About me

My name is Luis Rodriguez and I'm a self-taught software engineer from Belize, Central America. I fell in love with coding at a relatively young age and over the years I've developed a burning passion for all things coding and tech. Coding truly is my passion on and off the job. I really enjoy building cool stuff for the web through the power of code. For me, it's the number one field in the world, it is at the forefront of innovation and many other other fields would cease to existing without software and technology. To be able to conceive an idea, then being able to realize it with just a computer and have it readily available to millions over the internet is just one of the many reasons why I absolutely love what I do.

As I mentioned, I'm 100% self-taught. I never finished university because I believed that it was a complete waste of time. I was learning a lot of irrelevant stuff and then a lot of my programming related courses were outdated and too text book based. So I decided to stop, and I knew I had a passion for coding that would take me through the self-taught route, and this ended up being the best decision I've made in my life.

I've worked with a number of languages, frameworks and cloud providers over the years but my favorite technologies these days are:

- React.Js
- Typescript
- tRPC
- AWS
- Rowy
- Vercel

On a more personal level, when I'm not coding I enjoy playing and watching sports. My favorite sport is basketball and my favorite team are the Lakers. My favorite athlete of all time is the late great Kobe Bryant. My favorite TV show is by far Breaking Bad then Game of Thrones, and my favorite movie is Lord of the Rings. Also, I must say that I'm a food lover, specifically pizza is by far my favorite food to eat.

Contact information

If you want to talk about tech, coding, projects or opportunities I can be contacted through any of the following:

- Email - rodgetech@gmail.com
- Personal website - https://www.rodgetech.com
- Phone number - 501-608-2077
- Twitter - https://twitter.com/rodgetech
- Github - https://github.com/rodgetech
- LinkedIn - https://www.linkedin.com/in/rodriglu/

My programming skills

So I've been programming for over 9 years now and I have a good deal of experience building web applications, provisioning cloud computing services on AWS, I've used a number of server-side technologies, and I've even built a few mobile apps along the way.

Programming languages I've used:

- **Ruby** - Ruby is one of my favorite programming languages. I absolutely love the syntactic sugar of the language and I like it even more because I can write server side applications with it by using one of my favorite frameworks - Ruby on Rails.
- **Typescript/Javascript** - Out of all the languages I know, Typescript is by far the one that I've used the most. I use it extensively at my job and in my personal projects. I like javascript too but I like typescript more because of the type-safety it offers but the number one reason I love Typescript and use it a lot is because I can have one language for the frontend using React JS and for the backend using Node Js. Absolutely powerful.
- **Php** - Php has a special place in my heart because it's the first server-side language I learnt and used. I know it's old and a lot of people look down on it but I can still remember how blown away I was when I used php to store persistent information to a mysql database. For me php was a prelude to full stack development.
- **Python** - The only time I've used python in the past was for web scraping and creating utility scripts for updating a shopify store inventory. It's a nice programming language but I havent had the need to use it a lot.
- **C#** - Just like with python, C# is a programming language that I've used but not extensively. In one of my jobs, I had to maintain and add updates to a legacy Asp.Net project which we eventually migrated to Nest.Js.
- **Html/Css** - Yes I know html and css are not really considered programming languages but all websites and web applications rely on it so I'll add it here. I still remember the hours I spent on youtube trying to learn html and css. It took me well over a year to become comfortable with it but it was well worthed because I was able to build my first website which was just a portfolio website. Honestly it was kinda ugly but it was a proud moment for me.

Every year I like to challenge myself to learn something new and this year I've been trying to learn Rust. Rust is a language gaining in popularity and I thought it would be a good idea to learn the fundamentals of a systems programming language. Once I learn the fundamentals of it I plan to make simple command line program. I am firm believer that project-based learning is one of the best ways to learn a new technology.

Frameworks & Libraries

I have worked with tons of frameworks and libraries over the years. These days it seems like every other day another framework is being released especially in the javascript/node js community.

Here are some frameworks and libraries I have used:

- **React** - This is by far the number one ui library that I've used. I've used in the past for building the ui of web apps and even cross platform mobile apps with react native. React is super popular and I've seen why, it makes me so productive when writing ui components because I can easly make them reusable using composition and jsx makes have my ui logic mixed with my ui code feel like a good thing. Before react I used to just rely on vanilla javascript or jquery for creating interactive web apps, but now those days are long gone.
- **Angular** - OK, I'm not the biggest fan of Angular but I did used it before React in one of my previous jobs where I was tasked to build out a dashboard for a local Tv & communication company. It's a nice framework, but for me it's kinda bloated, and the reason I Prefer react over angular is because it's less opinionated.
- **Ruby on Rails** - Let me switch to the backend for a bit, and talk a about one of my favorite technologies of all time - Ruby on Rails. Before ROR, I used to use PHP and the LAMP for writing server-side applications but that quickly changed once I discovered ruby and then ruby on rails. I still remember, staying up late at night, reading the Rails tutorial book by Michael Hartl, and to this day I think this is the only book that I've read through to the end. What I loved about ruby on rails was: ruby, the magic nature of the framework, and maturity of the technology and community. Rails was magical in a sense, because you could quickly scaffold an entire crud service, and ActiveRecord made working with the database unequivocally unmatched. Stuff like User authentication and handling payments, were all made easy by using gems. Learning ruby on rails was a prelude to cloud computing for me. Because of it, I learned how to setup and configure my own VPS from digital ocean. I learned to setup a webserver and mysql database. It was truly a huge step forward for my career.
- **ASP.NET Core** - I've only once had the opportunity to work with dotnet and this was in one of my jobs where I had to maintain an existing API. This is also one of the few times I had to use C#. This was an api only project that connected a redshift database, we used the entity framework nuget package for modeling the database structures. One of the more interesting I did was dockerizing the existing application so that we could deploy it to ECS. But after a while, we decided to migrate from ASP.NET core to Nest.JS, thus ending my time with dotnet, for now.

Personal Projects

I previously mentioned that I consider project-based learning one of the best ways to learn a new programming language or technology. As such, I have built a couple of personal projects to help me better understand how a particular technology works.

If you're interested, you can find the code for most of these projects over on my github profile: https://github.com/rodgeTech?tab=repositories

Now let me go through a few of them that are worth sharing:

1. bmmerce

bmmerce is an app I built that allows you to buy and sell stuff nearby.It connects sellers with the nearest buyers thus making buying and selling in Belize as simple as meeting and exchanging. You can think of it as a more basic version of the OfferUp app.

In its current state bmmerce is an in-person transaction first service, and because of this buyers & sellers become more than just buyers & sellers; they become legitimate vouchers for one another. All of this evaluates to a more reliable and trustworthy buy and sell platform for Belizeans to utilize.

Main Features:

- Browse nearby stuff to buy.
- Post to sell. Nearby users are instantly notified of your new listing.
- Receive notifications when someone nearby posts something new to sell.
- Engage with nearby buyers & sellers. Message to meet & exchange.
- Manage all your listings.

I took on this project with the goal of learning react native as well as learning the fundamentals of mobile development. In completion of this project I managed to grasp a solid understanding of react native but even more importantly I learnt what it took to take an app from idea to design and from design to implementation and finally to publishing on the google playstore. I was unable to publish to the appstore because at the time I did not have an apple computer and it is required by Apple for you to have macOS in order to create a production build.

The app is currently unavailable for downloading as I intend to migrate it from DigitalOcean over to AWS.

2. Quikapply

Quikapply is an easy to use service that you can use to bring your printed applcation forms online. It's flexible enough so that you can use it to design almost any application form and has

support for the most common forms on user inputs. Each application published will have a public apply form that you can share with your target audience and start receiving submissions.

Going into this project I already had a solid understading of using react and redux however I wanted to learn how to use react with typescript instead of vanilla javascript, and I also wanted to learn what seems to now be the recommended way of using redux - redux-toolkit.

I wanted to focus as much as I could on the frontend so I decided to use Ruby on Rails for the backend since still to this day it's the backend framework I'm most familiar and comfortable with.

All in all, after working on this small project I now have a solid understanding of using react with typescript and using redux-toolkit and actually from here on I don't see myself ever using react without typescript 😀.

3. RapidBoxing

This was a shopping and shipping service that I developed for a local business who wanted to target both web and mobile platforms.

Shopping and shipping from the US to Belize has always been a cAn intuitive and modern experience driven by two platforms – web and mobile. The solution was a platform that encompasses all the most vital features essential to any international shopping and shipping service:onfusing and inefficient process, most of the time being impractical for most Belizeans.

Main features:

- Order via link submission
- Catalog ordering
- Order estimation
- Order tracking
- Invoices & receipts
- Order updates (email & push notifications)
- Customer dashboard
- Admin dashboard

The project’s code base was divided into four parts:
API (Rails) + Website (Rails & React) + Android app (React Native) + iOS app (React Native)

AWS has now become one of my goto provider for all things cloud computing. Goodbye DigitalOcean and Heroku.

I used a variety of AWS services to host this application. One service that I want to highlight is AWS CloudFormation. In short this is a tool that enables you to define your infrastructure via code. I used this tool to setup all the provisioning you see below. It is a must have for me for the

mere fact that it eliminates a lot of redundancy and saves time. I highly recommend you take a look at it.

In the old days when I was just learning how to setup a VPS I would manually install the system tools, system libraries and settings. Good for learning but it gets repetitive.

More recently, I've started using Elastic BeanStalk to deploy and manage my web applications. This eliminates the need for me having to SSH into the server and manually setting the stage. It's also makes deployment a walk in the park.

Thanks to react native I was able to quickly develop a prototype of not only the iOS app, but of the android app as well. Being a one man band made it impractical for me to develop for both platforms using their native SDKS.

I still have it down as a goal of mine to one day develop and app using its native SDK. But for this project, react native was a no brainer.

In conclusion, this was a very challenging project to work on mostly for the fact that I was the only one working on it. As such I had to make some tradeoffs, some of which I'm not too proud of.

The main one being the overall test coverage of the codebase. The mobile app has literally zero test coverage and for the API I only tested the critical parts such as invoice generation, order estimation, and email delivery.

4. Hedjintrealtyltd

Hedjintrealtyltd is a web service that serves as realty manager and customer engagement tool. It has support for user management, listing management, and a special feature called engagements - an engagement between listers and potential buyers via SMS or Email.

Main features:

- Listing management
- User management
- Engagements
- Content management

When I started working on this project, I was still in the process of learning Ruby on Rails and at the time I had limited experience with full stack development aside from the LAMP stack. I was actually in the middle of reading this amazing book which you can find at https://www.railstutorial.org/book. It really helped me more than anything to grasp a good understanding of the framework but even more importantly it helped me better understand the

workings of backend frameworks. Things such as MVC and migrations are just a few of the foundational knowledge I picked up by reading this book.

For hosting, I used DigitalOcean. This was another thing I had to learn upon completion of this project. Prior to this point, I had no experience of setting up a server and installing and configuring all the tools required by this application such as the webserver, database, redis, etc. Luckily for me, Ruby on Rails is a very mature framework and there were tons of tutorials that showed how to deploy to a multitude of providers, but I went with DigitalOcean.

Definitely not the best looking site out there as design is not really my forte, but aside from that I'm really glad with the outcome and even more so with the things I learnt along the way.

4. cron-ai

You can find this project live at https://cron-ai.vercel.app/. With the introduction of GPT, a lot of AI related projects have been trending these days. This is a natural language to cron expression generator that I built with the main goal being to learn how to use OpenAI and it's GPT model. I built this project over the weekend and the reason I was able to build it so quickly was because it really was a basic project to work on, but also because I used Next.js along with Vercel which is a match made in heaven.