Setting up the Narrative for Success  
Southern New Hampshire University  
Cortland Burns  
03AUG2025

The two artifacts I am using for the database category are my Travlr Getaways full-stack travel booking application and my InvenTitan cross-platform inventory management application. Travlr Getaways was originally developed in May-June 2025 as part of my CS-465 Full Stack Development, built using the MEAN stack. InvenTitan began as a Java-based Android application in an earlier mobile development course and was migrated to Flutter/Dart in July of 2025 to make it cross-platform. Both projects are heavily dependent on databases, but they also present a wide variety of challenges I am trying to address. Some examples include Travlr in managing a secure scalable cloud NoSQL system, and InvenTitan in working with an offline persistence and integrates cloud synchronization to allow for distributed inventory management.

I included these artifacts in my ePortfolio because they can each show different sets of database skills that are important for professional developers. In Travlr Getaways, I was able to show a good design, query optimization, secure data access, and integration with authentication workflows. I improved the existing MongoDB models by adding validation rules in Mongoose, creating compound indexes for more constant queries, and implementing role-based access controls that are connected to JWT authentication. This not only helps reduce the latency but help protect sensitive data. I also redid the API endpoint to use more restricted queries and input sanitization, trying to protect from injection vulnerabilities and meeting the needs for secure development and following best practices.

For InvenTitan, the improvements focus on preparing the project for scalable, cloud-backed data storage. The current SQLite local storage works well for offline use, but it is really limited when it comes to synchronization and multi-device access. I looked at Firebase Firestore for real-time sync and ease of integration with Flutter, AWS DynamoDB for the enterprise scalability and security, and a hybrid setup that could combine both. My conclusion was that Firebase would be great for small to mid-sized real-time inventory updates like mobile devices in the field, while DynamoDB would be great for secure, long-term, and durable data storage for compliance and auditing purposes. This choice required that weighed the trade-offs between speed, cost, data flexibility, and requirements which are all skill connected with database designs.

In Module One, my goal for this was to show a secure database design and my optimization skill using multiple architectures. I met this outcome with Travlr Getaways by improving MongoDB’s performance and security, and I am extending it with InvenTitan by creating a migration plan to a cloud-backed architecture that is scalable and has real-time features. This approach has given me a chance to work with both NoSQL cloud databases and embedded databases, showing my versatility in choosing and creating the right database for the right problem.

The process of improving these artifacts has taught me that database work is as much about the security and build of it as it is about being fast and efficient. Travlr Getaways, I had to learn how to implent Mongoose middleware for pre-save validation and to carefully work on the indexes to avoid performance issues. In InvenTitan, the challenge was more about the building itself and choosing between single-cloud, multi-cloud, or hybrid database builds while making sure that Flutter’s new cross-platform build would be able to integrate properly. The biggest challenge in both was balancing between my personal convenience with having decent security and performance, especially when I was working on conflicting issues regarding speed and proper security.

By improving the database security, build, and scalability in Travlr Getaways, and by planning and partially implementing a more modern, cloud-based database strategy for InvenTitan, I have shown my ability to create and work on computing solutions using the database principles, address potential vulnerabilities, and work on new techniques to meet goals. These improvements set both applications up for long-term maintainability and a more real-world development environment.