**Jeffrey Ray**

Software Engineer

**Contact**

**Location:** Toronto, Ontario, Canada (Open to Remote)

**Phone:** (647) 221-1703

**Email:** [jeffrey.ray@bell.net](mailto:jeffrey.ray@bell.net)

**LinkedIn:** <https://www.linkedin.com/in/jeffrey-ray-29271b30a/>

**GitHub:** <https://github.com/jray-8>

**Portfolio:** <https://jray-8.github.io/portfolio>

**Education**

**Bachelor of Computing, University of Guelph (2020–2024)**

Computer Science | Mathematics (minor)

**Certificates**

**Statistics with Python Specialization** – University of Michigan

**Statistics for Data Science Essentials** – University of Pennsylvania

**SQL for Data Science** – University of California, Davis

**Technical Skills**

**Languages:** Python, SQL, C, C++, JavaScript, CSS, HTML, LaTeX, Markdown, MATLAB

**Python Libraries:** Pandas, Numpy, Matplotlib, Seaborn, Plotly, BeautifulSoup, Requests, Pillow, Scipy

**Development Tools:** Git, GitHub, Jupyter, Excel, MySQL, PostgreSQL, VS Code, Docker, Kubernetes, Linux

**Mathematics & Algorithms:** Combinatorics, Set Theory, Graph Theory, Linear Algebra, Calculus, Probability, Statistic, Divide and Conquer, Dynamic Programming, Hashing, Big O Notation, Complexity Analysis, Wrappers

**Visual resume:** <https://jray-8.github.io/resume/Jeffrey_Ray_SE_Resume.pdf>

**Projects**

**U.S. Presidential Dataset – May 2025**

* Engineered a **Python data pipeline** to scrape, clean and maintain U.S. presidential datasets, automating **ETL workflows** and producing versioned **CSV outputs** for analysis.
* Applied **statistical analysis** techniques, including **Welch’s t-test**, to evaluate inaugural age trends; generated **data visualizations** using **Pandas**, **Seaborn**, and **Matplotlib** to support **data-driven insights**.
* Crafted **Jupyter Notebooks** demonstrating **exploratory data analysis (EDA)**, **hypothesis testing**, **statistical modeling**, and **historical trend analysis** with stakeholder-focused documentation and reproducible reporting.

**Boids – February 2025**

* Developed an interactive **physics-based simulation** in **JavaScript** and **HTML5 Canvas**, modeling decentralized **flocking behavior** through **alignment**, **cohesion**, and **separation algorithms**.
* Implemented **multiple** **flock types** with distinct parameters (**speed**, **vision radius**, **turn factor**, **cohesion**, **alignment** and **separation weights**), enabling exploration of **emergent behaviors** in a complex system.
* Built a **real-time control panel** for dynamic tuning of **flock properties**, improving usability and experimentation.
* Added advanced features such as **anchor-based attraction**, **swarming**, **solo-mode visualization**, and a **toggleable collision system** to support user-driven analysis of **flock dynamics**.

**Poker Hand Analysis & Strategy Simulation – February 2025**

* Modeled probabilities of **5-card poker hands** using **combinatorics** and **statistical analysis**, presenting derivations in **Jupyter Notebooks** with neat mathematical formatting using **LaTeX**.
* Designed and evaluated fair prize systems, then built **simulation experiments** to optimize **strategy under uncertainty**, leveraging **expected value analysis** and **decision theory**.
* Created **Matplotlib visualizations** to illustrate **probability distributions**, **prize payouts**, and **convergence of simulation results**.

**Mandelbrot Set Explorer – January 2025**

* Developed and deployed a **JavaScript web application** for interactive **fractal exploration** across the **complex plane**, leveraging **Web Workers** for **parallel computing** via **data parallelism**.
* Implemented **algorithmic optimizations** (eliminating costly math operations like **sqrt()**, minimizing **object allocations**, reducing **iteration overhead**) to achieve **deeper zoom levels** and **high-performance rendering**.
* Showcased **low-level optimization skills** and ability to identify and address **performance bottlenecks** in **compute-heavy applications**.

**Duel-Life – January 2025**

* Designed a browser-based **stochastic cellular automaton (SCA)** inspired by **Conway’s Game of Life**, simulating war between two agent groups across structured terrain.
* Applied **probabilistic state-transition rules** and **dynamic color-mapping** to model **strategic patterns**, **territorial expansion**, and stable **stronghold positions**.
* Investigated applications of randomness in **agent-based modeling**, **complex systems**, and **simulation-driven experimentation**.

**Genetic Algorithm Package – December 2024**

* Authored a series of interactive **Jupyter Notebooks** solving original **genetic algorithm (GA)** challenges, culminating in a reusable **Python GA package (Pyvolver)**.
* Implemented and experimented with GA components including **encoding schemes**, **fitness evaluation**, **crossover** and **mutation operators**, **selection methods**, **replacement strategies**, and **termination conditions** to analyze their effects on optimization outcomes.
* Designed a flexible **framework** for applying GAs to any **optimization**, **combinatorial** or **permutation** problem.
* Built a **browser-based GA visualizer** with real-time evolution control (e.g., **mutation rate**, **fitness scoring**) to demonstrate **exploration–exploitationtrade-offs** and convergence dynamics.

**Image Processing Toolbox – Spring 2023**

* Built a modular **REPL-based image editing suite** in Python using **Pillow (PIL)**, supporting **custom convolution kernels**, **histogram equalization**, **pixel-intensity visualizations**, **pointwise transformations**, and **non-linear filtering**.
* Engineered original, **psychedelic visual effect**s including **channel-splitting distortions**, **glitch-art pixel sorting**, and **ghost split**.
* Implemented a wide range of transformations—**rescaling, color manipulation**, **filter design**, **geometric warps**, and **blending/overlay techniques**—highlighting skills in image processing algorithms and **mathematical creativity**.

**Online Chatroom & File-Sharing Server – Summer 2021**

* Architected a multi-user **chat network** with **Python**, **sockets**, **multithreading**, and a **custom message protocol** supporting **headers**, **user authentication**, and **reliable communication**.
* Built a **color-coded terminal UI** on top of the **ncurses library**, enhancing **user interaction** and real-time display of messages.
* Designed a **command-parsing system** with a secondary socket for **concurrent server–client operations**, including **serialized file transfer**, **private messages**, and **network error handling**.

**Graphic Maze Generator – Winter 2021**

* Implemented a randomized **Depth-First Search (DFS)** algorithm in **Python** to construct unique mazes with guaranteed solvability and variable complexity.
* Developed a responsive **GUI** with **Pygame** which permits **panning**, **scrolling**, **zooming**, **in-game play**, and **exporting** **mazes** as printable images.
* Enhanced **user experience (UX)** with custom menus for **key bindings** and **maze personalization** (color themes, dimensions, borders, day/night UI modes).