**The Book**

Along with another brother in my fraternity, I built The Book to serve as a tool dedicated to the Rush process at my university. Essentially, members in our Greek organization are now able to better organize and sort through new prospective members as we learn more about them. More importantly, The Book acts as a platform where we can quantify interest in those prospective members and visually represent the progress of our recruitment process.

My primary role in the creation of The Book was in conceiving its design and implementing its front-end development. This was actually my first time working on a project built in Ruby on Rails, so my first step was just tackling the slight learning curve that goes along with that: understanding the Rails structure and getting a hand on embedded ruby. Afterwards, in the development of the site, I also utilized Masonry, a grid layout library to display the array of prospective members.

Approximately 120 people are now actively using The Book.

**Master of Science in Robotics Site**

After working for a summer to develop projects and curriculum for one of the courses in the new Master of Science in Robotics Program at Northwestern University, my boss asked me to build the new program’s website. His vision was to have a site that adequately displayed everything that the program’s students would be working on in addition to housing a database of resources that students and faculty would contribute to over time. In addition, he wanted a website that could be easily maintained using GitHub and Markdown.

To meet that last requirement in particular, I chose to develop the website using Jekyll, a “back-end” (not really) static-site generator. Jekyll’s main strengths include its incredibly painless integration into the GitHub Pages hosting service and its feature that allows a site’s content to be maintained in Markdown.

**Census**

During RedesignNU, a hackathon hosted at Northwestern University, I, along with three friends built Census, a platform that allows students to communicate and share feedback with professors in real time during lecture. Using Census, students can communicate: “Please slow down”, “Could you repeat that last point?,” “Please give an example,” and other relevant items. My team recognized the problem that in large lectures, students often feel uncomfortable interrupting class to ask a question, even if that means not understanding a concept.

Building Census, we chose to use Meteor to facilitate the real time interaction. As the interface designer and front-end developer in the group, for me that meant understanding and implementing the template structure that Meteor uses in addition to my more standard responsibilities. I also developed and provided Census’ branding.

At the hackathon’s culmination and after a week of student voting, RedesignNU awarded my team the Student Choice award.

**RJ**

I began working on RJ the summer after my freshman year at Northwestern University’s Neuroscience & Robotics Lab. RJ’s real name is Baxter, and he’s a 7 degree-of-freedom robot built by Rethink Robotics. My task was to create a demo using RJ that would advertise the lab as it ran non-stop in a popular student lounge. After about a month of becoming somewhat proficient in operating RJ and developing software for him, I decided to combine his capabilities with those of an Asus Xtion Pro, a sensor very similar to a Microsoft Kinect that uses infrared and depth-detection technology intelligently.

With these two powerful pieces of hardware, I created a natural interaction interface that allows passerby to stand in front of RJ and have their arm motions mimicked in real time by his own.

The demo ran for most of the 2013-2014 academic year in the Willens Atrium in Northwestern’s Technological Institute.

The following summer of 2014, I continued using RJ as I developed projects and curriculum for the new Master of Science in Robotics program at Northwestern.

**JonRovira Dot Com**

This website has been in iterative development since my senior year in high school when I first went through the HTML and CSS track on Codecademy. I initially thought that it would be a little redundant to include it as an item in my portfolio here, but I’ve put a considerable amount of work into it and have learned immensely while building it, so I think it deserves to have a spot among the rest of the work I’ve done.

When I made the first instance of the site, my goal was just to put my newly learned knowledge of HTML and CSS to work and experiment in building a simple website. Since then, I’ve integrated multiple front-end and back-end tools into the site and I am comfortable saying that the result is a good representation of what I have done in web development, software development, and design.

To facilitate the contact form in a clean way and set up Handlebars, this site’s backend is built in Node with Express as a framework. The front-end view engine uses Handlebars and styling is done with SASS in addition to the typical HTML and Javascript/jQuery combination.

**DJ Raffe**

Tasked with creating a tangible system that promoted learning, I, along with two teammates built DJ Raffe, a tool that helps children develop polyrhythmic skills. We recognized the difficulty present in learning polyrhythms, rhythms that involve playing multiple independent rhythms on top of each other, and agreed that adding a physical component to a digital interface would stimulate learning effectively.

Essentially, a user would be given two physical color-coded drums (green and blue) that send signals to our interface running in a browser whenever pressure sensors on the drums’ surfaces are triggered. The interface itself would stream notes across the screen in different polyrhythmic patterns and provide feedback when the user taps along to the rhythms correctly. To enhance the concept of rhythm, we even added an animated giraffe (hence, DJ Raffe) that passed across the screen once every rhythmic measure.

My role in developing DJ Raffe lay in designing and implementing the interface. In order to create a system that accurately modeled rhythm, the interface’s timing functionality was vital and required a more accurate time perception than Javascript’s setInterval() or setTimeout() functions could provide. Luckily, other developers had run into similar issues before and I was able to solve the problem using and enhancing methods they had found useful.