Prasanna Natarajan Resume

Table of Contents

Objective....3
Summary....3
Education....3
Employment....4
Projects....4
Skills....9
  Technical skills....9
Readings....10
Interests....11
Contact....11
Personal Details....11

Objective

My number one career objective is to benefit the business I’m working with, in 2 ways: by increasing their revenue, and/or decreasing their operational cost by leveraging my technical skills.

Summary

I’m a Ruby on Rails and ReactJS expert with 9+ years of total and 5+ years of relevant experience.

I can design cloud-scale infrastructure, develop great webapps with Rails, and deploy them in the cloud with modern Devops tools.

I'm a full-stack developer with expertise not only in Ruby, Rails and Javascript, but also in the other technologies surrounding this ecosystem like SQL, Git, Shell Programming, Unix, Server Configuration/Provisioning, Application deployment, HTML, CSS, Chef, AWS etc.

I'm aware of agile approach to software development and have been part of many sprint planning and scrum calls.

Education

I have a Bachelors degree in Computer Science Engineering, from Anna University Chennai, India. I graduated in 2008.

Employment

Current

-

Previous

- (April 2017 – Nov 2017) Zanec Software Technologies. Role was “Technical Lead”.
- (Mar 2014 – Feb 2017) RubyEffect. Role was “Solutions Architect”.
- (May 2016 – Dec 2016) Velankani. Role was “Lead Engineer/Solutions Architect” as a Contractor from RubyEffect.
- (Nov 2008 – Jan 2013) Infosys. Last role was “Technology Analyst”.
- (Feb 2013 – Feb 2014) Aricent. Last role was “Senior Software Engineer”.

Projects

Mozo’s main product (Zanec 2017)

https://mozo.com.au/ is Australia’s leading financial comparison and review site. I was also responsible for:

- leading the remote team at Zanec to collaborate with Mozo’s technology director and his team. We are 2 devs and 1 designer.
- managing project delivery. Mozo will usually just give specs for any new feature they want developed, and set a deadline. It’s my responsibility to get my team to build and deliver the project successfully with as little help as possible.
- Writing ruby and javascript code that the daily tasks require.

The main projects I worked on for this client were:

A new React app (backed by Rails) for Mozo’s partner Choice.com.au. I learnt react for this and developed all of the react and ruby code. The app is behind a paywall. But here’s a screenshot: https://npras.in/assets/images/portfolio_zanec_choice_react.png. Things used in react: redux, saga, multiselect, redux-form, css modules and webpack.

Built from scratch a custom CMS tool that’s used for creating marketing articles. It’s designed to create articles of various pre-defined “templates”. An example article created from this tool: https://mozo.com.au/personal-loans/articles/get-money-ready-for-christmas-with-a-low-cost-personal-loan

Built another React app for Mozo's backoffice admin site. It was a complex redux form that fetched data remotely and rendered many components conditionally based on business logic.

Tata Communications’ Sponsored Data Exchange (Velankani 2016)

This was an exciting project that I worked on. In a span of 4 months, I implemented 3 projects (2 Rails, 1 RubyScript), designed a cloud-scale infrastructure, and then deployed them to the cloud using modern devops tools.

I lead a team of 3, did over 80% of the design and development. I was responsible for client communication for requirement gathering and status reporting.

In the first phase, I built the 2 Rails apps that exposed many APIs. I used Rails 5. Based on the client’s high-level model requirements, I designed the low-level database and entities with relationships. I had to take care of all query performances too. I wrote test cases for all APIs and made sure the test coverage is over 90%. I also used rubo-cop gem to enforce a coding standard among all members of the team.

For billing requirements, the client wanted to use a NoSql DB too, apart from the traditional Mysql RDBMS. We chose AWS’s DynamoDB, and implemented the right partition and sort keys that would prove crucial to building the RubyScript app later on.

In the second phase, I designed and deployed the infrastructure required to scale this project in the future, in AWS.

- I used Opsworks to setup these 3 app in 3 different stacks.
- I wrote custom Chef recipes, when Opsworks’ in-built Chef Recipes fell short.

- I defined a secure architecture where these apps would run within AWS: I created
    - VPC,
    - private and public subnets,
    - route tables and routes to define traffic among them,
    - NAT Gateway and Internet Gateway to enable the network to talk to internet,
    - Security Groups to control access to ports,
    - Elastic Loadbalancers to distribute traffic among multiple instances running the same application,
    - Cloudwatch metrics and alarms to watch important KPIs
    - Setup Loadbased Autoscaling instances that would come to life when the load on app instances go beyond a certain threshold
    - Defined IAM Users and Roles to provide/restrict access to these resources

In the third phase, I built a Cloudformation template that codified everything I manually did above. This template was the final delivery of this project. With this template, the client will be able to deploy these 3 apps, along with the secure infrastructure defined above, in any AWS account, in any region across the world.

The client Tata and Velankani was impressed with the work I did here, especially about how I was able to dive deep into new areas like most of the AWS Services. My suggestions about improving/changing some already-documented specs were also noted positively.

Numerex IoT (Velankani 2016)

From Rubyeffect, I went to Velankani as a Contract-hire. Numerex was a big project. They were championing IoT in various domains. We were set to take over the Rails development work from the Onshore team that was managing it previously.

I lead a team of 3. During the initial phases of the project, I would gather the team together and we will together walk through any particular component of a project. I would show them many ways to navigate around the codebase quickly, explore certain classes and libraries, and share with them testing and debugging techniques.

With limited time, documentation and knowledge sharing, we had to reverse engineer and build custom internal tools to simulate the devices' behaviour, and to help our testing team verify our implementations.

It was a set of 5 Rails applications. They were using Rails 4 and Ruby 2.x.

In the Rails apps, we followed best practises and wrote elaborate ruby test cases to verify our implementation. The client was happy with the level of details, and quickly trusted our abilities.

We were tasked at building many features for these apps using the Agile methodology.

ParkMobile USA (Rubyeffect, 2015)

Parkmobile http://us.parkmobile.com/ is the leading parking solution provider in USA. We (3 members) worked for them for 8 months.

We worked on new feature requests and fixing bugs. I remember 2 interesting tasks that *I* specifically worked on:

1) Fixing memory leak:

The production rails app, which was deployed in 4 servers would have its memory rise high whenever there was more of a specific activity in the site. We'll get alerted from Nagios once it gets the threshold. It was an established routine that when we get the alerts, we'll hop on to all the app servers, and restart the apps 1 at a time. This solved the issue, but we were only treating the symptoms.

I investigated the issue using tools like: rbtrace, memory_profiler, and ObjectSpace, found the offending code and fixed it.

2) Adhoc script to charge a million dollars from customers:

The script would be run once in a while when a discount has to be applied on a few customer's purchases. Since we are charging money, I was careful to capture all possible cases where the input might not be as expected, and collect and report relevant errors, and the customer data that caused it. In just 4 days, I had to write this script. When it was deployed in production, it ran successfully without any incident, while bringing in an additional million dollars in revenue.

In both the cases, my work was appreciated from the client.

The SeaChange Rails Project (Aricent, 2013)

It was 7 rails applications for a single client, all in telecom domain http://www.schange.com/. My managers and the onshore client liked some of my ideas that we implemented. We both were responsible for implementing user stories and fixing bugs.

From our end, I was the one who introduced testing (using rspec) for one of the projects here and made sure the coverage was over 90%.

Ubertor Rails+PHP Project (Rubyeffect 2014)

My first project at Rubyeffect. Itwas to fine-tune the performance issues of a real-estate website project http://ubertor.com/. The website was earlier written in PHP, and they had transitioned recently to Rails, but the company who did the transition did a poor job. Many things had broken, and complex time-consuming queries were written. I was tasked to find and implement as much as performance optimisations. I implemented the “rails-best-practices” gem to find out and fix many issues. Using “explain plan” I made many queries faster, and using tools like Newrelic and Bullet gem, I was able to find and fix many N+1 queries too. There were many unnecessary loops and variable initializations inside the loops that caused the performance bloat. I fixed it all.

HighRankWebsites, a Cuba Project (Rubyeffect 2014)

I learnt a lot about SEO in this project http://www.highrankwebsites.com/. This popular SEO company had 3 tools: KeywordCompetition Analysis Tool, Link Prospector and Rank Tractor.

These tools were webapps, but the highlight was, they weren't built using Rails, the industry standard. Instead, these apps were built using the tiny Cuba Framework http://cuba.is/ built on top of Sinatra. But the apps themselves weren't tiny. There were many typical components of a standard webapp built into it.

I took control of all 3 tools, and the site's CEO would often want me to add features on a weekly basis. The client was impressed with the work I did here, many times.

Parkmobile Stomp AngularJS (Rubyeffect, 2016)

This was a short term project, about 3 months. We were working with the onsite team that developed a new AngularJS SPA for one of their requirements. The Angular app would make API requests to about 10 rails apps that were structured as microservices.

In the few days prior to starting the project, I started learning AngularJS, and was able to fully serve the project needs by the time it started. I was tasked with adding APIs to the backend microservices, adding directives, and controllers to the Angular app.

Qzick (Rubyeffect, 2015)

Qzick http://qzick.com/ was our in-house product that we built at Rubyeffect. It is an assessment tool that can be used by schools/colleges to test their students, it can also be generalized and used by other institutions like software companies and restaurants.

I was one among the few core members that built this app. I designed and implemented the APIs required for the application in Rails using Grape API.

JASS Tool (Infosys, 2009)

I was part of a 4 member team that built a Java tool that could deploy a J2EE app to multiple servers (weblogic, websphere, tomcat etc). I specifically coded 2 things: the module that deployed the app bundle to weblogic server using the latter's api, the module that allows the deployments to be configured and scheduled.

Jboss Configuration Engineer (Infosys, 2009)

Yes, I configured Jboss. I used perl or shell script to automate the process in a server. Jboss was another java application server that was hot at the time. I bought a book about it, and also attended a workshop training about it.

Distributed Chat Tool (Infosys, 2010)

I was made a lead for 3 junior devs, and together we set out to bring into reality, the vision of our manager. He had an idea for a distributed desktop chat tool, that would benefit from his unique architecture design. Together we spent a lot of time discussing over the finer details of the application.

After the design was finalised, we set out to build the tool using C#. Our manager applied for patent for this tool, and we got it. I'm listed as one of the 5 co-inventors: https://www.google.com/patents/US20130325962

Skills

Technical skills

Ruby/Rails

My main expertise. I can do things with these that will make the bottom-line of your business rise high.

- I have worked with till Rails 5/Ruby 2.4.
- I've developed REST API apps with authentication, authorization and Admin backoffice systems.
- I can delegate tasks to background workers like sidekiq, delayedjob.
- I can fine-tune the performance of your Rails app, and make queries faster.
- I write tests using minitest, rspec, cucumber and mocks.
- I'm aware of Ruby's metaprogramming, and can dive into the source code of any gems to understand and debug issues.
- I can debug the root cause of a bug quickly using a variety of tools like: byebug, Tracepoint, by raising and catching exceptions. I can find and fix memory issues in a Rails app.
- I've opensourced a gem I built for a project in my company RubyEffect at https://github.com/boyhavoc/craigs_gem_pub. You can sample my code there.

Javascript

Javascript is as essential as ruby for a Rails programmer. From being able to use it as just jquery, to being able to write full-fledged modules, I can do it all in Javascript.

I can use it to build Single-Page-Applications using the modern frameworks like ReactJS. I've built 2 react apps for the client Mozo.

I'm experienced with the recent changes in the JS world: webpack, ES6/7, Node, SPAs, promises etc.

Design infrastructure for your Apps

I can design a working architecture for your apps that could scale well in the cloud. I know how to setup, in AWS, a VPC, create subnets, setup route tables, routes, NAT Gateway and Internet Gateway. I can configure Security Groups, Elastic Loadbalancers and RDS for your application.

Devops

I can deploy your apps to production using Capistrano, Opsworks, Cloudformation, Chef, or Terraform. I have experience with all.

I can write custom chef recipes to codify any server setup. I've written recipes to:

- deploy rails and non-rails apps to servers
- install and setup monit, apache, mysql etc.
- Setup cron job
- write custom configuration files into the server

I can use AWS' Cloudformation or HashiCorp's Terraform to codify your entire infrastructure. The resulting configuration template can then be used to bring up instances of your app in a predictable manner.

I can wrangle any ubuntu server to setup firewall, install packages, configure ssh, webservers etc.

Readings

Here are the technical books I've read so far:

- Fullstack React – https://www.fullstackreact.com/. This is the book from which I learnt in detail about ReactJS.
- The Ruby Programming Language Everything You Need to Know – David Flanagan, Yukihiro Matsumoto – http://shop.oreilly.com/product/9780596516178.doWhen I decided to get serious about ruby, this is the first book I got. It builds a solid foundation, much better than blogs and tutorials.

- Metaprogramming Ruby 2 – Paolo Perrotta – https://pragprog.com/book/ppmetr2/metaprogramming-ruby Great little book that taught me the metaprogramming secrets of ruby.
- Design Patterns in Ruby – Russ Olsen – http://designpatternsinruby.com/ This book translated the classic Design Patterns into Ruby. Jan 2016 The examples were nice, and knowing about the patterns helps me in grasping any new code base easily and design my code too in a standard way.
- Confident Ruby – Avdi Grimm – https://www.confidentruby.com/ Still reading it.A great book from the master Chef Avdi, explaining how to write confident ruby. It explains what other coding books don't: how to structure and write an actual method! This knowledge can be used in any other languages too.
- Exceptional Ruby – Avdi Grimm – http://exceptionalruby.com/Nice little book about ruby's exceptions, from, again Avdi the master chef.Lots of nifty tricks, and theories explaining when to use exceptions and when not to.
- Practical Object-Oriented Design in Ruby – Sandi Metz – http://www.poodr.com/I read it too early in my ruby career, when I was just beginning to learn ruby. I don't remember anything from the book now.Especially because I started using Anki to manage my learnings only recently. I have to read this masterpiece again.
- Professional Javascript for Web Developers – Nicholas C Zakas – http://www.wrox.com/WileyCDA/WroxTitle/Professional-JavaScript-for-Web-Developers-3rd-Edition.productCd-1118222199.htmlRecommended by Derek Sivers as the "go-to book" for mastering javascript basics. Didn't fail to meet my expectations. My understanding of prototype, inheritance, apply, call, property lookup, functions and objects all stem from this book.
- Javascript Cookbook – Shelley Powers – http://shop.oreilly.com/product/9780596806149.doThe cookbook format of this book was very refreshing. Standard problems, and the javascript snippets for them were useful to me.
- Multitenancy with Rails – Ryan Bigg – https://leanpub.com/multi-tenancy-rails-2The books main pitch is to teach multitenancy using apartment gem. But that wasn't interesting to me. What interested me was the book's approach to TDD. I learnt a lot about writing clean and meaningful tests. Tests that only test the behaviour of a feature, and not the little technicalities.
- The Passionate Programmer – Chad Fowler – https://pragprog.com/book/cfcar2/the-passionate-programmerGreat little book emphasizing on the art and the profession called "Software Programming". Inspiring little bits about how to improve yourself as a programmer.

Interests

When left alone, I’ll prefer reading books from my Kindle, or watch good movies.

Contact

If you think we can work together and if you think I’d be a good fit for your culture, I’d love to talk to you.

Email: prasann87@gmail.com (yes, the trailing ‘a’ is not present)

Mobile: +91 7708087998, +91 8095536527

Skype: prasanna-n

Personal Details

I’m married and I live in Bangalore since 2010. I’m originally from the beautifully hot city of Madurai, from Tamilnadu.

As of 2017, I’m 30 years old.