Huy Le CV

SOFTWARE ENGINEER · LEADER

(+84) 983-543-444 huy.lenq@gmail.com huylenq.com huylenq huylenq HCM City, Vietnam

Summary

Having 12 years of experience on a widespread of software development spectrum taught me into being a generalist with a broad, versatile and adaptive mindset. This gives me compound-value perspectives and a top-down mindset when venturing on any domain verticals or horizontal layers of software systems.

But it is meaningless for one to say being a jack-of-all-trades (an equivalent cliché of saying "I am a quick learner", and it does cost the niche and specialized eyes into a particular field or stack), here are my recent focused hard skills over the past few years:

System Architecture	Particularly on data systems and/or heavily distributed ones. While this is a discipline few could claim expertise in, I do have enough of broad understanding to able to devise a grand approach and design, making choices on and implement the components (choices of storages, message queues, load balancing & caching strategies, APIs & RPCs design and contracts, etc.)
Databases	While I didn't write any piece inside a database engine, I grasped a well bit of the their inner workings and the implications they makes on the application-side.
Java / JVM	I know in-depth working mechanisms of JVM, its memory management, concurrency model; general development practices, and Java-fitting design patterns.
Python	What I am fairly effective with. Adequate knowledge of its interpreter, memory allocation behaviors, and familiarity with a good range of libraries for off-the-shelf solutions.
Client-side / JavaScript *(on browsers)*	Vanilla JS, React, Redux, Webpack, and some other frameworks that are not born this year.

Some other rusty skills or only used occasionally would include: **Clojure**, **iOS** (Objective-C), **C++**, assemblies (**ARM-arch**), **UNIX** (enough for shell scripting), etc. For tooling, I'm using a "vim-ified" version of **Emacs** as my main text editor. For large and structured projects, **JetBrains IDEs** are what I would go with.

Work Experiences

PrimeData

2021 - 2023

Director of Engineering / Co-founder

Situation

The company builds a so-called *CDXP* product. In brief, it is a *data platform* that tailor-made to host and process *customer profiles* (the keystone solution is how would you able to identify and unify your customers data from heterogeneous schemas and formats from diverged services and systems).

I joined the company after it had been running for *a year and a half*. The culture, team attitude, and the startup vibe is great. However, the mix ratio of engineering maturity is *lop-sided toward junior*. The system is developed in a *build-to-sell* manner rather than build-to-last.

Key Contribution

Problem: The very foremost stage of this data platform is *integration,* which, at that time, was far from being mature. Existing data connectors were a mixture of *ad hoc scripts* alongside cron jobs (and a surprising amount of creative uses of other kinds of schedulers). *Scaling*—in terms of both throughput and features—and maintenance cost *painfully*. The development cycles would undergo around **3~6 months** to have what we could call a solid connector. Bug fixes took **weeks**, even for straightforward ones.

What I did: I designed and implemented a **platform layer** instead of having every connector is a vertical work from the ground up. Recruited and handpicked members for a newly formed **Integration team**. A *development process* and *responsibility model* were also established.

Results: After the platform and processes were in place, a new connector could be in production in **2~3 months**. Still not a significant upgrade (a good portion of it comes from researching and API learning phase), but the maintenance and bug fix efforts shrunk down to a **couple of days** rather than weeks. The *development process* became *more direct* as well, instead of "everyone asking for everyone".

Aside from those two that I considered are my most meaningful contribution, other responsibilities are:

Leadership: I oversaw multiple teams of 20+ people, Profiles Backend, API Backend, Integration, DevOps, DataOps, and Frontend apps. Generally, I made sure each team has high ownership and accountability. I remain a consultant, occasional tie-breaker, and very rarely a veto holder for components that I don't consider myself the owner.

Engineering: I designed and implemented multiple components (Integration platform, DataOps toolset, part of the Profiles modeling, system tests template, etc.). A good chunk of time for writing documentation (and a comparable effort to compel everyone to do so).

Key techs: **K8S**, **Kafka**, **Airbyte**, **Unomi**, **Python**, **Java**, **Elasticsearch**, **PostgreSQL**, **ClickHouse**.

TreasureData

2018 - 2021

Senior Data Engineer

> **Yao Lu** — *Senior Director* of Engineering, Treasure Data Mountain View
>
> “Huy is one of the most talented engineers I have worked with. He brings a rich background in various techniques, solid problem solving skills, and great passion for innovation.
> Huy also did an excellent job leading delivery of projects, managing the schedule and progress.
> I have no doubt Huy will be a star employee for any team he works for.”

Situation

The company existing architecture is already well-sound and battle proven. My initial responsibility landed here is to take over the development of their iOS SDK and develop a new Unity SDK. A later handed hat is developing other server-side connectors.

Key Contribution

Problem: There is a well-acknowledged pain within the team is having to write and maintain **thousands lines of boilerplate** Java code *per data connector*.

What I did: I initiated the creating of a "**Connector Builder**", managed to complete a *DSL* and a mini-*IDE*—for composing connectors—in half a year.

Results: The ratio of SLOC reduction is around **20:1** (one connector has an average of *2,000 lines* in Java, would be cut down to around *100 lines* if written in the DSL).

Although it was not adopted company-wise (due to the dual-maintenance costs of having two ways to author a connector, otherwise facing the pain of transpiling hundreds of legacy connectors over), it sparked the creativity and originality engineering spirit throughout the company. VPs pushed for adoption. The entire integration and sales teams got training (so sales engineers could write PoCs with Connector Builder to prove the data sources are "connectable").

Beside that personal initiation, my other other contributions are: **a JVM bytecode analysis tool** for detecting runtime compatibility issues, **constructing Unity SDK** from the ground up, several new data connectors, traveled back and forth to Mountain View and Tokyo offices for engineering alignments.

Key techs: **Java**, **Clojure**, **Embulk**, **iOS**, **Unity3D**.

Earlier Career

EgoPulse
2016 - 2018

A health-domain startup. I was one of the founding engineers.
Stack: **Java**, **Rx**, **VertX**, **Rancher**.

Atlassian
2016

Jira, Confluence creator, Slack owner. Although unfortunately the HCMC office closed down when I was just onboarded for more than a month, this landed me the next opportunity in a newly formed startup EgoPulse by the former Head of Atlassian Vietnam and a dozen of other ex-Atlassian employee.
Stack: **Java**.

Simple Solution
2014 - 2016

We built a broad-communication product. It is used in SEA Games 2015 in Singapore.
Stack: **Java**, **VertX**.

ItsyBits
2012 - 2013

An out-sourcing company. Most projects revolve around mobile apps and games.
Stack: **JavaScript**, **Lua**, **PhoneGap**, **Cordova**, **Sencha**.

Personal Activities

I am an occasional speaker:

- Once on a VTC Academy Talkshow.↗
- Once for 2359 Media about the “data engineering career”.
- Twice at Barcamp, one is about "Interactive Development with Vim", and the other is "Game Development in 30 minutes".

Outside of work, I enjoy drawing↗, playing guitar, and practicing the art of diapers changing for my adorable newborn daughter.

I'll take the liberty to guess that there's not a lot of CVs/Resumes to screen today by the fact that you scrolled to this part. Then I'm gonna grab my chance to yada to you more about the writing of this resumé:

Q: I felt dizzy. Why use too many `font` **styles** *and* colors?

A: I did my best to optimize for "skimmability". Those choices of styling and layout is my hope to aid readers to navigate to the part that they want to pay interests in.

Q: You sure talked a lot, too informal and too 🧀.

A: I prefer having human touches and originality over the typical tiresome formality of daily work. Even if this perhaps will render a negative impact on the result of this resumé screening, I hope you found it amusing to read (or at least you can be sure that this wasn't written ChatGPT).

Q: What did you use to create this document?

A: I used OmniGraffle, a diagramming and illustration software. Probably not the most suitable tool out there, but I'm familiar with it enough from the daily basis of diagrams drawing to make it a pain using Microsoft Word or LaTeX.

```
mov x16, #1
mov x0, #0
svc 0
```