ZAHIN M. ZAMAN

**TECHNICAL SKILLS**

[](https://devpost.com/alvii147)[](https://www.linkedin.com/in/zahin-zaman/)[](https://github.com/alvii147)[](mailto:zm2zaman@uwaterloo.ca)

**Open-Source Software Developer**

*codePrentice | Sept 2020 – Present*

* Streamlined **Python** code flow for open-source **multiparty-computation** project **MP-SPDZ**
* Expanded machine learning computation to support **Tensorflow SqueezeNet** model

**Display Verification Engineer**

*Qualcomm | Jan 2020 – May 2020*

* Engineered **SystemVerilog assertions** and **C++** simulations tests to verify display processor design
* Attained **20%** increase in functional coverages by debugging waveform using **Synopsys Verdi** tool
* Automated **formal verification** using **Perl** scripting to extract design hierarchy and formulate assertions
* Web-scraped design database and employed **PyQt5** to build interactive GUI for managing hardware registers

**Embedded Software Developer**

*Imagine Communications | May 2019 – August 2019*

* Reconstructed source code in **C/C++** on a **Linux** environment to fix firmware bugs
* Extracted IP routing data from data structures and developed troubleshooting functions and mapping tables
* Utilized **SoapUI** to inspect and track **REST API** processes

**CAN Interfacing Team Member**

*WATonomous | Jan 2019 – April 2019*

* Developed **Python** code in **ROS** framework for car’s lock and turn signals and inertial navigation system driver
* Enhanced low-speed **CAN** interfacing system of the car and performed simulation in **Virtual CAN Driver**

**EDUCATION**

**University of Waterloo**

*B.A.Sc. in Electrical Engineering, 2B | Sept 2018 – May 2023*

* **Term Dean’s Honour List,** for outstanding academic performance
* **President’s Scholarship of Distinction,** for 95%+ admission average

**Software Development & Testing Engineer**

*Wind River Systems | Sept 2020 – Dec 2020*

* Rectified multi-threading and memory-based defects for **VxWorks** RTOS and **Helix Virtualization Platform**
* Debugged and reconfigured source code in **C** and utilized inline **Assembly** to write thread-safe functions
* Formulated **git hook** to block restricted commits and echo terminal warning
* Developed **Python** script with interactive GUI using **PyQt5** to assist in git commit message format

**EXPERIENCE**

**Programming:** C, C++, Python, HTML, CSS, Javascript, Perl, SystemVerilog, VHDL, ARM Assembly

**Tools & Frameworks:** Linux, Windows, Git, Tensorflow, Keras, scikit-learn, OpenCV, Django, Flask, PyQt5

519-721-2837

zm2zaman@uwaterloo.ca

devpost.com/alvii147

linkedin.com/in/zahin-zaman

github.com/alvii147

**Smart Wardrobe**

*Python, TensorFlow, Pillow, Matplotlib, PyQt*

* Modelled multi-layered neural network with **TensorFlow** and **Keras** to categorize pieces of clothing
* Extracted image data using **Pillow** and visually presented training data accuracy with **Matplotlib**
* Optimized cost function based on **mean-squared error** by implementing **stochastic gradient descent**
* Incorporated **multi-threading** in Python to develop interactive user interface with **PyQt**

**Ubisoft Game Dev Challenge**

*C++, SFML, HackerNest API*

* Employed **Ubisoft’s HackerNest API** and **SFML** library in to develop adventure minigame
* Executed realistic physical movement by installing smooth sprite animation
* Enhanced gameplay and mechanics by incorporating precise game physics

**Digital Piggy Bank**

*C++, Raspberry Pi Zero W*

* Constructed coin-identifying piggy bank with lasers and photodiodes using **Raspberry Pi Zero W**
* Assembled circuit to read analog voltage using operational amplifier and differential comparator
* Implemented **state machines** and **watchdog timers** to program and monitor state of lasers

**Spoilers Alert**

*Python, SciPy, Universal Sentence Encoder*

* Incorporated **natural language processing** tools to construct spoiler detection program for popular TV shows
* Restructured text into vectors using **Google’s Universal Sentence Encoder** to assess semantic similarity
* Implemented **least-squares regression** using **SciPy** with **sigmoid function** model to execute binary classification

**PROJECTS**