Mohamed Sayed Ali

Full-Stack Web Developer (Node.js, React.js)

+20 1140933253 | mohamedsaid0573@gmail.com | github.com/MohamedSayed0573 | linkedin.com/in/mohamed-sayed3
Cairo, Egypt

Summary

Computer Science student with hands-on experience across the stack — frontend with React and TypeScript, backend with Node.js, Express, and PostgreSQL. Built and shipped a cross-browser extension now used by **4,000+ active users.**

Education

Zagazig University — Al-Sharqia, Egypt
Bachelor of Computer Science — *Expected Graduation: 2027*

- **Academic Standing**: Excellent (GPA: 3.5)
- **Relevant Coursework**: Data Structures & Algorithms, OOP, Database Systems, Computer Networks, Compiler Design

Experience

Summer Training Program (1 Month) — Aug 2026
Information Technology Institute (ITI) — *Zagazig, Egypt*

- Completed a 1-month training program covering **Frontend Development using React.js**.

Technical Skills

Languages: JavaScript, TypeScript, C++, SQL

Backend Technologies: Node.js, Express.js, Fastify, Socket.IO, REST APIs

Frontend Technologies: HTML, CSS, React.js, Tailwind

Databases / Caching: PostgreSQL, Redis

Cloud / DevOps: Docker, AWS, CI/CD, Linux

Projects

TubeSize | *Cross-Browser Extension (Chrome, Firefox, Edge)* — GitHub | Chrome Store

- **4,000+ Active Users**
- Built a **browser extension** (TypeScript, React) that tracks per-site bandwidth usage and estimates video/livestream sizes on YouTube, Twitch, and Kick; published on Chrome, Firefox, and Edge stores.
- Engineered a fully client-side attribution pipeline combining `chrome.webRequest` header inspection with a page-world `fetch`/Worker patch, removing a `yt-dlp` server dependency and cutting metadata latency to **35ms**.
- Reverse-engineered undocumented streaming APIs and parsed HLS playlists to estimate per-quality data cost for videos, Shorts, and livestreams

ChessLab | *Real-Time Multiplayer Chess Platform (React, Node.js)* — GitHub | Live Demo

- Built a **full-stack real-time chess platform** with online multiplayer, matchmaking, and synchronized gameplay using **React**, **Express**, and **Socket.IO**.
- Integrated the **Stockfish** chess engine using **WebAssembly** and **Web Workers** for responsive single-player gameplay without blocking the main thread.
- Engineered a **server-authoritative backend** that validates every move, manages per-player chess clocks, and handles resignations, draw offers, disconnects, and reconnection windows with no trusted client logic.

Languages

Arabic: Native
English: C1