Jordan Doe

555-010-2000 | jordan.doe@example.com | linkedin.com/in/jordan-doe | github.com/jordan-doe | jordandoe.example.com

EDUCATION

Riverside State University Riverside, ST
Bachelor of Science in Computer Science *May 2026*

Honors/Awards: Winner — 2026 Riverside Innovation Challenge (Widgetize), Dean's List (4 semesters)
Related Coursework: Algorithms, Database Systems, Applied Statistics
Certifications: Data Analytics Certificate (Coursera – May 2026)
Organizations: Code Collective, Data Science Club, Women in Computing

TECHNICAL SKILLS

Languages: Python, TypeScript, Go, SQL
Frameworks: React, Node.js, FastAPI, Docker
Data: PostgreSQL, Redis, Pandas

EXPERIENCE

Software Engineering Intern Remote
Acme Cloud Systems *Jun 2026 – Aug 2026*

- Migrated a legacy deployment pipeline to containerized infrastructure, cutting release time from 45 minutes to 6 minutes, by scripting a Docker Compose workflow and automating health-check gating.
- Reduced API p99 latency 38% under peak load, by profiling the request path and adding a read-through cache layer in front of the primary database.
- Closed a test-coverage gap on the billing module from 52% to 94%, by writing integration tests against a seeded staging environment.

Technical Lead Riverside, ST
codecollective.example.com *Sep 2025 – Present*

- Cut new-member onboarding time from 3 weeks to 1 week as measured by time-to-first-contribution across a 10-person team, by designing a structured onboarding track and a buddy-pairing system.
- Standardized project handoff process across 4 concurrent teams as measured by zero lost-context incidents in the following semester, by authoring a shared handoff template covering scope, blockers, and next steps.

PROJECTS

Widgetize | *React, Node.js, PostgreSQL* | widgetize.example.com Feb 2026 – Present

- Built a full-stack inventory dashboard serving 40+ concurrent users, by designing a normalized 6-table schema and a role-based access layer.
- Cut page-load time from 4.2s to 900ms, by adding server-side pagination and lazy-loading chart components.
- Shipped an automated CI pipeline with 90% test coverage, by writing integration tests against a seeded test database and gating merges on a green suite.

DataLoom | *Python, Pandas, DuckDB* | dataloom.example.com Sep 2025 – Dec 2025

- Built a data-cleaning pipeline processing 500K+ rows of survey data, by implementing schema validation and automated outlier flagging.
- Reduced manual review time 70%, by generating an automated summary report with column-level anomaly counts.
- Documented the pipeline's assumptions and edge cases, by writing a data dictionary referenced by 3 downstream analyst teams.