Chip Kragt
616-528-8140 | chip@kragt.me | 1001 Alexander St SE, Grand Rapids, MI 49507

Ruby on Rails Applications

I have been designing, deploying and maintaining Ruby applications for non-profit agencies since 2011. From full-stack MVC apps to lightweight APIs, I have broad experience building and customizing Ruby applications in a variety of environments. I have worked with clients in all phases of development from design to deployment and beyond for upgrades and feature implementations.

This is a comprehensive list of the applications I have deployed and a brief overview of their capabilities and requirements. I also added code comments in several of my repositories to highlight specific code that I think speaks to my creativity and capabilities. A list of these code samples can be found in the HELLO INTERVIEWERS document on the Chip Overflow code repository. Where allowable, I have provided links to the production applications.

MediaHitTracker:

Client: American Red Cross of Greater Grand Rapids
Purpose: Record and report on media contacts, interviews, appearances and published articles. Allow users to communicate quickly and broadly with multiple news outlets at once. Provide time-scoped reports to guide media engagement strategies and identify trends.
Implementation: September, 2011
Tech stack:

- Ruby on Rails 3.1
- MySQL and PostgreSQL databases
- Puma webserver
- Rails Admin interface for forms
- Custom report pages with ActiveRecord SQL
- Javascript, jQuery for UI components
- Bootstrap CSS framework
- Heroku hosting (AWS SaaS service)

ResponseReady:

Client: American Red Cross of Greater Grand Rapids
Purpose: Track disaster response assets spread across 6 counties in a flexible inventory system that accounts for asset readiness, maintenance timelines, expiration dates and supply staging needs. Allow users to perform inventory functions via mobile-friendly web interface.
Implementation: November, 2012
Tech stack:

- Ruby on Rails 3.2
- PostgreSQL database with polymorphic relationships
- Unicorn webserver
- Rails Admin interface customized to leverage polymorphic associations
- Mobile-first web design with barcode scanning capabilities for inventory control
- Devise authentication system with customizable roles
- CoffeeScript, JavaScript, jQuery for UI components
- Bootstrap CSS framework
- Heroku hosting (AWS SaaS service)

TimeShift:

Client: Refugee Education Center
Purpose: Employee and volunteer hours tracking system specifically customized to meet State and Federal grant reporting guidelines. The application featured intuitive invoice-like time tracking and customizable, flexible reporting capabilities.
Implementation: January, 2014
Tech stack:

- Ruby on Rails 4.0
- PostgreSQL database
- Devise authentication customized to the client's domain and permission sets
- CoffeeScript, JavaScript, jQuery for UI components
- Heroku hosting (AWS SaaS service)
- ActionMailer and Mailgun for outgoing email processing
- RSpec and capybara test suite

PromiseTasks:

Client: Family Promise of Grand Rapids
Purpose: Lightweight project planning system for generating and tracking tasks related to purchasing, securing, inspecting and remodeling homes for the Partners in Housing program. The clients needed a simple, succinct way to generate specific recurring tasks to help them reliably, accurately and quickly purchase and prepare homes for homeless families. Since the clients already utilized Google Workspace applications, I wrote a lightweight API integration with Google Tasks. The clients can create and manage "auto tasks" that are auto-generated and assigned every time a task list is created in Google Tasks.
Implementation: April, 2016
Tech stack:

- Ruby on Rails 4.2 (upgraded through Rails 6)
- OAuth 2.0 authentication with Google Workspace accounts
- Materialize CSS and JS framework for simple front-end interface that mirrors Google Tasks UI
- Dedicated background job and real-time synchronization framework using Delayed::Job and PostgreSQL
- ActionMailer and MailGun for outbound email

LitersTracker (https://track.20liters.org):

Client: 20 Liters
Purpose: A program reporting and storytelling application for 20 Liters that allows program staff in Rwanda to submit data reports and staff in the U.S. to create micro-stories from those data reports. The application is designed to communicate to supporters how frequently program service delivery occurs and help program staff track implementation goals. The donor-facing system utilizes proficient querying and caching to render large quantities of images and text very quickly while staying affordable and small-scale.
Implementation: January, 2019
Tech stack:

- Ruby on Rails 5.2 (upgraded through Rails 6)
- PostgreSQL database with complex relationships and polymorphic associations
- Customized data reporting flow utilizing polymorphic assignment of geographical attributes
- Image uploading and editing features using ActiveStorage and ImageMagick
- AWS S3 Bucket integration for image storage
- Devise and Warden authentication system
- Pundit policy-based authorization system
- ActionMailer and SendGrid for outbound email
- Railway hosting (AWS SaaS service)

FilterBuildScheduler (https://make.20liters.org):

Clients: 20 Liters, Village Water Filters, Business Connect

Purpose: Initially a custom event scheduling and registration system for 20 Liters, this application has grown into a monolith that includes intelligent inventory management, donor CRM and Google Workspace API integrations, and payment record synchronization between multiple services.

This system allows 20 Liters to schedule and register volunteers for program events, manage program leadership volunteer assignments, synchronize event participant information with their donor CRM, record donations made through a 3rd party social fundraising platform into their CRM, and track donor email communications automatically within their CRM. Additional features include intelligent volunteer and inventory reporting, low-inventory alerting, automated inventory extrapolation, and real-time, multi-user inventory system synchronization.

I was able to provide two other organizations with their own instance of the base-level event, registration and inventory systems in their own custom deployments.

Implementation: November, 2016

Tech stack:

- Ruby on Rails 5 (now Rails 6.1, in process of upgrading to Rails 7)
- Devise and Warden authentication system
- Pundit policy-based authorization system
- Heavily customized Rails Admin interface
- Sidekiq and Redis background job system
- Websockets via ActionCable for real-time multi-user inventory synchronization
- ActionMailer and SendGrid for outbound email
- Puma webserver
- AWS S3 bucket for image and file storage
- Mobile-first UI with Bootstrap CSS, Chartkick, FontAwesome and Turbolinks
- RSpec and Capybara test suite focused on model, service and system tests

Chip Overflow (https://www.chipoverflow.com):

Purpose: My portfolio site, designed to show off my Ruby on Rails abilities while also serving as an informal resume and Q&A style written interview.

Implementation: October, 2022

Tech stack:

- Ruby on Rails 7 (latest)
- Stimulus.js and Turbo (using ImportMaps) for UI
- ActionMailer and SendGrid for outbound email
- MiniTest and Capybara test suite
- Tailwind CSS
- Custom rake tasks for database record seeding
- Gravatar integration for identicon generation
- Stack Overflow site scraping to capture my current reputation score and badges