

SHIBALI MISHRA

[Portfolio](#) | +1 (207) 798-9384 | smishra@bowdoin.edu | [LinkedIn](#)

EDUCATION

Bowdoin College, Brunswick, ME

Bachelor of Arts, Computer Science and Mathematics Major

May 2026

Aquincum Institute of Technology, Budapest, Hungary [*Study Abroad*]

January 2025 – May 2025

Relevant Coursework: Distributed Systems, Operating Systems (*Spring 2026*), Mobile Software Development, Cryptography, Multivariate Calculus, Data Structures, Linear Algebra, Artificial Intelligence, Data Science, GIS, Ordinary Differential Equations

Activities: Bowdoin Women in Computer Science Leader, Career Center Peer Advisor, Grace Hopper Conference 2023 & 2025, HackHarvard 2023, CodePath 2025, Bowdoin Hackathon 2025 Finalist, Bowdoin Career Bootcamp Leader

SKILLS & INTERESTS

Languages: Python, Java, Kotlin, C, HTML, CSS, JavaScript, SQL

Frameworks & Web: React, Tailwind CSS, FastAPI, Uvicorn, Jinja2, SendGrid, Azure SQL, LangChain, PostgreSQL

Tools & Systems: AWS S3, Visual Studio, Android Studio, ArcGIS, OpenAI API, MapReduce, Hadoop (HDFS/YARN), XML-RPC

Concepts: Client server architecture, concurrency & synchronization, RESTful APIs, environment-based deployment

AI & Data Science: GPT APIs, GitHub Copilot, K-Means clustering, Folium, Matplotlib

WORK EXPERIENCE

WinePair LLC, Connecticut, USA

May 2025 – August 2025

Founding Software Engineering Intern

- **Led end-to-end development and production deployment** of a custom full-stack platform, migrating the system from Wix to a **Python Uvicorn/FastAPI backend**, an **Azure SQL** database, and a **React/Tailwind** frontend.
- **Engineered GPT-4 simulation and server-side caching** to precompute pairing results, reducing end-to-end latency from **30 s to <3 s** and improving throughput under real user load.
- **Built production-grade ingestion pipelines** for 10+ restaurant menus, extracting and normalizing **1,000+ dishes** with deduplication, validation, and integration into a **scalable Azure SQL schema**.
- **Improved system reliability** by redesigning backend routing, error handling, and UI integration, increasing successful pairing completions and reducing failed requests.

ForeFlight (Boeing), Portland, Maine, USA

May 2024 – August 2024

Software Engineering Intern

- **Refactored legacy C++ scripts into Python** for FAA Parachute Jump Area data, integrating with **AWS S3**, implementing **unit tests**, and handing off automated workflows to the data team on **TeamCity**.
- **Automated Temporary Flight Restrictions (TFR) data extraction** using a custom **Python web scraper**, storing results in a **PostgreSQL** database with real-time **XML/HTML parsing**.
- **Designed an SQL-based spatial query** to filter light polygons intersecting with **OpenStreetMap** place labels, eliminating false positives from uninhabited illuminated areas and improving urban boundary classification accuracy.

PROJECTS

Bowdoin Network Plus (Python, FastAPI, LangChain, Google Sheets API)

- Led a **four person team** to design an intuitive frontend and **LangChain driven matching system** that suggests alumni and generates personalized cold email templates based on shared interests and career paths.
- Built a **full-stack AI networking platform** that parses student **resumes** and recommends relevant alumni from live **Google Sheets** data, creating a faster and more intuitive career-networking experience.
- Delivered a **finalist prototype** at the **2025 Bowdoin AI Hackathon** and collaborated with the **Career Development Office** to expand it into a campus-wide professional mentoring tool.

Honeycomb: Distributed P2P Disaster Messaging Network (Python, XML-RPC, AWS EC2)

- Built a **distributed system** for **lightweight, low-bandwidth** disaster communication using a Chord-inspired ring (SHA-1 IDs) with **minimal per-node state** and background maintenance to self-repair under churn.
- Implemented **fault-tolerant message dissemination** via **message-ID deduplication** and **probabilistic forwarding** to create redundant paths without network-wide flooding.
- Ran controlled experiments on a **70 node AWS EC2** testbed, injecting node failures and measuring **convergence time** and **delivery reliability** to quantify **reliability trade-offs** under failures and load.