




BARDIA PARMOUN

✉ bardiaparmoun@gmail.com  [bardiaparmoun](https://www.linkedin.com/in/bardiaparmoun)  [bardia-p](https://github.com/bardia-p)  [bardia-p.github.io](https://github.com/bardia-p.github.io)

EDUCATION

Carleton University

Bachelor of Software Engineering (Co-op)

Sept. 2019 – PRESENT

Ottawa, Canada

- **GPA:** 11.89/12 (3.99/4.0) | **Graduation Date:** 2024/04/30 | **Awards:** Dean's List and Carleton Academic Scholarship
- Teaching assistant for Computation and Programming, Data Management, and Real Time Concurrent Systems.

SKILLS

Programming Languages: Python, Java, C, C++, Swift, JavaScript, Bash, SQL, PHP, Racket, Visual Basic

Technologies: Linux, SwiftUI, Spring, HTML/CSS, Bootstrap, jQuery, Node, Express, EJS, Mongo, PostgreSQL, MySQL, Docker

Tools: Git, Perforce, Jira, Azure, ROS2, Vim, GDB

EXPERIENCES

Apple

Software Engineer - Wireless Technologies & Ecosystems

June 2023 – PRESENT

Montréal, Canada

- Working as part of the Tap to Pay on iPhone team.

Microsoft - The Coalition Studio

Software Engineer Intern - Rendering

June 2023 – Aug. 2023

Vancouver, Canada

- Implemented a hard drive benchmarking tool for **The Coalition Studio's** custom **Unreal Engine 5** using **C++**.
- Integrated two upscaling frameworks into the engine for evaluation and added a custom console variable for them.
- Helped develop a custom wind generator tool for the rendering team using **C++** and **HLSL**.

Apple

Software Engineering Intern - Wireless Technologies & Ecosystems

May 2022 – Dec. 2022

Montréal, Canada

- Developed internal tools for **Tap to Pay on iPhone**, using **Swift, SwiftUI, and Objective-C**.
- Participated in the verification of a Tap to Pay on iPhone framework and its new features for **iOS 16.0**.

BlackBerry QNX

Core OS Software Development Student

Jan. 2022 – April 2022

Ottawa, Canada

- Performed various unit tests for the **QNX kernel** and **filesystem**, using **C** and **gcov**, achieving **100%** coverage.
- Designed and conducted **regression tests** for the **/proc filesystem**, using **C** and BlackBerry's testing API, following **Automotive SPICE** to ensure the customer needs are being met and the components are behaving as expected.

Ross Video

Software Developer - softGear

May 2021 – Dec. 2021

Ottawa, Canada

- Built two internal testing tools for the softGear team, AES67 Player and Recorder, to convert **.raw audio** to **AES67 streams** and generate custom tones, using **C++, JSON, HTTP, and Docker**; improved softGear's testing capacity by **20%**.
- Accelerated the release of two of softGear's products, **RSAP** and **NWE-IP**, by resolving bugs and adding new features.

PROJECTS

🔗 **Carleton Mail Delivery Robot** | Python, ROS, Java, Spring, HTML/CSS, JavaScript, Azure

April 2024

- Designed a **ROS** based system to deliver mail in the Carleton University tunnels using programmable roombas.
- Applied principles of agent based programming and developed thorough state machine for the system.

🔗 **Opinion Owl** | Java, Spring, HTML/CSS, JavaScript, Azure

Dec. 2023

- Created a clone of the Survey Monkey website using **Java and Spring** and hosted it on **Azure**.

🔗 **Real-Time Elevator Simulator** | Java, UDP, C-LOOK, State Pattern, XML, Serialization, Swing

April 2023

- Simulated an elevator control system in **real-time** using **Java threads** following the **state design pattern**.
- Utilized **UDP** to connect the components and implemented the **C-LOOK** algorithm to schedule the elevator requests.

🔗 **LIBER: Online Bookstore** | PostgreSQL, HTML/CSS, PHP, Apache

Jan. 2022

- Implemented an online bookstore website using **PHP and PostgreSQL** that allows users to order books online.
- Designed and implemented the database in **PostgreSQL** with proper triggers, functions, views, and procedures.

🔗 **Embedded Pong** | C, Python, Pygame, UART, GPIO, Serial

Jan. 2022

- Built a controller for Pong game using **Python** and the **MSP432R** board through the **UART protocol**.
- Wrote a program in **C** to configure the LEDs, switches, and ports with the proper GPIO and interrupts.