Personal Profile

I am a hard-working individual with confidence in software design and development, troubleshooting and working in teams. I am highly passionate and motivated about computer science, and often teach myself various different concepts in my spare time. I have designed and developed projects in various different fields, including creating servers in multiple server software languages such as Python, NodeJS and Go. I am also experienced in low-level high performance languages such as C++ and Rust. I have taught myself a wide range of these skills within computer science, and have worked in teams with other software developers.

Skills

- Highly experienced in Typescript, Go, Python, C#, Java and Rust.
- Highly confident in server-side REST API frameworks such as NodeJS + Express, Golang, Ruby on Rails, and Python Flask/Django.
- Highly confident in SQL/NoSQL databases such as MySQL, SQLite (SQL), and MongoDB (NoSQL.)
- Highly experienced in advanced programming concepts such as object-oriented programming, asynchronous/concurrent programming and functional programming.
- I am able to design and implement complex solutions for various different issues that may arise during a project.
- Highly confident in cybersecurity and different user authentication methods such as JWT (JSON Web Tokens) and symmetric (AES)/asymmetric (RSA) encryption.
- I have extensive experience administrating linux systems, such as Ubuntu servers to work with server applications such as my own servers, or existing server builds such as MySQL.
- I have experience working with Docker, creating my own Docker images for my applications, and frequently use Docker with my own self-hosted PaaS (Caprover);

Projects

URL Shortener (MERN Stack + Typescript):
Project Description:
In this project, I use *Express* to create my own REST API which runs on the backend. This REST API is queried by the front-end (designed in *ReactTSX*). The REST API is used for either creating a new shortened URL, or querying the existing ones. When a shortened URL is created, it creates a new entry in the MongoDB database to maintain persistence. When a shortened URL is accessed, the server will reply with a 302 HTTP redirect page, redirecting the user to the corresponding URL.
Skills Demonstrated:
- MongoDB
- Express + Node
- React
- Typescript
- REST API

Repository Source Code: https://github.com/saifsuleman/BetterShortener

JSON Web Token Authentication Service (Go):
Project Description:
This project consists of me designing a JWT server that will serve an authentication server via a REST API endpoint, and a library written in Go to validate the token. I use asymmetric RSA encryption here so various different third-party services would not require the server's secret encryption key to validate the data. Instead, the server will sign the payload with it's private key, and the library will validate the token via the server's public key, which it will fetch from a REST API endpoint.
Skills Demonstrated:
- JWT Authentication

- Asymmetric RSA signing
- Go
- REST API endpoints

Repository Source Code: https://github.com/PlasmoidIO/qbit

Chat Application (Go + React):

Project Description:

This project is a chat application I developed in Go and React. We use websockets to facilitate communication between the backend and frontend. When you launch the backend application, it will automatically listen on port 8000, and the backend will serve the frontend on that port too. The application creates a websocket endpoint the frontend will access for the basic chat application. There is also a console panel on the backend for creating users, removing users, and kicking users. When I build the application, I use go embeds to embed the frontend resources into the main executable, allowing for a single executable that serves both the backend API and the frontend.

Skills Demonstrated:

- Go
- React
- REST API websocket endpoint

Repository Source Code: https://github.com/saifsuleman/gochat