback, und für Backups gibt es ein Wiederherstellungsverfahren, das tatsächlich geprobt ist. Die Anwendung läuft wieder auf einer unterstützten Version (Rails 7 → 8.1), N+1-Queries konnte ich fast vollständig entschärfen. Deploys dauern 4 Minuten aus Git statt manuell 30 Minuten. Der Kunde behielt seine Anwendung, statt einen auf 45.000 € geschätzten Neubau zu bezahlen.

Ruby on Rails · PostgreSQL · Sidekiq · Redis · GitHub Actions · Docker

PROJEKTERFAHRUNG (Fortsetzung)

Backoffice-Anwendung für ein Hosting-Geschäft

2015 - heute

Whyservices OÜ, eigenes Produkt (WellPress / WellHost)

Branche: Hosting & IT-Dienstleistung · Rolle: Product Owner & alleiniger Entwickler · Team: allein

Managed Hosting für viele kleine Kunden hieß: Rechnungen, Kundenkommunikation, Statusmeldungen und die Hosting-Historie jeder Website lagen in getrennten Werkzeugen. Jeder neue Kunde brachte mehr Handarbeit und das Risiko einer vergessenen Rechnung oder eines nicht angekündigten Wartungsfensters. Ich habe eine Rails-Backoffice-Anwendung entworfen und gebaut, die das zusammenführt: Rechnungsstellung, Kundenkommunikation, öffentliche Statusseiten und eine Hosting-Historie pro Website. Sie spricht die APIs der Hosting-Anbieter an und verwaltet die Administrations-Runbooks. Ich habe bewusst schlichtes Rails verwendet, damit die Wartung für ein Ein-Personen-Unternehmen günstig bleibt. Eine Person betreibt 30 Kunden-Websites und -Anwendungen. Ein neues Projekt anzulegen dauert 30 Minuten statt 8 Stunden. Rechnungen und Statuskommunikation sind automatisiert.

Ruby on Rails · Ruby · Ansible · REST-APIs · e-Rechnung

Neue Rails-Anwendung gemeinsam mit dem Fachbereich

2025-2026

MTÜ Ruumiharidus · über Whyservices OÜ

Branche: Bildung · Rolle: Rails-Entwickler, von der Anforderung bis zum Betrieb · Team: allein, mit zwei Ansprechpartnern im Fachbereich

Der Verein wickelte seine Kernprozesse von Hand in Tabellen und E-Mails ab. Ein WordPress-Plugin verwaltete Schüler, Gruppen und Stundenpläne. Doppelte Erfassung und Fehler kosteten das Team 16 Stunden pro Woche, und einen verlässlichen Überblick über Softwarefunktionen hatte niemand. Ich habe die Anforderungen direkt mit dem Fachbereich erarbeitet: kurze Termine, schriftliche Spezifikationen und kleine Releases, die früh ausprobiert werden konnten. Gebaut habe ich die Anwendung als konventionellen Rails-Monolithen. Hosting, CI und Monitoring habe ich ebenfalls eingerichtet, sodass der Kunde vom ersten Gespräch bis zum Betrieb einen Ansprechpartner hatte. Nach 12 Wochen live, 2 Personen arbeiten täglich damit. Der manuelle Aufwand sank um 2 Drittel.

Ruby on Rails · PostgreSQL · Sidekiq · CI/CD

Head of System Operations

2013 - 2017

Cash on Go Ltd · Tallinn

Branche: Fintech, reguliert (PCI DSS, FCA) · Rolle: Leiter Systembetrieb · Team: 3 Systemadministratoren

Hauptanwendung und Datenbank des Unternehmens liefen auf von Hand administrierten Servern im Rechenzentrum und im Büro. Ein Hardwaredefekt hätte das Geschäft gestoppt, und Audits verlangten dokumentierte, wiederholbare Prozesse. Ich habe das Team der Systemadministratoren geleitet und mit der Entwicklung Release- und Deployment-Richtlinien vereinbart. Wir sind von Handarbeit auf der Shell zu Infrastructure as Code gewechselt, zuerst mit Chef (Ruby), später mit Ansible. Gemeinsam mit den Entwicklern haben wir die PHP/Symfony-Anwendung und ihre MySQL-Datenbank auf eine redundante Umgebung an mehreren Standorten in AWS migriert. Der Single Point of Failure war beseitigt, Releases folgten einem dokumentierten Lebenszyklus. Als Hauptverantwortlicher für die PCI-DSS-Prozesse habe ich das Unternehmen durch 3 Audits ohne wesentliche Feststellungen geführt. Ausgezeichnet mit dem internen Einstein Award 2014 für abteilungsübergreifende Zusammenarbeit.

Chef · Ansible · AWS · MySQL · PHP/Symfony

FRÜHERE STATIONEN

2016 - 2024	Ausstellungs-IT für estnische Museen und EU2017 (über Whyservices OÜ): Windows-Terminals mit Ansible verwaltet, Deploy-Pipelines für bis zu 50 Geräte pro Ausstellung.
2008 - 2012	Inhaber, Whyservices, Berlin: Webentwicklung, Hosting, Webdesign und Fernadministration für kleine Kunden.
2009 - 2010	Studentische Hilfskraft IT-Administration, Humboldt-Universität zu Berlin: Benutzer, Netzwerk und Active Directory.