Group Members:

Athanase Abayo

Mabinty Mambu

Olivier Kwizera

Victoria Ama Nyonator

Course Name: Web Technologies

Cohort B

Individual Contributions Report

Lecturer: Dr. Osafo-Maafo Kwadwo

Teacher Assistant: Barbara-Marie Doh

November 30, 2025

SwapIt: Individual Contributions Report

1. Athanase Abayo – Team Lead and Backend Architect

Responsibilities

- Oversaw system architecture and design
- Database schema design and implementation
- Authentication system architecture and session management
- API endpoint structure
- Integration of all team contributions
- Code review and quality assurance
- Git repository management

Key Features Implemented

Feature	Description
Authentication System	Handles login, registration, and session management; secure cookies with 24-hour persistence; IP tracking for hijacking detection
Database Architecture	MySQL connection with automatic fallback; MockDatabase for development; UTF-8 support
User Dashboard System	Displays user listings, stats, and provides edit/delete functionality
Landing Page	Main landing page with responsive navigation, hero section, featured items, testimonials
Routing System	Custom PHP router handling API endpoints and HTML pages with 404 handling
Shopping Cart Integration	CartManager with localStorage persistence; add/remove items, calculate total

Important Algorithm / Design

Session-Based Authentication with Security Layers:

- Multi-layered approach: HttpOnly cookies, server-side validation, rate limiting
- IP checks for session hijacking, account lockout after failed attempts

Challenges and Solutions

- Session Persistence: Configured cookie path, lifetime, and fetch request credentials
- Database Fallback: MockDatabase allowed independent development without MySQL
- Router Root Handling: Added specific root URL handling to avoid 404s
- Security Integration: Used the decorator pattern for adding rate limiting and logging

Lessons Learned

- Architecture upfront saves debugging time
- Security must be designed, not patched in
- Communication and documentation accelerate development
- Test early and often; use Git workflows effectively

2. Mabinty Mambu – Backend Developer and User Management Specialist

Core Responsibilities

- User registration, profile updates
- Wishlist and shopping cart functionality
- Listing creation and management
- API integration and error handling
- Category and location-based filtering
- Planning of system design and architecture

Key Features Implemented

Feature	Description
User Registration	Signup method with duplicate email checking and transaction handling
Profile Update	Validates changes, updates the session, and handles avatar/name updates
Wishlist System	Toggle, add/remove items, persistent in localStorage, badge count updates
Shopping Cart	Quantity management, cart display, wishlist integration
Browse Page Filtering	Filter by category and location with dropdown handling
Dashboard Listings	Load user-specific listings and statistics

Important Algorithm / Design

Wishlist Toggle with State Management:

- Adds or removes items with UI feedback
- Persistent across page refreshes
- Badge counter updates dynamically

Challenges & Solutions

- Duplicate Email Prevention: Database validation and error logging
- Wishlist-Cart Synchronization: Differentiated UI states; "Move to Cart" feature
- Profile Update Session Sync: Updated session immediately after database update
- Filter State Management: Maintained selected filters across renders

Lessons Learned

- Immediate user feedback is critical
- Validate data on the client and server sides
- localStorage simplifies rich features without a backend dependency
- Edge-case testing uncovers hidden bugs

3. Olivier Kwizera – Security Engineer and Frontend Developer

Core Responsibilities

- Application security: input validation, sanitization, logging
- Rate limiting & brute force protection
- Password security and validation
- Browse page filtering (price & sorting)
- Profile page and avatar management
- Planning of system design and architecture

Key Features Implemented

Feature	Description
Input Sanitization	Prevents XSS; validates email and image uploads
Security Logging	Logs events with timestamps, IPs, user agents
Rate Limiting	Limits login attempts; account lockout; sliding window
Image Upload Validation	File type, size, integrity, MIME whitelisting
Password Security	Strong password enforcement; integrated with rate limiting
Browse Page Filtering	Price range filters; sorting low-to-high / high-to-low
Profile Page and Avatar	Upload, compress, display profile pictures
Client-side XSS Prevention	Escapes HTML in user-generated content

Important Algorithm / Design

Sliding Window Rate Limiter:

- Limits 5 failed attempts per 15 minutes
- Hashes identifiers for privacy
- Unlocks automatically; logs all security events

Challenges & Solutions

- Rate limiter permissions: auto-created directories
- Strict image validation: multiple checks to avoid false positives
- Session hijacking false positives: log only, no forced logout
- Security headers: balanced CSP to allow inline scripts where safe

Lessons Learned

- Balance security with usability
- Comprehensive logging is essential
- Defense-in-depth prevents multiple attack vectors
- Testing real attack scenarios uncovers vulnerabilities

4. Victoria Ama Nyonator – Frontend Developer & UI/UX Specialist

Core Responsibilities

- Oversaw system architecture and design
- User interface design and implementation
- Form validation and user feedback
- Notification system
- Shopping cart and add-listing UI
- Login, signup, dashboard, and profile pages
- Image processing and compression

Key Features Implemented

Feature	Description
Add Listing Page	Form with category, location, price input, image upload, and validation
Login Page	Password toggle, "Remember Me", responsive design, validation
Signup Page	Password strength, Terms and Conditions checkbox, and real-time validation
Login Form Handler	Validates input, communicates with API, and provides feedback
Shopping Cart UI	Render cart, notifications, badge count, calculate total
Dashboard Stats	Active listings, total views, transaction metrics
Profile Avatar Management	Compression, preview, and upload handling
Notification System	Auto-dismiss, success, error, info types
Browse Page Debounce	Optimized search filtering to prevent excessive re-renders

Important Algorithm /Design

Image Compression and Upload with Preview:

- Compresses images
- Provides instant preview and feedback
- Reduces server storage and improves load speed

Challenges and Solutions

- Balancing image quality and size
- Form validation UX: inline, color-coded, clear messages
- Notification overlap: queued and auto-dismissed
- Responsive navigation

Lessons Learned

- User feedback is essential
- Performance optimization improves UX
- Mobile-first design simplifies development
- Accessibility and progressive enhancement matter

Final Group Reflection

Working on this project taught us a lot about server-side development and about using AI responsibly. Before, we only knew that servers store data and send it to users, but now we understand that the server is like the brain of a website. It manages logins, stores user data, and ensures everything runs smoothly. Using classes like SwapItAuth, DashboardManager, and ProfileManager, we learned how servers handle user accounts, sessions, and data safely. We also saw that small mistakes on the server side, like missing input checks, can cause security problems or make the website break. This showed us why following best practices is so important.

We also explored AI tools like ChatGPT and Gemini. These tools assisted us with coding, language translation, and generating ideas. But we realized AI isn't perfect; it can make mistakes or suggest things that don't work. Using AI responsibly meant that we had to verify its work, understand its limitations, and ensure it didn't create any security or ethical issues. For example, when AI was used for language translations or dashboard functions, we ensured it worked correctly in both English and French and didn't expose user data.

Working on the project also improved our problem-solving skills. Combining server-side logic with client-side features, such as search filters, carts, and multi-language support, required planning, testing, and bug fixes. We learned that AI can make our work faster, but we, as humans, still need to make the final decisions.

Overall, this project taught us the importance of servers in building a smooth and safe website, as well as how to utilize AI in a responsible manner. It improved our technical skills and reminded us to always be careful, thoughtful, and ethical when using technology.