# Sowmith Kunapaneni

sowmith.kunapaneni@wsu.edu | sowmithk.com | Pullman, WA

## Education

**Washington State University** — Pullman, WA
*PhD in Computer Science* — *Aug 2024 – Expected May 2028*
**Relevant Courses**: GPU Programming, Database Systems, AI, Programming Languages, Computational Genomics

**Velagapudi Ramakrishna Siddhartha Engineering College** — Vijayawada, India
*Bachelor's in Electronics and Communication Engineering* — *Aug 2016 – Sept 2020*

## Experience

**Graduate Research Assistant** — Aug 2024 – Present
*HARP Lab, Washington State University* — *Pullman, WA*

- Transferred to WSU to continue my work under Dr.Thomas Gilray on deductive databases and **HPC research**.
- Working on a batching approach to **Out-Of-Memory joins on the GPU.**
- Submitted work for memory optimized datalog GPU backend to ***International Conference on Supercomputing 2025***

**Graduate Research Assistant** — Aug 2023 – Aug 2024
*HARP Lab, University of Alabama at Birmingham* — *Birmingham, AL*

- Research assistant for Programming Languages and HPC research.
- Learning about Control Flow Analysis, Functional Data Structures and **Parallel Datalog engines**.
- Worked on Slog Lang; A MPI based shared memory deductive database engine, paper accepted at ***VLDB 2025***.

**Systems Engineer** — Oct 2020 – Dec 2022
*Tata Consultancy Services* — *Hyderabad, India*

- **Built internal CLI tooling** to improve support workflow and automate tasks.
- Implemented log management and monitoring tools, reducing failure rates and improving SLA.
- Developed comprehensive dashboards for tracking and monitoring job health, leveraging logs and various job-specific parameters.

## Projects

**Batching Joins on GPU** | A GPU Datalog Engine | *C++, CMake, CUDA* — July 2024 – Present

- Algorithm for batching Relational Joins on GPU to avoid OOM constraints.
- Built a **GPU Datalog engine** that supports join, project and union operations.
- Builds on top of the GPU optimized hash based indexing structure from GDlog adding support for splitting chained binary joins into batches.

**Brouhaha's Compiler & Run-Time** | *C++, Racket, CMake* — May 2023 – Nov 2023

- A full program analyzing compiler for sub-set of Racket
- Wrote a **C++ Run-Time** to support primitives and data structures
- Integrated Boehm Garbage Collector and GNU MP Big Num Lib into the Run-Time
- Worked on a Hash Array Mapped Trie implementation to support 'hash' and 'set' prims.
- Wrote a tree-based index over a hash table to **enable efficient query of analysis** results by the compiler.
- https://github.com/harp-lab/brouhaha

**Multi Threaded Maze Game Server** | *C++, Python* — Aug 2023 – Dec 2023

- Renders frames of a 2D Maze-game simulation, communicating with an agent over stdin/out
- Implemented Radial Sweep for agent vision, multi-agent support, and game object interactions.
- Wrote reference agents that use A*, D*Lite and Flood Fill algorithms.
- https://github.com/harp-lab/maze-game

## Technical Skills

**Programming Languages**: C++, C, Python, CUDA, Racket, Go, Datalog, HTML & CSS
**Developer Tools**: Git, Docker, Clang, Linux,Unix Core Utils, LSP, Vim, CLion, Tree-sitter, Gem5