# Bharavi Misra

+1 (484)-925-8392 | bharavimisra@gmail.com | linkedin.com/in/bharavi-misra | bmisra03.github.io

## EDUCATION

**The Pennsylvania State University** University Park, PA
Bachelor of Science in Computer Engineering. Cumulative GPA: 3.94/4.00. August 2021 – May 2025
*Relevant Courses: Operating Systems, Computer Architecture, Data Structures and Algorithms, Computer Networking and Security.*

## SKILLS

**Programming Languages:** Java, C, C++,Python, Golang, JavaScript, MATLAB, SQL, R, Verilog, Assembly.
**Developer Tools/Frameworks:** Git, Docker, Maven, TeamCity, Linux, Bash, GDB, PyTorch, Confluence, Jira.

## WORK EXPERIENCE

**MathWorks** Natick, MA
Software Engineer Intern May 2024 – August 2024
- Wrote automated tests for merge and deployment in Go for AI backend, augmenting code coverage by 50%.
- Designed scalable infrastructure for containerizing binaries and virtualization automation with Docker and Kubernetes, ensuring service reliability with 100,000 concurrent requests.
- Enhanced TeamCity CI/CD pipeline to calculate test statistics and enforce minimum code coverage in GitHub pull requests by refactoring Maven configurations, writing bash scripts, and calling GitHub's REST API, increasing visibility by 25%.
- Documented code and testing strategies in Confluence, planned work using Jira, and presented updates in scrum meetings.

**Siemens Digital Industries Software** State College, PA
Software Engineer Intern May 2023 – August 2023
- Automated monthly library updates for Siemens' PLM Vis using graph traversal algorithm on file system, reducing testing time by 89%, from 90 minutes to 10 minutes, working in an agile environment to deliver high-quality software.
- Designed webapp using d3.js to visualize and implement histogram-based comparison of rendered models in PLM Vis.
- Wrote unit tests for 29 2D file types supported by PLM Vis using JavaScript.
- Installed, validated, built, and pushed updates for 2 external dependencies of Vis desktop using C++ and Linux terminal.

## LEADERSHIP

**Penn State Advanced Vehicle Team** University Park, PA
Applied Research Department Head January 2025 – May 2025
- Identified, articulated, and assigned object detection/classification projects among 5 students based on technical strengths, providing technical guidance to ensure deliverable milestones were met.
- Developed localization pipeline using Python, leveraging publishers, subscribers, and topics to enable real-time data exchange across distributed sensor nodes, improving localization accuracy by 60% in GNSS blackout.
- Migrated low-level autonomous vehicle state machine code from Python to C++, added concurrent read/write support for CAN message buffers, and implemented multithreading, reducing latency by 75%.
- Fine-tuned lane line detection models in PyTorch, increasing detection accuracy by 50%, and reducing latency by 25%.
- Designed infrastructure to translate lane line detections into control decisions, enabling closed-loop autonomy.

## PROJECTS

**Concurrent Channel Implementation**
- Engineered a thread-safe channel in C, enabling blocking/non-blocking operations with POSIX threads and semaphores.
- Optimized synchronization, achieving error-free results under 10,000 parallel requests using Valgrind,sanitizers, and GDB.
- Implemented efficient queue communication, preventing race conditions, deadlocks, and CPU waste.

**Dynamic Memory Allocation Implementation**
- Wrote low-memory footprint implementations of malloc, realloc, and free in C, improving memory utilization by 50%.
- Conducted design space exploration in Python to calculate optimal segregated free list based on block size distributions.
- Designed a heap checker for debugging and performance/memory tracking, optimizing hardware resources.

**Course Scheduling System**
- Implemented a Java-based course scheduler app with GUI using Java 2D framework and object-oriented design principles.
- Developed backend for dynamic capacity, waitlists, and atomic enrollments using a MySQL relational database to ensure data consistency across >10,000 simulated operations with 100% integrity.
- Optimized query performance and data access layers, reducing average enrollment transaction latency by 40%.