Abdullah Khan

437-488-8659 | abdullah.khan1@uwaterloo.ca | khanzai.vercel.app | linkedin.com/in/khanzai | github.com/khanuzai

Education

University of Waterloo — Waterloo, ON
Bachelor of Computer Science (BCS) — *Expected 2029*

Wilfrid Laurier University (Double Degree) — Waterloo, ON
Bachelor of Business Administration (BBA) — *Expected 2029*

Technical Skills

Languages: Python, JavaScript, TypeScript, Java, SQL, C/C++, Racket
Frameworks & Libraries: React, Node.js, Express.js, FastAPI, Tailwind CSS, Material UI
Tools & Technologies: Git/GitHub, PostgreSQL, MongoDB, Docker, AWS (Lambda, S3, RDS), Azure, Vercel, JIRA, Figma

Experience

Software Developer — Milton, ON
PixelsBoost — *Sep. 2025 – Dec 2025*

- Delivered **5 full-stack client websites** to production using React, HTML/CSS, and JavaScript, implementing dynamic contact forms, interactive image galleries, and mobile-responsive navigation serving **2,000+ monthly users**.
- Architected and integrated **3 third-party APIs** including Stripe for payment processing (**$15K+ monthly transactions**), Google Maps for geolocation, and SendGrid for email automation, maintaining **99.5% uptime** through error handling.
- Boosted website performance by **42%** through WebP image compression, lazy loading, and Cloudflare CDN implementation, improving Google Lighthouse scores from **68 to 87** and reducing bounce rate by **18%**.
- Managed **5 concurrent client projects** using Git/GitHub, making **150+ commits** across **25+ feature branches** while collaborating with 2 designers and resolving **12+ merge conflicts**.

Software Engineering Intern — Missouri, USA (Remote)
Fast Webs — *May 2024 – Aug. 2024*

- Developed **12 production UI components** for 3 web applications using React and Node.js, building checkout flows, user dashboards, and admin panels with component-based architecture.
- Built comprehensive event tracking system to capture user interactions (clicks, submissions, navigation) and store behavioral data in PostgreSQL database, enabling product team to identify **3 UX improvements**.
- Optimized application performance by **35%**, reducing bundle size by **120KB** through React code splitting and lazy loading, measured using Chrome DevTools and webpack bundle analyzer.
- Completed **18 development tickets** across **12 Agile sprints** in Jira, delivering **11 feature implementations** and **7 bug fixes** while documenting **6 technical processes** for team onboarding.

Projects

Attack Surface Growth Simulator (ASGS) [GitHub] | *Python, React, FastAPI, SQLAlchemy, NumPy, SQLite, Recharts, Pydantic*

- Built full-stack security risk modeling tool that calculates attack surface scores (**0-100**) across **5** threat categories by normalizing system metrics (endpoints, users, MFA adoption, vulnerabilities) and generating ranked driver breakdowns.
- Developed **FastAPI** backend with **Pydantic** validation that computes quadratic risk functions and calculates derivatives using **NumPy** to identify unsafe growth zones where system complexity creates disproportionate security exposure.
- Designed **SQLite** database with **SQLAlchemy** ORM to persist assessment configurations and results, enabling users to compare different security scenarios and track how architectural changes impact overall risk posture.
- Built interactive **React** dashboard with **Recharts** that visualizes risk curves, growth rates, and danger zones in real-time as users adjust system parameters, helping teams identify critical inflection points before scaling.

AutoForm (Spur Hackathon) [GitHub] | *React, Vite, OpenRouter API, pdf-lib, Tailwind CSS*

- Developed AI-powered form filling application using **React** and **OpenRouter API** to automatically populate PDF forms with user profile data, processing **5+ form types** including tax forms, rental applications, and visa documents.
- Engineered PDF processing pipeline using **pdf-lib** and **pdfjs-dist** to detect and fill form fields programmatically, achieving accurate field mapping across PDFs with varying complex layouts while maintaining complete document integrity.
- Implemented error handling for API rate limits, missing fields, and malformed PDF structures; built real-time status indicators displaying field completion states with **copy-to-clipboard** functionality for seamless user export.