Capstone Weekly Status Report 8

Date and Time: October 26, 2025

Student Name: Satwik Alla

Capstone Project: SBOM Manager - Software Bill of Materials Management System

Capstone Sponsor: Software Engineering Department **Ethical Software Lab** – In-house project

Capstone Advisor: Prof. Mohammad Samarah & Prof. Melissa Sahl

Status Details:

Work accomplished this Week

Frontend Foundation & Authentication UI:

- React Project Setup**:** Initialized Vite + React 19 + TypeScript project with professional folder structure (components, pages, layouts, hooks, services, store, types, utils). Configured development environment with hot module replacement.
- Core Dependencies**:** Integrated React Router DOM v7.9.4 (routing), Zustand v5.0.8 (state management), TanStack React Query v5.90.5 (server state), Axios v1.12.2 (HTTP), React Hot Toast v2.6.0 (notifications), Tailwind CSS v3.4.1 (styling), React Hook Form v7.65.0 + Zod v4.1.12 (form validation).
- API Client & Auth Service: Built Axios client with interceptors for automatic token injection, error handling, and retry logic. Implemented authentication service with register, login, logout, token refresh, getCurrentUser, and password reset methods.
- State Management**:** Created Zustand store for authentication state (user, tokens, isAuthenticated, isLoading) with localStorage persistence and automatic token refresh mechanism.
- Authentication Pages: Completed 5 pages - RegisterPage (multi-step with validation), LoginPage (remember me, password toggle), VerifyEmailPage (success/error states), ForgotPasswordPage (reset request), ResetPasswordPage (new password with validation).
- Protected Routing**:** Implemented ProtectedRoute component with authentication checks, automatic login redirects with return URL preservation, and seamless navigation after authentication.

- Navigation & Pages**:** Built responsive Header with logo, nav links, and user dropdown. Created DashboardPage (welcome, stats cards, quick actions) and ProfilePage (user info display, editable forms, account management). Developed mobile-responsive hamburger menu.
- UI Components**:** Developed reusable components - Button (primary/secondary/danger variants), Input with validation, PasswordInput with show/hide, Spinner, LoadingSkeleton, EmptyState for no-data scenarios.
- Error Handling & UX**:** Implemented comprehensive error handling with user-friendly messages, network error detection, form validation with real-time feedback, loading states for all async operations, and toast notifications (success/error/info/warning variants).
- Responsive Design & Testing**:** Ensured mobile-first responsive design across all breakpoints (mobile: 375px-428px, tablet: 768px-1024px, desktop: 1280px-1920px). Conducted end-to-end testing of authentication flow, protected routes, token refresh, form validations, and error scenarios.

Work Planned for Next Week

File Upload Interface & SBOM Viewer:

- File Upload Page**:** Create drag-and-drop interface with file type validation (ZIP, APK, IPA, EXE, DEB, RPM), 50MB size validation display, multi-file support, real-time progress bar, and status indicators (uploading/processing/completed/failed).
- Application List View**:** Implement pagination, sorting (date/name/platform), filtering by platform/OS/binary type, search functionality, and responsive table/card layouts.
- SBOM Detail Page**:** Display SBOM metadata (platform, version, file hash, upload date), component tree visualization, license breakdown charts, and export options (CycloneDX JSON, SPDX JSON, CSV).
- Component Management**:** Build sortable component table with columns (name, version, license, category), expandable rows for dependency details, search/filter within components, and pagination.
- Real-time Updates**:** Implement polling mechanism for SBOM processing status, notification system for completion, WebSocket connection (stretch goal).
- Testing**:** End-to-end upload workflow testing, SBOM data display accuracy validation, export functionality verification, error handling for upload failures, responsive design testing.

Problems, obstacles, needs, or questions that I need help with from client, advisor, or instructor

- CORS Configuration**:** Need to verify backend CORS settings for localhost:5173 development. Should we add additional security headers?

- Component Visualization**:** Should we use simple collapsible tree or advanced graph library (React Flow, D3.js) for dependency visualization? Trade-off between complexity and user value for MVP.
- File Upload UX**:** Should we show warnings for large files before uploading? Is chunked upload implementation necessary for Alpha demo?
- Real-time Updates**:** Is polling sufficient for Alpha, or should WebSocket be implemented now?
- Testing Strategy**:** Prioritize automated frontend tests (Jest, React Testing Library) now or focus on features for Alpha and add tests afterward?
- Deployment**:** Should I do test deployment to Vercel this week to identify environmental issues early, or wait until features are more complete?

Overall Status

- Project Status Color: Green
- Project Status Color from Previous Week: Green