# CS 499 Module 5-2 Narrative

# Databases

**Briefly describe the artifact. What is it? When was it created?**

The artifact is an **Inventory Management application** originally developed as a mobile application using Java, designed to manage inventory data with a SQLite database. This application allowed users to perform CRUD operations (Create, Read, Update, Delete) on inventory items, storing the data in a structured relational database. It was created during a previous course (CS-360 Mobile Application Development), aimed at showcasing proficiency in mobile app development and database management using a traditional relation database model.

As part of the enhancement plan, I chose to migrate this artifact to a modern web-based application using the MEAN stack (MongoDB, Express.js, Angular, Node.js). The migration involved replacing the original SQLite database with MongoDB, a NoSQL database, to better support the application’s scalability and flexibility needs. The project was updated to incorporate the latest web technologies, streamline data management, and enable a more robust and scalable infrastructure.

**Justify the inclusion of the artifact in your ePortfolio. Why did you select this item? What specific components of the artifact showcase your skills and abilities in software development? How was the artifact improved?**

The inclusion of this **Inventory Management application** in my ePortfolio provides an opportunity to showcase my skills in software development, particularly in database management, full-stack development, and application migration. This project reflects a significant transformation from a mobile application built in Java with a relational database (SQLite) to a modern, scalable MEAN stack web application using MongoDB. The migration process highlights my ability to adapt to new technologies and re-engineer software systems to meet evolving business needs.

**Why I selected this item**

I selected this artifact because it exemplifies the breadth of my expertise across multiple areas of software development. The transition from a mobile to a web application, combined with the shift from a relational to a NoSQL database, required not only technical proficiency but also strategic planning and problem solving. This project allowed me to demonstrate my ability to manage complex database migrations, develop RESTful APIs, and integrate a robust front-end with a dynamic backend.

**Components showcasing skills and abilities**

Several components of this artifact effectively showcase my skills and abilities:

1. **Database Migration:** The shift from SQLite to MongoDB required an in-depth understanding of both relational and NoSQL databases. Designing a new schema in MongoDB to accommodate the existing data while leveraging MongoDB’s flexibility highlights my database management skills.
2. **Backend Development:** Rewriting the backend services in Node.js and Express.js to interact with MongoDB showcases my proficiency in full-stack development and my ability to create scalable, maintainable APIs.
3. **Frontend Integration:** Updating the Angular frontend to communicate seamlessly with the new backend services required a strong grasp of both client-side and server-side technologies, demonstrating my ability to develop cohesive and user-friendly interfaces.
4. **Software Engineering Principles:** Throughout the project, I applied software engineering best practices, including modularity, code optimization, and documentation (using JSDoc style), which are crucial for maintaining software quality and ensuring long-term maintainability.

**Improvement made to the artifact**

The transition from the original Java-based mobile application to a MEAN stack application brought about several significant improvements, particularly in terms of database architecture, user experience, and overall software design.

**Scalability and Flexibility**

One of the most impactful enhancements was the shift from SQLite, a relational database, to MongoDB, a NoSQL database. In the original mobile app, the database structure was fixed, with predefined tables for users, inventory, and notifications. This rigid schema could make it difficult to scale and adapt the application as new requirements emerge. By adopting MongoDB, the application now leverages a document-based model, which offers greater flexibility in handling data structures. For example, MongoDB allows storing varied and nested data types within documents, which simplifies the management of complex data without requiring major changes to the schema. This flexibility is particularly beneficial as the application grows and evolves, allowing for seamless addition of new features or data types.

**Performance Optimization**

The migration to MongoDB enables performance optimizations through the use of indexing and data retrieval methods. In the original SQLite setup, complex queries could lead to performance bottlenecks as the database grew. MongoDB, with its support for indexing on any field, greatly enhances query performance. For instance, in the **db.js** setup, the connection is optimized to handle various events such as reconnection on failure, making the database more resilient and responsive. Additionally, MongoDB’s ability to handle large volumes of data efficiently means that operations like retrieving user information or inventory items are faster and more scalable.

**Enhanced User Experience**

On the frontend, the move to Angular in the MEAN stack provided an opportunity to enhance the user experience. The new Angular frontend is not only more responsive but also more intuitive, with features like real-time updates and improved form validations. For example, the login and registration screens now provide immediate feedback on user input, like validating the existence of a registered user without a page refresh. This creates a smoother integration with the backend, supported using RESTful APIs that ensure data consistency and quick responses to user actions.

**Modernized Architecture**

The architecture of the application was modernized by decoupling the frontend and backend, adhering to RESTful principles, and implementing a more modular design. In the original setup, the backend logic was tightly coupled with the database operations, making it difficult to maintain or scale. The new architecture, as seen in the **app.js** and **userController.js** files, separates concerns more effectively. The use of Express.js to handle API requests and manage routes makes the backend more organized and easier to extend. For instance, the user registration and authentication processes are now handled by dedicated routes and controllers, which not only improves code readability but also makes the application more secure by isolating critical functions. This decoupled structure ensures that the application is easier to maintain and future-proof, as new features can be added without disrupting existing functionality.

**Did you meet the course objectives you planned to meet with this enhancement in Module One? Do you have any updates to your outcome-coverage plans?**

Yes, I have met the course objectives I planned to meet with this enhancement. Through this process, I demonstrated skills in database migration, schema design, and the streamlining of data handling using MongoDB. The transition highlighted my ability to develop and integrate backend services with a NoSQL database, ensuring scalability as the application grows.

In alignment with the course outcomes, this project effectively showcased my ability to use innovative techniques and tools in computing practices to deliver a valuable, industry-specific solution. By designing and evaluating a computing solution that addressed the shift from a relational to a document-based databased, I managed the trade-offs involved in this design choice, particularly in how data is structured and queried.

As for updates to my outcome-coverage plan, I remain committed to achieving the original objectives.

**Reflect on the process of enhancing and modifying the artifact. What did you learn as you were creating it and improving it? What challenges did you face?**

The process of enhancing and modifying the artifact was both challenging and enlightening. One of the most significant learning experiences was understanding the separation of concerns between the backend and frontend logic. This required creating distinct folders like **app\_public**, **app\_api**, and **app\_server**, each with its purpose in maintaining a clean and organized codebase. This separation was crucial in managing the complexity of the application as it grew, ensuring that each part of the application could be developed and maintained independently.

As I delved deeper into the MEAN stack, I faced challenges in understanding the roles of backend controllers, models, and routes, especially in contrast to frontend controllers and routes. This required a shift in mindset, as the structure of a full-stack web application is quite different from that of a mobile app. However, as I worked through these challenges, I gained a much better understanding of how to structure a scalable and maintainable application, which is an invaluable skill in software development.

Creating the **db.js** file was another crucial step in the process. This file handles the connection to the MongoDB database, sets up event listeners for various connection states, and ensures that the database connection is properly closed when the application shuts down. This setup was essential for ensuring the reliability of the database connection, which is the backbone of the entire application. It was a learning experience to understand how MongoDB integrates with Node.js and how to handle different scenarios like app termination or database errors.

Throughout this process, I encountered several errors that required troubleshooting, especially when making HTTP requests from Angular to the Node.js backend. Debugging these errors taught me a lot about handling CORS issues, understanding how Angular’s HTTP client interacts with RESTful APIs, and ensuring that the correct data is being passed between the frontend and backend.