A little about me

Full Stack Developer, interested in IOT, robotics and tinkering with consumer electronics.

Working at Browserstack with Ruby on Rails, React and NodeJS.

Robocon Alumni and SIH Mentor. 2019 passout.

Designing Production Systems

Those tiny decisions that make your life easier, long-term

Edwin

Preface

Lessons Learned while supporting an in-house app I designed, and primarily coded while at Browserstack.

What I did, what I didn’t do, and how both of those came to bite me post midnight on a Saturday.

App Requirements

- A web-site that shows the details for each client.
- Connects to all in-house and external APIs.
- Should be more responsive than the previous solution while at the same time consume less resources.
- Should be able to run periodic tasks.
- Should run daily health reports.
- Should be intuitive, requiring little to no training for the Sales reps

Designing

- Choosing the Tech Stack
- Choosing/Declaring/Enforcing the API interfaces
- Choosing/Enforcing the coding standards
- Designing Repeatable Deployment Systems
- Integrating Redundancy
- Adding Revert Mechanism
- Adding Observability
- Enforcing Documentation

Final Touches/Fine Tuning

- Periodically Check Load Capacity(no. of requests that can be handled).
- Prevent machine from going down, due to your code.
- Add abstractions, but not to the point of absurdity.(All problems in computer science can be solved by another level of indirection .... except for the problem of too many layers of indirection.)
- Actionable Logs that can be comprehended within minutes.
- Memory Leak Detector.
- Error Reporting, Triaging, Assign To Dev.

Suggestions

- Clean Code - (Clean Code - Uncle Bob) https://youtu.be/7EmboKQH8lM
- Version Control Hygiene(Learn Git Flow, or Github flow, etc)
- Learn Common Design Patterns(Singleton, Producer-Consumer)
- Check out Code from Seniors at your place(Usually, there's a reason for the weird layout of code)
- Improve yourself by watching Conference talks
- Learn a bit on Haskell and Lisp. Gives a different perspective to coding.

Any Questions