Michael Musgrave
mmusgrave@pobox.com || 208 Second Street East, Pelham, New York 10803 || 914-255-0445
Github || Linkedin || Portfolio

EDUCATION

University of Michigan – Ann Arbor Ann Arbor, MI
Cognitive Science (BS), Concentration in Computer Science 2012 - 2016
Curriculum Highlights: Statistics & Data Analysis, Elementary Programming Concepts, Discrete Mathematics, Programming and Data Structures, Data Structures and Algorithms, Computer Organization, Database Management Systems, User Interface Development, Foundations of Computational Science, Technical Communication

App Academy New York City
Full Stack Web Development Course, acceptance rate < 3% 2017
Teaches Rails, React, TDD, scalability, algorithms, OOP, coding style, single-page apps, and web development best practices over 1000 hour course

SKILLS

Ruby, Ruby on Rails, RSpec, JavaScript, ES6, jQuery, React, Flux (Redux), AWS, SQL, C/C++, Git, HTML5, CSS3, UNIX, Postgres, SPSS, MATLAB

PROJECTS

Futbolistic (Ruby, JavaScript, HTML5, CSS3, PostgreSQL, AWS) Live | Github
Sports media website with open publishing model and subscriptions feed. Frontend written in Javascript, React, Redux and uses a RESTful API written in Rails.

· Developed CRUD features for React components through a Redux unidirectional data flow for reliable DOM rendering and structural coherence.
· Leverages CSS3 and React to create dropdown menus and use the hover attribute for optimal user experience and navigation
· Secures authentication by hashing and salting passwords with BCrypt Ruby gem.
· Utilized postgreSQL join tables and ActiveRecord through associations to simplify relevant data retrieval.

Travelling Salesman Problem (C++) Github
· Programmed Kruskal's algorithm to create a Minimum Spanning Tree among nodes Travelling Salesman Program.
· Implemented a solution to the Travelling Salesman, using the 2-Opt optimization to improve time complexity

Processor, Cache, and Pipeline Simulator (C) Github
· Engineered the assembler for a computer processor as well as a simulator to run the assembler.
· Incorporated advanced programming techniques and hardware concepts to program versions of the simulated processor that utilized pipelining and cache systems.