



OVIDIU KALKALA KAMBEMBO

🌐 <https://scintillating-sunshine-df5c78.netlify.app/> |  <https://www.linkedin.com/in/ovidiu-kambembo-b19659347/> | ✉ ovidiukambembok13@gmail.com |  <https://github.com/OviKams13> | ☎ +905338679971

/ SKILLS

- **Programming** : C++, Java, Python, C#, Javascript.
- **Web & Database** : HTML, CSS, MongoDB, MySQL, SQL Lite, Php myadmin.
- **Frameworks** : React js, Express js, Next js, Django.
- **Tech** : Node.js, REST APIs, GitHub, Postman, Visual Studio, VSCode, Eclips.

/ EDUCATION

Bachelor of Software engineering (Year of Graduation : July 2026)

Relevant Coursework : Java OOp, Internet Programming, D.S. & Algorithm, Database Management Syst., Operating Syst., Software Design and Architecture.

/ PROJECTS

Finance Tracker Application. (Full-Stack)

<https://finance-tracking-app-rho.vercel.app/>

- Built with **Next js** for server-side rendering (**SSR**) and static site generation (**SSG**), ensuring fast load times.
- Implemented **Clerk authentication** for secure user login and registration, ensuring data privacy and role-based access control.
- Used **Drizzle ORM** for efficient database interactions with **PostgreSQL** for structured storage.
- Designed a responsive UI with **ShadCN** and **Tailwind CSS** to provide a sleek and user-friendly interface.

Workout Tracking Application. (Full-Stack)

<https://github.com/OviKams13/Workout-Tracking-App>

- Built a **RESTful API** using **Express js** and tested endpoints with **Postman** to ensure robust API functionality.
- Utilized **MongoDB** as the database to store exercise data efficiently with timestamps for tracking history.
- Designed a responsive and interactive UI using **React js** to display workout details and forms for creating or updating exercises.
- Implemented state management to dynamically update the UI upon creating or deleting a workout, ensuring real-time user feedback.

Real - Time Chat Application. (Back-end)

https://github.com/OviKams13/real_time_chatt-app

- Integrated **WebSockets** using **Django Channels** to enable bi-directional communication, ensuring live messages.
- Designed **database models** with **Django ORM** and **SQL Lite** to store room-specific chat history and user messages.
- Implemented a basic authentication mechanism to identify users in chat rooms.
- Utilized **ASGI** and **in-memory channel layers** for efficient handling of concurrent users.
- Enhanced application security by integrating **CSRF tokens** into forms to prevent cross-site request forgery attacks.