Nolan Neustaeter Full-Stack Web Developer

Code Typer | CSS Wrangler | Data Organizer | Responsiviser | API Enthusiast

Hello, my name is Nolan and I make things. Mostly I make things for web browsers but I also make:

- overcomplicated woodworking projects
- a fantastic butternut squash soup
- lame “dad” jokes

I first started making websites for family and friends 19 years ago. Since then, I’ve worked with other platforms and languages but I’m always drawn back, eventually.
Some people complain about how fast the industry changes but that’s always been one of my favorite aspects of web development. There’s always a new or interesting language, library, or platform to learn so boredom is hard to come by.

Lately, I’ve been focussing more on front-end development so I’ve been able to learn some amazing libraries like React and Vue.js as well as some of the newer tooling projects like Webpack.

I currently use a Macbook Pro for development with as many monitors attached as I can manage.
I use Sublime Text 3 for my editor with Firewatch as my syntax theme and Fira Code for my font.

Languages and Frameworks and Platforms.

Stuff I use every day

Languages

- JavaScript (es6)
- PHP
- HTML
- CSS
- SASS
- SCSS

Frameworks

- React + Redux
- Vue JS + Vuex
- Laravel
- Bootstrap

Libraries

- Lodash
- Axios
- Draft.js
- Quill
- DropzoneJS
- Dragula

Platforms

- Nginx
- Apache
- Linux
- Docker
- Amazon Web Services (AWS)

Tooling

- Git
- NPM / Yarn
- Webpack
- Gulp
- Babel

Technologies

- Flexbox
- CSS Grid
- LocalStorage
- REST APIs
- JSON
- Websockets
- Continuous Integration

Stuff I have used

Languages

- C++
- C#
- Java
- COBOL (yes, really)

Frameworks

- Symfony
- Codeigniter
- ASP.net

Platforms

- Node
- IIS
- Google Compute Engine
- Wordpress

Stuff I am learning

Languages

- TypeScript
- Swift
- Rust

Frameworks

- Angular
- React Native

Technologies

- Augmented Reality (AR)
- Progressive Web Apps (PWA)

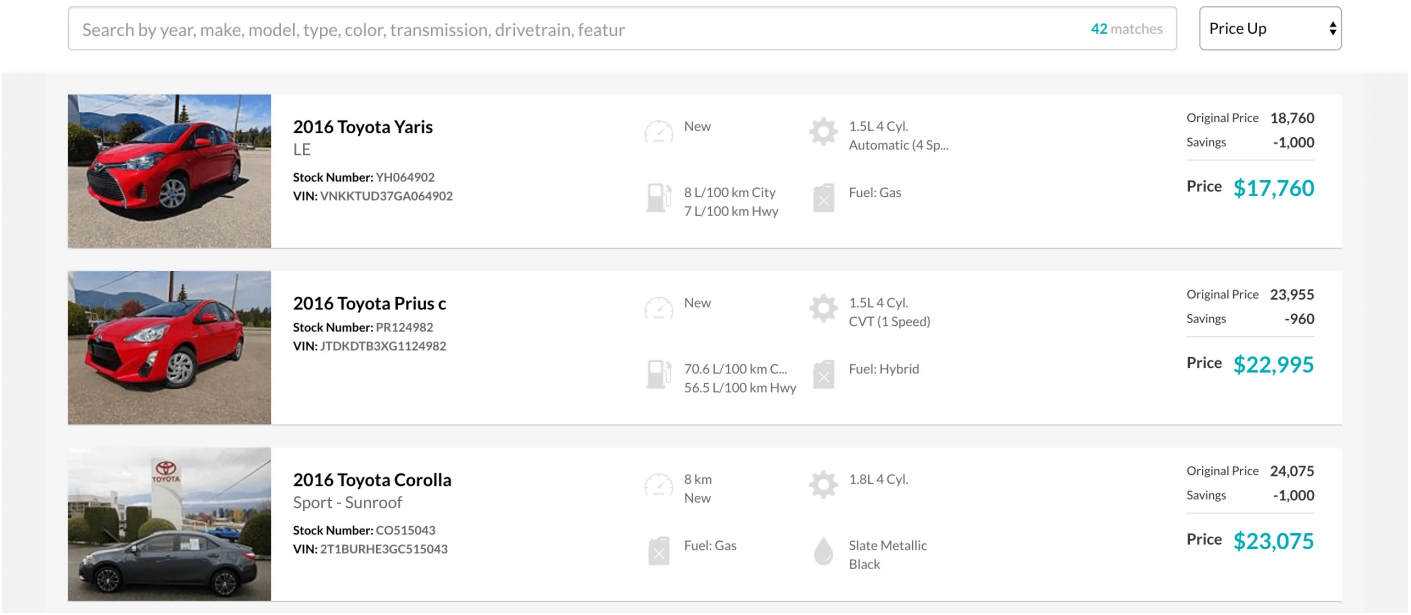
Recent Projects

In 2015 I rebuilt my employer's product listing component using React.

It features a prominent search bar, optional filters, and automatic pagination when the user scrolls near the bottom of the page.

The product data is stored locally as an array of JavaScript objects so the searching, sorting, and filtering is nearly instant. The design features a generous amount of whitespace and is fully responsive on smaller devices.

Here is a link to one of the sites that is using the product listing component.



Screenshot of the product listing page built with React

In 2016 I was tasked with building an online marketing platform with a website / page builder at it's core.

Over the years I had built multiple CMS' for my own use and I had been tossing around some ideas in my head on how to structure the server and database since then.

For the server, I split the functionality into three domains (both figuratively and literally), each was a separate application that shared common business logic code and libraries where possible.
The three applications could run on the same machine with the http server distributing requests based on subdomain or the applications could split off into their own machines/VMs/instances/containers if needed.

Application	Visibility	Subdomain	Responsibilities
Sites	Public	`www.`	In charge of displaying published websites, processing form submissions, and collecting analytics.
Control	Private	`control.`	Authenticates users and serves them the SPA
API	Private	`api.`	Provides endpoints for the SPA to allow it to manage businesses, users, websites, pages, etc...

Each application had it's own routes, controllers, and views but shared models, service providers, middleware, and config (through .env files).

Here's a listing of the stack we settled on:

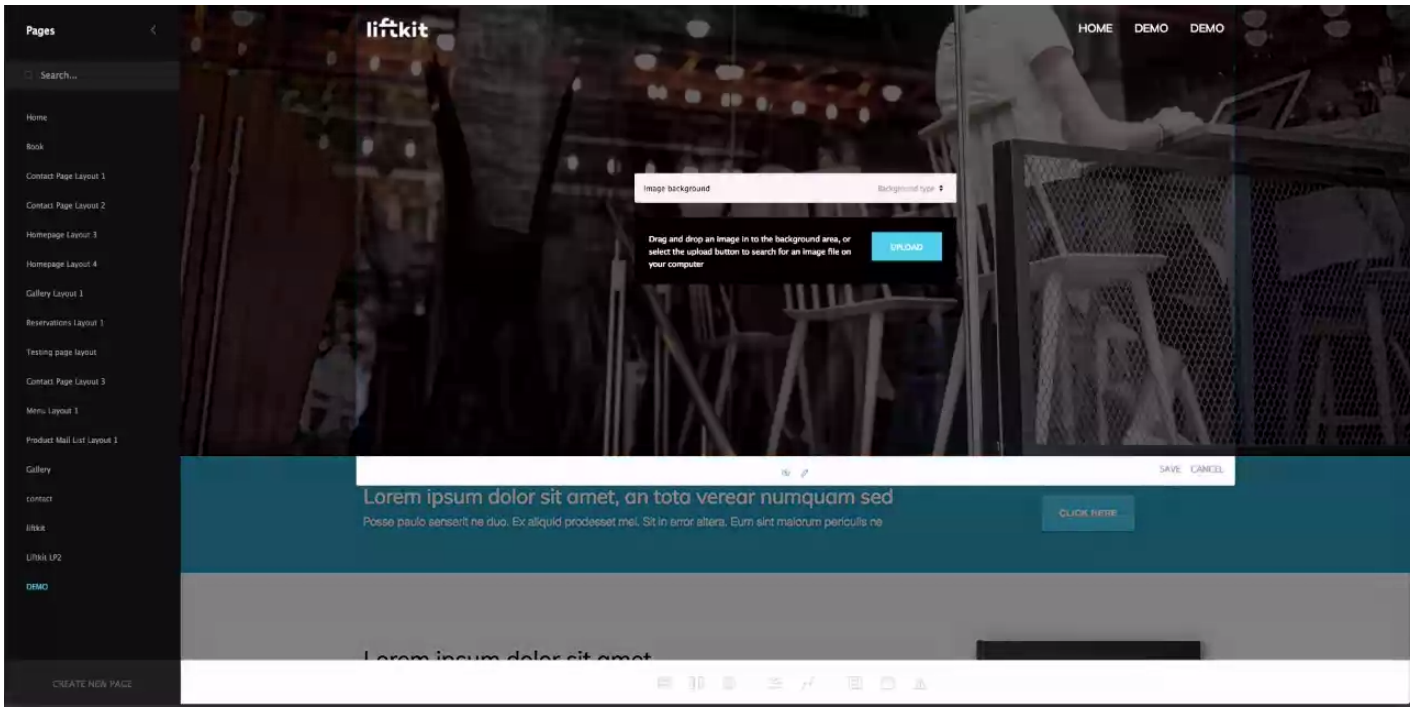
Hosting
Amazon Web Services EC2 - fast and cost efficient

Web Server
Nginx - fast and very flexible

Backend language
PHP7 - huge speed improvement over PHP5.x and the other devs are already familiar with PHP from our existing products

Backend framework
Laravel and Lumen - I've been using Laravel since version 2 and I love it's focus on code readability and providing sane defaults that are easy to customize

Frontend framework
Vue.js - This was a tossup between Vue and React. The main deciding factor was that I would soon be bringing some junior devs on to the project and Vue is much easier to learn

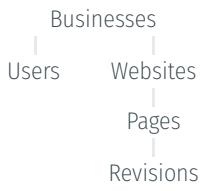


Teaser video of the page builder function of my previous project

To the right is a diagram of part of the data structure I designed. Pages don't store their content directly, the revisions do.
A new revision is created every time the page's content is updated.

- If the revision is from an autosave while editing, it will contain a delta representing that change and it will be marked as a `draft` revision.
- If the user is happy with their changes, they trigger a publish which creates a revision marked as `published` with the complete content.

When a page is requested through the Site application, the content from the latest published revision is used to build it.



This structure has a few advantages; content changes can happen without affecting the live version of the page, changes can be reviewed and reverted, A/B testing can be easily implemented by having one revision marked as A and another as B.

References available upon request.

send me an email at nolan@neustaeter.ca
or
call me at (250) 804-8589

The source code for this document and a few other experiments can be found at:

https://github.com/noolan