EMAIL: jason@richmond.is ~ TEXT: 574.855.6954 ~ SITE: jason.richmond.is ~ DATE: 2024.09.27

* Software Engineer with a Master's in Computer Science familiar with a diverse array of languages and platforms and a love for teaching seeking a return to the art of cultivating programming knowledge in the next generation

DEVELOPMENT SKILLS

LANGUAGES:
- Javascript
- Typescript
- HTML/CSS
- Python
- Rust
- Go
- Swift
- C
- C++
- C#
- Java
- SQL
- Assembly
- Supercollider
- CSound

METHODOLOGIES:
- CI/CD
- TDD
- Agile
- Scrum
- Kanban

TOOLS:
- Node
- React
- Vue
- Storybook
- p5
- Okta
- GraphQL
- MongoDB
- PostgreSQL
- Git
- Docker
- Mocha
- Hadoop
- Apache Pig
- Exasol
- Alluxio
- Jira
- GPTs

DOMAINS:
- UI/UX Design
- Full-stack Development
- Microservices
- REST
- Machine Learning
- Neural Networks
- AI

EXPERIENCE

SOFTWARE ENGINEER — 2021 ~ 2023
Aunalytics

* Maintained the microservices and REST API of our data solutions platform written in Node using MongoDB, GraphQL, Hadoop, and Apache Pig
* Achieved subject matter expert in Formations, our in-house data portability framework
* Contributed to initiatives in improve the robustness and fault-tolerance of our data pipeline
* Committed features that sped up our data delivery by an order of magnitude allowing us to achieve our on-time delivery goal of 75 consecutive days
* Took the reins on implementing two-phase procedure for data manipulation so that only valid results would be written to the destination
* Investigated and coded dynamic solution to a logging failure impacting our deliverables
* Pushed for and piloted new team structure to better communicate and increase collaboration
* Engaged in designing our next generation platform written in Typescript using React

LEAD INSTRUCTOR — 2018 ~ 2020
South Bend Code School

* Crafted interactive learning path spanning eleven lessons of around 25k words in p5, giving students an introduction to procedural, object-oriented, and functional programming paradigms
* Laid a concrete foundation for primary and secondary school students to build out abstract programming concepts using Scratch, HTML, CSS, JavaScript, C#, and Python
* Entrusted with running the Elkhart branch and being liaison to local schools keeping relevant stakeholders happy and extending Code School reach

LEARNING FACILITATOR ~ Computer Science — 2016 ~ 2019
Academic Center For Excellence

* Equipped dozens of graduates and undergraduates of all levels having trouble grokking the theory and practice of Computer Science with the knowledge and skills to succeed
* Debugged hundreds of student-written programs, usually on a tight deadline before submission without reference to a working answer
* Collaborated with professors to help compress the complex world of code into the tangible everyday for entry-level students

ACADEMIC EXPERIENCE

MASTER OF SCIENCE ~ Computer Science — 2021
Indiana University South Bend — GPA: 3.7

* Studied a wide spectrum in the discipline, from artificial intelligence to algorithm analysis, networking to neural networks, graphics to games, even writing the opcodes for a simulated CPU to run a puck-like robot with enough AI to navigate a maze

~ * ~

```python
# GENERATE TEXT RESUME FROM DATA ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#
import json
from collections import namedtuple
from datetime import date as d

data = json.load(open('data.json'), object_hook=lambda d: namedtuple('X', d.keys())(*d.values()))
letters = json.load(open('ascii.json'))
info, ed, work, craft, cl, gut, cr, t, sp = data[0], data[1], data[2], data[3], 31, 5, 75, 2, ' '
text, date, full, dev = '', d.today().strftime('%Y.%m.%d'), cl + gut + cr, craft.dev
deg, g = f'{ed.grad.degree.upper()} ~ {ed.grad.major.title()}', 'gpa: '

def display_name(n, letters, char, italic=True): # display name in ascii characters
  s, lines = '', []
  for line in range(len(letters[' '])):
    lines.append('')
  for ch in n.upper():
    for line in range(len(letters[ch])):
      for l in letters[ch][line]:
        lines[line] = f'''{lines[line]} ''' if l == sp else f'''{lines[line]}{char}'''
      lines[line] += sp
  for i in range(len(lines)):
    x = len(lines[i])-1
    while lines[i][x] == sp: x -= 1
    s += (sp*(len(lines)-i) if italic else '') + lines[i][:x+1] + '\n'
  return s

def bullet(s, mx, dent): # generate bullet
  a, s, i = [], sp*dent+'* '+s, 0
  while len(s) > mx:
    i = mx
    while s[i] != sp: i -= 1
    a.append(s[:i])
    s = sp*dent+sp+s[i:]
  a.append(s)
  return a

def bullets(arr, mx, dent): # generate bullets
  a = []
  [a.extend(bullet(s, mx, dent)) for s in arr]
  return a

def skills(obj): # generate skills text
  a = []
  a.append(f'''{obj.title.upper()+':'+(cl-len(obj.title)+1-t)*sp}{gut*sp}''')
  [a.append(f'''{t*sp}{n+(cl-len(n)-t)*sp}{gut*sp}''') for n in obj.names]
  a.append(cl*sp+gut*sp)
  return a

def jobs(emp, sub=False): # generate work text
  a, subject = [f'''*{(cr-2)*'~'}*'''], f'''{emp.role.upper()}{' ~ '+emp.sub if sub else ''}'''
  a.append(f'''{subject}{(cr-len(subject)-len(yrs := f'{emp.start} ~ {emp.end}'))*sp}{yrs}''')
  a.extend([f'''  {emp.name.title()}'''] + bullets(emp.text, 71, 2))
  return a

info_fields = f'EMAIL: {info.email} ~ TEXT: {info.phone} ~ SITE: {info.site} ~ DATE: {date}'
full_column = ['\n', display_name(info.name, letters, '/'), '']
full_column += [f'''{(full-len(info_fields)-7)*sp}{info_fields}\n\n*{(full-2)*'~'}*''']
full_column += bullets(info.text, 103, 8) +[f'''\n*{(full-2)*'~'}*''']
left_column = [f'''{craft.name.upper()}{(cl-len(craft.name))*sp}{gut*sp}''', f'''*{(cl-2)*'~'}*{(gut)*sp}''']
left_column += skills(dev.lang) + skills(dev.meth) + skills(dev.tool) + skills(dev.doms)
right_column = [f'''{(cr-len(f'{work.name}'))*sp}{work.name.upper()}''']
right_column += jobs(work.aun) + jobs(work.sbcs) + jobs(work.ace, True)
right_column += ['', f'''{(cr-len(f'{ed.name}'))*sp}{ed.name.upper()}''', f'''*{(cr-2)*'~'}*''',
f'''{deg}{(cr-len(deg)-len(ed.grad.year))*sp}{ed.grad.year}''',
f'''  {ed.grad.school.title()}{(cr-len(ed.grad.school)-len(g)-len(str(ed.grad.gpa))-2)*sp}''' +
f'''{g.upper()}{ed.grad.gpa}'''] + bullets(ed.grad.text, 71, 2)

for line in full_column: # print text
  text += line + '\n'
leftright = zip(left_column, right_column)
for line in leftright:
  text += line[0] + line[1] + '\n'
text += f'''\n\n{(full//2-len('~ * ~')//2)*sp}~ * ~\n\n\n'''

open('seeking.txt', 'w').write(text)
# THE END ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~#
```