Hi,

I am Ryan.

Catholic. Husband to beautiful wife. At Wiom, I build the internet architecture needed to bring unlimited internet to 500 million people in India. By designing, developing, and deploying advanced AI and Machine Learning models.

Long story, short - I put my head down to do GOD's work, and crazy GOOD things happen. Glory to GOD and JESUS, who is Lord.

Amen!

==============
MY FOOTSTEPS
==============

By 33

Early 2025, I started a slow painful marathon, to brush up the basics of high school mathematics - as a prelude to diving into some serious deep tech in the years to come. Created md2ltx along the way [pip install md2ltx; See: https://pypi.org/project/md2ltx/]. A command-line tool for converting Markdown to LateX-formatted PDF via Pandoc, that also allows you to write executable python code within your markdown. Over time, I kind of realised that I understand abstract concepts better if I just build stuff, rather than engage in hypothetical-BS-mental-gymnastics. Nevertheless, md2ltx could still be useful in documenting mathematical modelling exercises. Also, created GitGuru [GitGuru](https://github.com/ryangerardwilson/gitguru) to help define a uniform git workflow for complex tech projects, and DataSling [pip install datasling; See: [Datasling](https://pypi.org/project/datasling/) to allow me to use vim and python as a slick data analytics interface.

Married beautiful wife on March 1, 2025.

Found myself down a rabbit hole testing out and comparing architecture using pure functional approaches versus OOP paradigms. OCaml. Haskell. And the OOP-rich PHP. These experiments helped understand more coherently the problem OOP was built to solve, and why Functional Programming (though great for libraries) is simply not good enough to tackle hard real world problems. Realised using exclusively Functional Programming for library development, when combined with using exclusively OOP for core component engineering - can be a lethal combo. Built DeathStarPilot a to do list app, where you can destroy your completed tasks as if you were Darth Vader destroying planets. The objective was, to find a fun way to implement OOP techniques - and few things are more pleasurable than OOP + ThreeJS.

Changed my mind on the above propositions, when I realized that functional and procedural programming are in fact a form of OOP. The API interaction patterns in a project define OOP, not the language syntax.

Doubled down on using Vim (not NeoVim) as my IDE. Built a dozen of symbolic-linked python scripts, to replicate the few IDE features I actually use. Some include - peeingtom.py (to do a simple CONTAINS search in my project directory), copycat.py (to copy to clipboard my entire project files, so I can paste into Grok), skeletor.py (to print a ASCII-art style tree of my project directory) ... and yes, a deploy-php-app.py (to deploy my php project to a domain name on my VDS).

By February 2025, I rebuilt my personal website (this very one), in vanilla PHP. Quit CSS frameworks in favour of a simple well-organised BEM-styled main.css. And by July, I re-wrote it, again - coalescing all my personal web server and bot automations (deleting spam emails, etc), into a single Docker container. Realised, that as much as I agree with Linus Trovalds' critique of Docker, it is a necessary 'evil' given the chaotic evolution of the internet. Also, pivoted back to Python. Have a strong hunch, that as AI gets better and better, Python will be the only form of logical abstraction any engineer needs. Also, quit React in favor of a simple main.js, orchestrating hand-crafted ES6 modules.

Ironically, I changed my mind, again, and, ditched the above described website in favor of a simply nginx configuration that hosts pdf files at the `/` and `/resume` routes, which I push to my vpn using a simply update-website.py script which opens the writeups in vim, and generates, then uploads the pdfs of those write ups to the relevant nginx directory. 200 lines of code > 2000. Updated conclusion: Docker is, at large, unnecessary if you are not writing enterprise stuff.

Rebuilt Wiom's core matchmaking algorithm (simplifying both the code, and the Product design). 9 months of pain. Closet I will come to experience labour. Genie determines if a new lead is serviceable, and if so, which Partners should be notified of that lead, and when. First experience of deploying a operations-critical data science pipeline that impacts the entire platform dynamic of Wiom. Sharpened my ability to abstract and express complex ideas with elegance in Python.

In August 2025, fell in love with touch typing thanks to gtypist - one of the most underrated cli productivity tools of all time. Fell deeper in love with vim thanks to vimtutor and the VimBook. Explored NeoVim seriously, while on vacation, but ditched it, when I realized that vanilla vim is simply the superior product in every sense.

In September 2025, built vimtutor-advanced to solve this problem for myself: what rituals must I practice to use my computer as a samurai does his sword. This expands the classic vimtutor cli, covering advanced vim concepts set out in Steve Oualline's 2001 VimBook. The tutor also expands the idea of using a vim-based muscle memory to window tiling and browser based navigation, to maximize productivity by building muscle memory with the right conventions. At one point, the tutor also covered NeoVim, however, I removed NeoVim tutorials after concluding that the origianl vim is simply the superior product. See: [vimtutor- advanced](https://github.com/ryangerardwilson/vimtutor-advanced)

By October built pytutor, ctutor, atutor to solve this problem for myself: given that AI vastly improves my ability to access 'low quality code', what rituals must I practice to avoid AI slop in my projects in this era of 'vibe-coding'. These apps are systems to keep coding muscle memory sharp by practicing writing small snippets of code as a daily ritual, in python, c lang, and assembly, respectively. See:
[pytutor](https://github.com/ryangerardwilson/pytutor),
[ctutor](https://github.com/ryangerardwilson/ctutor), and
[atutor](https://github.com/ryangerardwilson/atutor).

Mid-October I realized that I could combine all my tutor apps into a single rtutor app with a gtypist-like interface, and feature a progression of tutor courses organized as courses -> parts -> lessons. This way, I can scale the detail of the lessons as per the competency of the user. See:
[rtutor](https://github.com/ryangerardwilson/rtutor).

By 32

Doubled down on my technical proficiency in RUST, expanding rgwml from version 0.1.0 to version 1.1.71 across 98 version updates. I also built CSVBRO, a Microsoft Excel replacement leveraging the rgwml library to streamline complex AI/ML/Data Analysis workflows via the GNU-Linux terminal, minimising cognitive overload.

2024 was the year of RUST. And boy, I did not miss out. Battle tested the RGWML library from version 0.0.1 to 1.3.81 [cargo add rgwml; See: https://crates.io/crates/rgwml], encompassing over 250 version updates. Downloaded over 2,00,000 times, RGWML evolved to become a slick RUST-dominant AI, Data Science, & Machine Learning Library designed to minimize developer cognitive load and replicate the Python Pandas Library with integrations for OpenAI, XGBoost, and data clustering techniques. Was saturated AF by RUST's verbose syntax within 6 months, craving the simplicity of Python and Javascript. Okay, it was a good exploratory project - but why re-invent the wheel?

Pivoted to a simpler tech stack, and experimented with the idea of "building in production" with Python, Javascript, and SQLite. Rewrote my entire RGWML rust library in Python. Then, re-wrote it again in Python as RGWFUNCS [pip install rgwfuncs; See: https://pypi.org/project/rgwfuncs/], with better adherence to SOLID principles. Consequently, re-wrote CSVBRO in Python [pip install csvbro; See: https://pypi.org/project/csvbro/]. Developed a deep hate for frameworks. Put these ideas to the test by using it to build a Salesforce substitute for Wiom's internal teams by coding directly into a VDS server. Had insane fun.

By 31

Taking charge of Wiom Labs, an innovation-focused offshoot from Wiom's baseline product design architecture, I created HAPP.AI—a lean language model leveraging RUST concurrency to mimic the output of OpenAI’s GPT-4-turbo. This involved managing transformer/model states with a MYSQL database, all based on training data organized in Google Sheets. Additionally, I built a free and open-source library to simplify operations in data science, machine learning, and AI, prioritizing CLI tools over GUIs.

28-30

Entrepreneurial journey was more like a rollercoaster ride! I started a LegalTech hustle, launched three productivity apps—SLTYE, LAVENDORB, and EQUITY'S DARLINGS—primarily for my own business needs but marketed them as products as well. I burnt cash (lots and lots of cash), and my savings dwindled to zero. The venture became cashflow dead and ultimately crashed.

Met Satyam Darmora by happenstance and joined Wiom, a tribe inspired by infinity, bringing unlimited internet to 500 million lower-middle-class homes.

Created HAPPY, Wiom's internal CX CRM; developed DREAM, an experimental social media concept; GURU, an OpenAI-powered chatbot; TERMINAL LOVE, an open and free MVC architecture leveraging AI as a Controller. I became multilingual—fluent in Bash, PHP, Python, and JavaScript. I also reached an intermediate level of classical piano proficiency and discovered a love for Gospel Piano (and the Gospel, of course!). I spent time with people smarter than me and exponentially grew as a result.

25-27

COVID-19 outbreak ruined my plan to start my own law practice.

Persisted working with law firms. Ouch. Worked as a lawyer long enough to know that LegalTech is an oxymoron. I learned HTML, CSS, and PHP (because that's what Zuck used to build Facebook’s first iteration). I became awesome—pretty awesome—at PHP, and revived my piano game.

At 24

Started working as an IP and tech attorney.

At 23

Joined a LegalTech startup, which tanked in 6 months.

At 18

Joined RMLNLU—a law school in Lucknow.

By 17

Dropped out of commerce. Or whatever DU taught as "commerce".

At 15

Dropped out of science. Or what CBSE taught as "science."

Till 15

Dreamed of becoming a mad scientist (more Mandark, less Dexter), playing for Liverpool, and doing fancy stuff on the piano.

Birth

And so it began.