Darshan Sharma

• thedarshansharma@gmail.com • +91-700-974-6321 • darshan.sh

INFO

Full Stack Developer and Associate Cloud Engineer with **6** years of experience in designing and implementing scalable web applications

EDUCATION

B.E in Computer Science (Panjab University) - 2018

EXPERIENCE

Software Engineer at MonkAI **Dec 2019 - Jun 2024**

Improved React video player load time in the app feed by pre-loading just 25% of the content, fetching the rest only after 90% viewership of the initial segment, guided by user data analysis
TS: TypeScript, React, Node, AWS-Lambda, S3, MongoDB

Reduced DB expenses by 10% by integrating Redis as a caching layer, resulting in a single definitive database call rather than 6-7 heavy mongodb calls.
TS: React, Nest, AWS-RDS, GraphQL, TypeScript

FullStack Developer at Paxcom **May 2019 - Nov 2019**

* Decreased the load on the node server backend as it was being used for heavy CPU computations using a load balancer and distributed equal load on 2 separate microservices.

* Achieved a 27% decrease in server expenses by transitioning from a monolithic architecture to deploying numerous microservices, coupled with the integration of automated scaling and efficient traffic management.

Software Engineer at Block8 **May 2018 - May 2019**

* Optimized a Node.js web application's performance by refining database queries, significantly improving runtime efficiency

SKILLS

- **Web:** HTML5, CSS3
- **Languages:** JavaScript, TypeScript, Python
- **Frameworks:** NodeJs, NestJs, ExpressJs, React
- **Database:** MongoDB, PostgreSQL
- **Tools:** GCP, AWS, Docker, CI/CD, Git

PROJECTS

- **Toyota:** Toyota Thailand website in React-Redux and Express with huge inventory management
  Tech Stack: React, Typescript, Express, AWS, Jest, Git

- **MyStake:** Share-trading application on geth, React, MongoDB, Nest and enhanced React-Node app speed by 25% by optimizing database queries and utilizing CDN for static assets.
  Tech Stack: React, Typescript, Nest, AWS, Jest, Graphql, Docker

- **Paxcel Website:** Reduced node server load by offloading heavy CPU computations (image processing, CSV to HTML conversion) using multithreading and microservices,enhancing server performance by 27% and created microservices and deployment to aws
  Tech Stack: React.Js, Typescript, Node, GCP, Kubernetes

- **Block8 website:** At Block8 we were using React for the front end and Node for the back end. The database queries, which were written inside Lambda functions, were not optimized, performing the cross-operation first and then the selection operation. I tweaked it a bit to perform the selection operation first, followed by joins, and then the final selection. In this way, we were able to reduce some latency.

- **MonkAI:** Improved React video player load time by 17% using CloudFront for caching, resulting in better TTFB.