Enock Bett

FullStack Software Engineer
Github Profile

Experience

Fullstack Software Engineer

TabPay – Community Financial Management Platform

Nairobi Kenya – May 2024 - Present

- **Core Backend Implementation:** Developed a Flask-based application with SQLAlchemy ORM for database management. Implemented a hierarchical community structure with Umbrellas, Blocks, and Zones models.
- **User Management System**: Built user management with role-based access control using Flask-Security. Integrated Africa's Talking SMS API for notifications.
- **Database Architecture:** Designed relational database schemas for member management, M-Pesa transaction tracking, community hierarchy management, and meeting tracking.
- **Authentication & Security**: Implemented user authentication with Flask-Security, built approval workflows, and integrated CSRF protection across forms.
- **Payment Integration:** Developed payment tracking systems with M-Pesa integration and transaction status tracking.
- **Frontend Development:** Created responsive search interfaces, dynamic form handling with WTForms, and an administrative dashboard for community management.
- **System Features:** Member management, meeting scheduling, payment processing, SMS notifications, and search functionality.

Technical Stack: Python, Flask, SQLAlchemy ORM, Flask-Security, WTForms, M-Pesa, Africa's Talking API, Bootstrap, Jinja2.

Projects

TabPay API (FastAPI)

GitHub Link: TabPay_API

- **Modernized API Architecture:** Migrated the TabPay platform from a traditional RESTful API to FastAPI, leveraging asynchronous programming to enhance performance and scalability.
- **Improved Concurrency:** Developed asynchronous endpoints that significantly improved the handling of concurrent requests, reducing latency and increasing throughput.
- **Robust Security & Error Handling:** Implemented comprehensive error handling, data validation, and JWT-based authentication to ensure secure payment processing and reliable API performance.
- **Automated Documentation:** Utilized FastAPI's built-in OpenAPI/Swagger support to create dynamic, self-updating API

Details

0798 354 820
enockbett427@gmail.com

Skills

Programming Languages: Python, HTML5, CSS3

Frameworks & Libraries: Flask, SQLAlchemy, Flask-Security, Flask-RESTful, Flask-Mail, Flask-WTF, Flask-Migrate, Bootstrap,FastAPI

Databases: PostgreSQL, SQLite

Tools & Technologies: Docker, JWT, Flask-Login, Alembic, Africa's Talking API, Swagger/OpenAPI, Git, CI/CD

Other: REST API Design, Database Design, Security Practices, Web Accessibility, Responsive Design, Mpesa Daraja API, Git & Github, Deployment

Links

LinkedIn Profile

documentation, streamlining developer integration and reducing onboarding time.

- **Seamless Transition:** Coordinated with cross-functional teams to ensure a smooth migration from the legacy RESTful system to the new FastAPI framework, maintaining backward compatibility for existing consumers.

Technical Stack: Python, FastAPI, Uvicorn, SQLAlchemy , Pydantic, JWT Authentication, Docker, PostgreSQL, OpenAPI/Swagger Documentation.

Weather Application (Flask/OpenWeatherMap API)
GitHub Link: Weather App

- Designed and developed a responsive weather application that retrieves real-time weather data using the OpenWeatherMap API..
- Implemented secure API key management using dotenv to keep sensitive credentials protected.
- Built functionality to fetch weather details based on user-provided city and country, including temperature, humidity, wind speed, and sunrise/sunset times.
- Developed dynamic date and timezone adjustments for localizing weather information using Python's datetime module.
- Created a user-friendly interface using Bootstrap, featuring a search form with validation and dynamic data rendering.

Technical Stack: Python, Flask, OpenWeatherMap API, Jinja2, Bootstrap, HTML5, CSS3, dotenv, requests, datetime, SQLite

Blog Platform (Flask/SQLAlchemy)
GitHub Link: Blog Platform

- Built secure user authentication with bcrypt password hashing and JWT-based password reset.
- Implemented CRUD operations for blog posts with user authorization checks and pagination.
- Created profile management with image upload and resizing using Pillow.
- Integrated Flask-Mail for automated password reset emails via Gmail SMTP.

Technical Stack: Python, Flask, SQLAlchemy, WTForms, Jinja2, Bcrypt, Flask-

Login, Flask-Mail, Pillow, SQLite.

Course Management System (Flask/SQLAlchemy)
GitHub Link: Course Management System

- Developed secure user authentication using Flask-Security and role-based access control.
- Implemented CRUD operations for courses and enrollments with user authorization checks.
- Created role management system for Admin, Teacher, and Student roles.
- Integrated Flask-Mailman for automated email notifications.
- Deployed the application using Docker and Vercel.

Technical Stack: Python, Flask, SQLAlchemy, Flask-Security, Flask-Mailman, Flask-Migrate, Alembic, Docker, Vercel, PostgreSQL, Jinja2, WTForms.

Todo List Application (Flask/SQLAlchemy)
GitHub Link: Todo List App

- Built CRUD operations for todo items with category-based organisation.
- Developed dynamic form validation using Flask-WTF and WTForms.
- Integrated PostgreSQL database and implemented flash messaging for user feedback.
- Implemented category-based filtering for todos.

Technical Stack: Python, Flask, SQLAlchemy, Flask-WTF, Flask-Migrate, Alembic, WTForms, PostgreSQL, Jinja2.

Sports Club Website (Bootstrap)
GitHub Link: Sports Club Website

- Created a responsive sports club website using Bootstrap 5 and semantic HTML5.
- Built an administrative dashboard for fixture management with sortable tables.
- Developed a contact form system with validated input fields for name, email, and messages.

Technical Stack: HTML5, CSS3, Bootstrap 5, Responsive Design, Web Accessibility.

Education

Africode Academy
Full Stack Software Engineering Nanodegree

Coursework:

- **Web Development:** HTML, CSS, and JavaScript fundamentals for building responsive and interactive web applications.
- **Python Programming:** Mastered core Python concepts, including data structures and object-oriented programming.
- **Database & SQL:** Proficient in CRUD operations, database design, and management for efficient data handling.
- **API Development:** Developed RESTful APIs using Flask, focusing on scalability and performance.
- **Access Management:** Implemented role-based access control (RBAC) for secure user authentication and authorization.
- **Server Deployment:** Gained expertise in Docker, and CI/CD pipelines for automated and scalable application deployment.