OBJECTIVE

To gain a position where my creativity, problem-solving abilities, and experience in Software Development can be used to improve productivity.

CONTACT

- daniellarbi947@gmail.com
- +233 (0)504243523
- +233 (0)542008013

EDUCATION

KWMAE NK. UNIVERSITY OF SCI. AND TECH
(TERTIARY, BSC. COMPUTER SCIENCE)
2018-2022

REFERENCE

Rita Okronipa
Lead IT Service Management
GCB Bank PLC
Phone: +233 (0)202007026
Email: reokronipa@gcb.com.gh

William Ofori Boadu
Head of IT Strategy and Service Management
GCB Bank PLC
Phone +233 (0)24 430 4593

PALMER DANIEL BEKOE LARBI

FULL-STACK DEVELOPER | SMART CONTRACTS, WEB & MOBILE APPS | BLOCKCHAIN SOFTWARE ENG. | dApp | Web3

I am an experienced Full Stack developer (7 years' experience) who specializes in the development of both client and server-side components of a web application or software system. I am responsible for creating the user interfaces, infrastructure, logic, and databases that power the functionality of a website or application, ensuring that it runs smoothly and efficiently.

EXPERIENCE

VERSIFIED TECHNOLOGY
2022-2023

I occupied the position as a backend developer. I was responsible for building and maintaining end points (APIs) for frontends to consume.

GCB BANK PLC (NATIONAL SERVICE)
2022-2023

- Helping workers troubleshoot little issues they encounter with their workstations during work hours.
- Routing of technical issues to specific IT units for resolving

---

LANGUAGES SPOKEN

- English

SKILLS

Programming Languages

- PHP
- JavaScript

Server-side frameworks(Backend)

- Node.js (Javascript)
- PHP

Web Frameworks (Client side / Frontend)

React.js

Next.js

Mobile App Development

React Native

Database Management

- SQL (Structured Query Language)
- NoSQL (MongoDB)

Blockchain

- Smart contracts in solidity for Ethereum
- DApp development with Truffle and Web3.js
- Defi & NFT solutions including token development

Event Driven Development

Microsoft Power Platform (Power Apps, Power Automate, Power Pages)

API Development

Creating or maintaining new and existing APIs.

Version Control

Experience with version control system (Git) which is essential for collaborative development and code management.

Security.

Having a strong grasp of security and best practices as a backend developer is essential where software development is concerned. Some practices include:

- Authentication and authorization
- Data encryption
- Input validation

Performance Optimization

Identifying and resolving portholes in codes and databases, and server configuration is critical for scalable application.

Testing and Debugging

Unit testing, integration testing and debugging are essential for backend developers.

Collaboration and Communication

Communication and collaboration with front-end developers, designers and other team members plays a key role for successful project outcomes.

Problem Solving

Facing complex technical challenges is very normal with backend development, as result developing a very good problem-solving character is necessary which I can.

Documentation

Documenting code, APIs and system architecture is essential for knowledge sharing and future development.

Understanding Designing Patterns.

Getting used to software design patterns can help in backend development.

Version management.

Staying up-to-update with language and framework updates and best practices is essential for backend development.

MAJOR PROJECTS UNDERTAKEN

EAGLE-MEMO Project (National Service Initiative)

- **Problem Identified**:
  - Memos were manually written on paper, making tracking difficult.
  - Security was weak — memos were often left on desks, exposed to unauthorized viewing.
- **Solution Delivered**:
  - Developed a digital application called **EAGLE-MEMO** to replace the manual memo system.
  - Enabled users to raise new memos digitally for approval directly within the app.
- **Key Features**:
  - **Secure Access**: Users can only view memos they've created or are involved in.
  - **Workflow Integration**: Line managers and stakeholders are assigned per memo.
  - **Privacy Focus**: Memo details are only visible to relevant users — ensuring confidentiality.
  - **Paperless Efficiency**: Entire memo lifecycle is now digital — from creation to approval.

Tech Stacks: PowerApps and Power Automate

T-PAY

• Implemented a loan application system utilizing React Native, MongoDB, React JS, React Native and Node.js technologies.
• Leveraged MongoDB for efficient storage and retrieval of borrower information, ensuring seamless data exchange and real-time updates.
• Utilized React JS and React Native to create responsive and intuitive interfaces that adapt seamlessly to various devices and screen sizes, enhancing the user experience.
• Employed Node.js for fast and scalable server-side applications, exchanging data between client and server.
• Implemented an innovative incentive program powered by MongoDB's data analytics capabilities, incentivizing responsible borrowing behavior and fostering customer loyalty.
• Offered a straightforward interface for users to navigate through the borrowing process with ease.
• Utilized advanced algorithms for swift loan approvals and disbursements.
• Ensured transparency by providing clear information on interest rates, repayment terms, and associated fees.
• Introduced a unique reward system for users who repay their loans on time, incentivizing responsible financial behavior.
• Designed the application to adapt to the evolving financial landscape, incorporating features such as personalized notifications and reminders for upcoming payments.
• Redefined the borrowing experience by combining convenience with responsible lending practices, benefiting both individuals and the lending ecosystem.
With MongoDB being a non-relational database helped in nesting some of the data of the users on the project
This is not fully ready yet. Still building some of the functional parts of it.

Tech Stacks: React Native, React JS, MongoDB and Node.js

TekHealth
https://github.com/dannymccall/tek-health.git,,,https://github.com/dannymccall/tek-health-backend.git

•Developed an app for accessing healthcare from home.
• Implemented user registration functionality.
• Provided three main services:
• Enabled online conversation with a doctor for immediate assistance.
• Facilitated appointment booking for future medical consultations.
• Offered the option to order blood type tests.
• Ensured user-friendly interface and seamless navigation.

Tech Stacks: React Native, React JS, MongoDB and Node.js

Savings App
https://github.com/dannymccall/accurate-ventures-susu-frontend/tree/master

• Developed a platform for a money-saving company to keep records of money collected from customers.
• Recorded the date of collection along with the amount.
• Implemented a messaging feature to send notifications when necessary.
• Administrator user privileges for adding users (employees) and monitoring and system.

Tech Stacks: React JS, MongoDB and Node.js

Inventory Management System

Asset Management System (Built with React & PHP)

- **Tech Stack**: React (frontend) and PHP (backend) for a responsive and dynamic system.
- **Purpose**: Designed to **track and manage assets** such as furniture and equipment.
- **Condition Monitoring**:
  - Assets are categorized as **functional** or **non-functional**.
  - Enables quick identification of items needing attention.
- **Repair Workflow**:
  - Non-functional assets are assigned to **technicians** for repair.
  - Technicians provide real-time status updates (e.g., completed, pending further action).
- **Admin Oversight**:
  - Admins receive immediate feedback on repair progress.
  - Facilitates proactive decision-making and improved accountability.
- **Outcome**:
  - Streamlined asset tracking.
  - Enhanced communication between technicians and administrators.
  - Increased transparency and efficiency in asset management.

Tech Stacks: PHP/MYSQL, React

FREELANCE PROJECTS

Business Software

Freelance Project – Sales & Inventory Management System

- **Tech Stack**: JavaScript (frontend interactivity), PHP & MySQL (backend and database).
- **Project Scope**: Built a **comprehensive system** to manage:
  - Sales and product inventory
  - Employee roles and access control
  - Expenses and dynamic reports via **Chart.js**
- **Core Features**:
  - **Authentication** with secure password management
  - **Product management**: add, update, and track products with stock alerts
  - **Invoice generation** and purchase tracking
  - **User & worker management** with role-based access
  - **Expense tracking** to monitor operational costs
  - **Report generation** with visual analytics
  - **Product expiration alerts** for inventory health
  - **Search functionality** for quick data retrieval
- **Impact**:
  - Replaced manual tasks with automation, improving operational efficiency
  - Delivered a **scalable** and **user-friendly** interface with smooth data handling
  - Ensured **data security** and reliable performance using PHP/MySQL backend

Tech Stacks: JavaScript, PHP/MYSQL

MICROFINANCE MANAGEMENT SYSTEM

The **Microfinance Management System** is a comprehensive application designed to streamline loan management, client tracking, and payment schedules for microfinance organizations. The system empowers administrators to efficiently manage loans, monitor repayments, and generate detailed financial reports.

KEY FEATURES

1. **Loan Management:**
   - Ability to create and manage loan applications.
   - Integration of loan details, including disbursement dates, loan terms, interest rates, and repayment schedules.
   - Automatic tracking of loan statuses (e.g., active, repaid, arrears, default).
   - Real-time loan approval notifications using **WebSockets** to notify administrators instantly.
2. **Payment Schedules:**
   - Dynamic generation of payment schedules based on loan terms.
   - Tracking of weekly or monthly payments, including amounts due, amounts paid, and outstanding balances.
   - Categorization of schedules into statuses such as "Paid," "Arrears," "Default," and "Outstanding."
3. **Client Management:**
   - A database of client profiles with personal information, loan history, and repayment behavior.
   - Ability to link loans and payment schedules to individual clients for easy tracking.
4. **Real-Time Notifications:**

    - Implemented **WebSocket** technology to deliver instant notifications to administrators for pending loan approvals or critical updates.
    - Ensures timely actions and enhances user experience.
5. **Reporting and Analytics:**
    - Generate detailed reports for repayments, arrears, and defaults within specific date ranges.
    - Summarized financial insights, including total amounts paid and outstanding balances.
    - Visual data representation using charts and graphs for better decision-making.
6. **Admin Dashboard:**
    - Centralized dashboard for managing loans, clients, and payments.
    - Role-based access control for secure management.
    - Notifications for overdue payments or upcoming due dates.
7. **Scalability:**
    - Designed to handle large datasets, ensuring smooth performance even with a growing client base.

Technologies Used

- **Frontend:**
    - Built using **Next.js** (App Router) with **TypeScript** for enhanced type safety.
    - **Tailwind CSS** for responsive and modern user interface design.
- **Backend:**
    - Powered by **Node.js** with a RESTful architecture for efficient API handling.
    - **MongoDB** (via **Mongoose**) as the database for handling structured and unstructured data.
    - Aggregation pipelines for advanced data analysis, including grouping, filtering, and lookup operations.
- **Real-Time Communication:**
    - Implemented **WebSocket** for live notifications to administrators, ensuring immediate actions on loan approvals.
- **Features for Scalability and Performance:**
    - Efficient data querying using MongoDB aggregation.
    - Role-based access control for security.
    - Optimized database structure for seamless data retrieval.
- **Version Control:**
    - Managed using **Git**, ensuring proper versioning and collaboration.

Key Contributions

- Designed and implemented the loan and payment schedule models with MongoDB.
- Developed a custom aggregation pipeline for generating filtered payment data by status (e.g., repayment, arrears, default).
- Integrated advanced lookup operations to join multiple collections (loans and clients) for comprehensive reporting.
- Built responsive and dynamic components in **Next.js**, ensuring a seamless user experience across devices.
- Created custom business logic for handling financial calculations, including outstanding balances and totals.
- Designed and implemented **WebSocket-based notifications** for real-time updates on loan approvals, improving admin efficiency.
- Implemented data visualization using modern tools for analytics and reporting.

VALUES DELEVERED

- Enhanced operational efficiency by automating manual processes, reducing errors, and saving time.
- Improved financial visibility and decision-making through robust reporting and analytics.
- Increased user engagement with a modern, responsive interface and streamlined workflows.
- Enabled real-time responsiveness with WebSockets, improving organizational communication and task resolution.

Skills Demonstrated

- **Frontend Development:** Next.js, TypeScript, Tailwind CSS.
- **Backend Development:** Node.js, Express.js, Mongoose.
- **Database Management:** MongoDB, Aggregation Pipelines.
- **Real-Time Communication:** WebSocket implementation for notifications.
- **Problem-Solving:** Developed algorithms for payment schedules and reporting.
- **Collaboration:** Version control with Git for team integration.
- **UI/UX Design:** Created intuitive, user-friendly dashboards and interfaces.

BLOCKCHAIN PROJECTS

KidsBank DApp

Solidity, React.js, Web3, Smart Contracts

Built *KidsBank*, a full-stack DApp that enables parents to create blockchain-based bank accounts for children, set allowances, control withdrawals, and monitor activity using Solidity smart contracts and a React.js frontend integrated with Web3.

- Designed and developed a decentralized banking application (DApp) for kids, allowing parents to create, manage, and monitor bank accounts for their children on the Ethereum blockchain.
- Implemented smart contracts in Solidity to securely manage deposits, allowances, and withdrawals with strict parent-child access controls and inactivity auto-deactivation.
- Built the frontend using React.js with Web3 integration, enabling real-time interaction with deployed smart contracts (account creation, deposit handling, withdrawal requests, and account status management).
- Enforced time-based constraints like monthly allowances and inactivity checks using blockchain timestamps to maintain financial discipline mechanisms.
- Emphasized security best practices (e.g., access control modifiers, reentrancy protection) to ensure safe fund management across the DApp.
- Focused on creating a user-friendly experience for non-technical users while maintaining on-chain transparency and traceability of all actions.

NFT Marketplace – Full-Stack Web3 Application

Tech Stack: Next.js · TypeScript · TailwindCSS · Web3.js · Truffle · Solidity · IPFS (Pinata) · Node.js · MongoDB

Overview

Developed a decentralized NFT marketplace using **Next.js** and **Web3 technologies**, enabling users to mint, list, and buy NFTs directly on the Ethereum blockchain. Assets are stored via **IPFS** using **Pinata**, and smart contracts are written in **Solidity** and deployed using **Truffle**.

KEY FEATURES

- **Wallet Integration (MetaMask)**
  Wallet connection and signature-based authentication using **Web3.js**.
- **NFT Minting via IPFS**
  Users can upload images and metadata to **IPFS** using **Pinata**, and mint NFTs using smart contracts.
- **Smart Contracts with Solidity + Truffle**
  - Wrote and deployed **ERC-721** compliant smart contracts.
  - Managed token ownership, transfers, and sale listings directly on-chain.
- **NFT Marketplace Functionality**
  - Users can list, buy, and transfer NFTs.
  - Marketplace reflects real-time blockchain state via contract event listeners.
- **Next.js Frontend + API Routes**
  - Built the frontend using **Next.js** for server-side rendering and API routes for lightweight backend logic.
  - Fully responsive UI with **TailwindCSS**.
- **Dynamic NFT Dashboard**
  - View NFTs you've created, own, or have listed.
  - Status indicators and filters for user assets.
- **MongoDB Integration**
  - Optionally logged user profiles or activity off-chain for analytics or enhanced features.

WHAT I SOLVED

- Removed the need for centralized asset storage with **IPFS**.
- Ensured private, on-chain NFT ownership and secure marketplace transactions.
- Designed a smooth **Web3 UX** to abstract away blockchain complexity from users.

RESULTS

- Built a fast, decentralized app combining **blockchain + modern web** best practices.
- Delivered full-stack capability: from smart contract deployment to responsive frontend and backend logic with Next.js.
- Proved ability to build real-world **DApps** with modern tooling and decentralized principles.