Zachary Loveless

Software Developer and IT/Systems Administrator

Contact: Zack (He/Him) – zack.loveless@live.com – https://zloveless.com

SKILLS

- Languages: C#/.NET, Python 3, Bash, Batch, SQL, TypeScript
- Tools: Git, Docker, Docker Compose, Ansible, Hypervisors (Proxmox)
- Frameworks: ASP.NET, .NET Core/Framework, React, React Toolkit (RTK)
- Cloud platforms: Amazon Web Services (AWS), Cloudflare
- Operating systems: Windows, Linux (Debian, Fedora)

WORK EXPERIENCE

Part-Time System Administrator Totem Arts (Remote) July 2020 – present

- Automated deployment and provisioning of all infrastructure using custom Ansible roles.
- Consult with team members for ongoing updates to team processes.
- Dockerized all team services and created custom Ansible role to deploy across virtual machines.

Software Developer Drift Glass Ventures (Remote) February 2024 – September 2024

- Developed a double-entry accounting financial modeler in Python.
- Created a React application in TypeScript for viewing simulation results from a custom financial modeling program.
- Integrated all components into a single-page application including running python in the browser with web assembly.

Computer Lab Aide Arizona Western College (Yuma, AZ) August 2008 – June 2023

- Delivered tier one / frontline support to customers and aided them in achieving success in an academic environment.
- Managed over 140 machines and assisted hundreds of students per day administer their accounts.
- Performed troubleshooting and debugged lab hardware and faults.

Web Developer (Contract) S2 Systems (Remote) March 2019 – October 2019

- Implemented designer wireframes for websites in JavaScript, C#, and HTML on Amazon EC2.
- Overhauled CSS development process to use reusable SCSS snippets.
- Maintained a multitenant website project in ASP.NET Core with senior developer.

EDUCATION

- BASc Public Administration, Northern Arizona University, Yuma, Arizona. December 2021

SOFT SKILLS

- Solid organizational skills for collaborating with coworkers and across teams.
- Excellent writing skills with a focus on argumentative, technical, and summary writing.
- Strong analytical and reasoning skills for effective decision making and problem solving.

PORTFOLIO

Home Lab / Self-Hosting 2010 – present Mixed privacy

- **Technologies**: Ansible, Docker, Apache/Nginx, MySQL/MariaDB, PHP, Proxmox
- **Sources**:
  - https://github.com/zloveless/ansible-common
  - https://github.com/zloveless/ansible-containers

At home and in personal-use self-hosting, my *Lab* is rather small. I maintain a couple Proxmox servers to have space to spin up virtual machines as needed. My current hosting requirements are minimal as I host a simple Pihole to provide network-based ad-blocking for my entire home and a small Jellyfin (media library) instance. I also host a Foundry VTT install at home, which is accessible through Cloudflare Tunnels and provides a virtual tabletop for playing and running tabletop games. On rented servers, I host a GameSpy master clone for C&C Renegade, a third-person shooter set in the C&C universe.

RxCmd 2014 Open Source

- **Technologies**: C#/.NET (Framework), Managed Extensibility Framework (MEF)
- **Source**: https://github.com/zloveless/RxCmd

RxCmd is a command-line utility that interacts with a Renegade X game server's RCON-like (remote console) interface. The team behind Renegade X developed a rudimentary remote console interface for moderating game servers, so to learn it, I wrote a console app that let you connect to an instance of a game server, subscribe to logs, and add bots. The structure was modular, allowing commands to be loaded at startup and executed later in the console. The basic commands included wrappers around fetching basic information and subscribing to game logs for testing an IRC regulator.

Financial Modeler 2024 Closed Source

- **Technologies**: Python, TypeScript, React, HTML, CSS/SASS, Git
- **Link**: Link TBD

At Drift Glass Ventures, I built a financial modeler in Python that simulates the cashflow of an idea the user has. The model gave the user a better picture of where obvious shortfalls get highlighted as errors in the model output. The modeling software includes both a python library and command-line component, as well as a React website. The python library allows the user to install the python library locally in their own environment, while the React website provides a web editor using Monaco (embedded VScode) to create a financial model. The most difficult problem that I faced was in implementing the editor's autocomplete which at the time of implementation is not as complete as a regular VScode with a proper language server (LSP) for Python.