Vansh Khanna

+1 765 701 9288 | khanna57@purdue.edu | LinkedIn | GitHub | West Lafayette, US

Education

Purdue University West Lafayette, US

B.Sc. in Computer Engineering; *Aug 2020 – May 2024 (Expected)*

Relevant coursework: Advanced C Programming, Data structure and algorithms, Digital circuit design, Python for Data Science, Object Oriented Programming with C++, Operating Systems,Microprocessor Systems and Interfacing, Networks, Compilers, Artificial Intelligence, Encryption and Secuirty, Signal Processing

Skills

Programming: C, C++, Java, Python, MySQL, Angular, React, Node.js Assembly, Rust, MATLAB

Technologies: Git, Arduino, STM32, Rest API, API Six, Linux, Simulink, LTspice

Languages: English (Professional), Hindi (Native)

Developer Tools: GitHub, Linux, BootStrap, VS code, Docker, Postman, Jenkins, Apache Maven, Pandas, NumPy, SciPy, PyTorch

Work Experience

Ernst & Young New Delhi, India

SWE Intern *Jul 2023 – Aug 2023*

- Participated in the comprehensive development of a Single Sign-On (SSO) solution, utilizing JAVA and APISIX technologies.
- Collaborated seamlessly with cross-functional teams, eliciting and analyzing requirements to architect SSO functionality.
- Developed robust and secure APIs Transport Layer Security (TLS) to facilitate smooth authentication and authorization processes, ensuring optimal user interactions.
- Helped with the seamless integration of the SSO functionality with diverse citizen centric applications, centralizing user access control and bolstering data security.
- Rigorously conducted testing, debugging, and performance tuning to guarantee the resilience and efficiency of the SSO implementation.

TestBook.com New Delhi, India

Frontend Developer Intern *May 2022 – Aug 2022*

- Developed Frontend using Angular and JS.
- Developed Rest API and integration with backend.
- Worked in Agile Environment with active participation in weekly sprint planning and daily scrum calls
- Enhanced page performance through techniques like lazy loading, asset loading, and client-side caching.
- Active team member of static and dynamic content distribution planning using Cloud flare CDN.
- Developed and deployed Configuration management system using Git.

Projects

Multi threaded C++ Server

- Designed, implemented, and optimized a multi-threaded server.
- Leveraged advanced C++ programming techniques.
- Implemented TCP to ensure seamless client-server interactions, fostering efficient data exchange.
- Conducted rigorous testing and debugging to identify and rectify potential bottlenecks and synchronization issues, resulting in a stable and reliable server architecture.

Developed a Linux distro

- Optimized memory management with one and two-level paging.
- Implemented multi-threading for concurrent process execution.

- Developed a dynamic task scheduling algorithm.
- Integrated trap handling for error management.
- Designed a comprehensive file system module for data storage and retrieval.

Developed Full stack Employee Information Management System

- Backend: Java-based API for efficient communication.
- Frontend: Dynamic React UI for data management.
- Database: MySQL for secure, scalable data storage.
- RESTful endpoints: Real-time data updates.

Movie finder web app | *Website*

- Frontend: Designed an intuitive Angular-based UI for seamless movie searches.
- Backend: Implemented a robust Node.js server with RESTful APIs for data retrieval.
- Data Integration: Utilized external APIs like IMDB for real-time movie information.

Compiler

- Scanner and Lexer Implementation: Designed and coded a highly efficient scanner and lexer, allowing the compiler to tokenize source code accurately and efficiently, forming the foundation for subsequent compilation phases.
- Abstract Syntax Trees (ASTs): Developed a robust AST framework that faithfully represented the syntax and structure of source code, enabling precise analysis and manipulation of program logic.
- Code Generation Engine: Engineered a versatile code generation module that translated ASTs into optimized intermediate code and further into target machine code, demonstrating proficiency in code optimization techniques.
- Control Structure Handling: Implemented support for a wide range of control structures, including conditionals, loops, and branching, ensuring proper code execution and control flow in compiled programs.
- Function Management: Integrated comprehensive function support, including function prototypes, parameter handling, and return values, enabling the compilation of modular, reusable code blocks.
- Register Allocation Algorithm: Designed and implemented an advanced register allocation algorithm, effectively managing CPU registers to enhance the runtime performance of compiled programs.
- Pointers and Arrays: Extended the compiler's capabilities to include pointers and arrays, empowering the manipulation of complex data structures in high-level languages.
- Error Handling and Reporting: Implemented robust error-handling mechanisms and user-friendly error reporting, enhancing the overall usability and debugging experience of the compiler.

Plant Monitor Dashboard

- Responsive React UI Development: Engineered an intuitive and responsive user interface using React, ensuring a seamless experience across various devices for monitoring plant parameters.
- RESTful API Integration: Implemented robust REST APIs to seamlessly connect the React dashboard with an online MySQL database. Facilitated efficient data retrieval and updating functionalities for real-time monitoring.
- Real-time Data Visualization: Designed and integrated dynamic charting components to visually represent sensor data in real-time. Empowered users with instant insights into plant conditions for proactive decision-making.
- Downloadable Data Capability: Implemented a feature-rich system allowing users to download historical data easily. Enhanced the dashboard's utility by providing users with the ability to analyze trends and patterns over time.
- Secure Database Management: Ensured the security and integrity of the MySQL database by implementing robust authentication and authorization mechanisms. Prioritized data privacy and protection against potential vulnerabilities.
- Cross-functional Collaboration: Collaborated with cross-functional teams, including sensor engineers and data scientists, to understand requirements and integrate diverse functionalities seamlessly.

Certificates

Decoding DevOps Udemy *Aug 2023*

Google Tensor Flow *Dec 2022*