Ahmad Qureshi

+49 176 28674059 | ahmadqureshi386@gmail.com | linkedin.com/ahmadq | github.com/CodeInCachemire

Portfolio: **ahmadq.me**

Summary

Computer Science and Cybersecurity student at Universität des Saarlandes focused on backend and fullstack development. Practical experience with Python, FastAPI, Django, React, PostgreSQL, Docker, AWS, and CI/CD, including REST APIs, authentication, database-backed logic, and deployed web applications. Interested in contributing to practical software solutions in a Praktikum or Werkstudent role in Backend Engineering, Full Stack Development or AI Engineering.

Education

Universitaet des Saarlandes, B.Sc. Informatik & B.Sc. Cybersicherheit — Apr 2023 – August 2027

Technical Skills

Programming Languages: Python, JavaScript/TypeScript, Java, C, Kotlin, Assembly, HTML, CSS
Frameworks & APIs: FastAPI, React, Django REST Framework, REST APIs, Pydantic, AI/LLM API Integration
AI & Agents: Multimodal LLM Integration (Gemini, Claude), Tool Calling & Agent Loops, Schema-Constrained Output, MCP, Google ADK
Databases: PostgreSQL, SQLite, Redis
Cloud & Infra: AWS (EC2, RDS, S3), DigitalOcean, Vercel, Docker, Nginx, Cloudflare
DevOps: GitHub Actions, CI/CD, GitHub, GitLab, Gradle, n8n
Testing & Observability: Pytest, JUnit, Unit Testing, Integration Testing, Sentry

Experience

Tutor for Mathematics for Computer Science II — Apr 2024 – Oct 2025

Universität des Saarlandes

- Led weekly tutorial sessions for 40 students across two semesters, covering **linear algebra** and discrete mathematics with a focus on proof techniques and algorithmic thinking.

Software Design Engineering Praktikum — Aug 2024 – Oct 2024

Universität des Saarlandes

- Completed a selective 7-week practicum with criteria-based progression across qualification, group, and individual phases, meeting every functional and code-quality threshold to advance through all stages.

Group Phase (4 weeks, team of 6)

- Co-designed system architecture for a **multi-agent event-driven simulation** using **UML** class, sequence, and state diagrams; defended architectural decisions in two formal design reviews with the Chair of Software Engineering.
- Implemented core **MVC** controller logic supporting complex multi-agent behavior across the simulation.
- Led the team testing strategy using **JUnit** and structured logging; passed **98/100** server-provided functional tests and **23/25** mutation tests to advance to the individual phase.
- Enforced code quality through Git workflows, cross-team reviews, and static analysis with Detekt within an **agile, Scrum-based** team process.

Individual Phase (3 weeks)

- Extended the system via core simulation controllers and handlers under incomplete specifications, implementing **event-driven** state-based logic using State and Command design patterns.
- Enhanced **JSON scenario parsing** with additional schema validation and robust error handling.
- Passed final evaluation: **48/50 tests**, **9/10 mutants**, full code coverage, and a formal design and code review.

Projects

Distributed Sudoku/SAT Solver Platform | **Live Demos:** Sudoku Solver | Distributed Sudoku Solver | SAT Solver

- Deployed a live fullstack platform with vanilla JS/HTML/CSS frontends and a **FastAPI** backend, exposing a **CDPLL SAT solver**, a synchronous Sudoku solver, and a distributed async Sudoku solver with real-time queue visibility.
- Designed a **four-layer backend architecture** (API, service, data, solver) with **FastAPI dependency injection**, **PostgreSQL**-backed idempotency, and separated business, persistence, and solver logic.
- Architected a **Redis**-backed distributed job queue with atomic job claiming, Dockerized workers, and dashboard-style visibility into asynchronous job states, making backend data flows and worker behavior observable in real time.
- Built an automated ingestion pipeline for external puzzle data, including parsing, validation, persistence, and backend integration for puzzles from **New York Times Sudoku.**
- Engineered a multimodal AI extraction pipeline where user-uploaded puzzle images are processed by **Gemini 2.5 Flash** and converted into structured, schema-validated outputs using **Pydantic**.
- Deployed on **AWS** (EC2, RDS, S3) with **Nginx** reverse proxying, **GitHub Actions** CI/CD, and Sentry observability.

Agentic Coding Assistant | Python, React, FastAPI, Anthropic API, MCP, Google ADK

- Built a fullstack AI coding assistant with a **React** frontend and **FastAPI** streaming backend, enabling real-time display of agent responses as they stream from the model.
- Implemented an **agent loop** over Claude for multi-turn task execution, combining codebase navigation, controlled file edits, command execution, and iterative feedback handling.
- Re-architected the system with **Google ADK**: a root orchestrator routes tasks to specialized sub-agents for code execution, SAT solving, and IBAN validation, isolating tool access and responsibilities per agent.
- Designed tool-scoped agent workflows with structured outputs and validation, focusing on transparent model behavior, controlled tool access, and inspectable execution steps.

MIPS IBAN Validator-Generator | **Live Demo:** MIPS IBAN Validator

- Built and deployed a live fullstack application: IBAN validation core written in **MIPS Assembly**, exposed via a **FastAPI** service with a vanilla JS/HTML/CSS frontend, **Pydantic** input validation, privacy masking, and persistent conversion history in **SQLite**.
- Dockerized and deployed on **AWS** with automated testing via **GitHub Actions** CI/CD.

Fame-Based Social Network | Django, Django REST Framework

- Built fullstack web application features end-to-end using **Django** and **Django REST Framework**, covering backend logic, ORM queries, and REST API design.
- Implemented a reputation-driven content moderation pipeline: post submission triggers fame evaluation, automatic adjustment on negative truth ratings, and cascading user banning with bulk post unpublishing.
- Designed a non-symmetric **user similarity algorithm** ranking users by shared fame levels across expertise areas, with ranked negative reputation queries across the full user base.

Languages

English: Native Proficiency (C2) | **German**: Professional Working Proficiency (telc C1)