3-2 Milestone Two: Enhancement One: Software Design and Engineering

Gavin M. Bish

Southern New Hampshire University

CS499: Computer Science Capstone

Professor Neil Kalinowski

March 16th, 2025

3-2 Milestone Two

The artifact selected for this ePortfolio is the Patient Management System, a Java application designed to manage patient records and appointments for healthcare providers. The original system was developed as part of a software development course project in early 2023, focusing primarily on basic appointment scheduling. The enhancement process took place during the summer and fall of 2024, with significant improvements to the data persistence layer and search functionality.

The Patient Management System is a Java-based desktop application that enables healthcare providers to manage patient records and schedule appointments. The original system included basic functionality for creating, retrieving, updating, and deleting appointments but lacked robust patient data management and advanced search capabilities. The core components of the system include:

1. Data models such as Patient and Appointment classes
2. A persistence layer for storing and retrieving data
3. Service layers that implement business logic
4. Testing components to validate functionality

The artifact was initially created as part of a software development course project that focused on object-oriented programming principles and basic application development using Java. The original implementation demonstrated fundamental programming concepts but had several limitations and bugs that needed to be addressed.

I selected this artifact because it demonstrates my ability to identify and resolve complex software issues while implementing enhancements that significantly improve the functionality and reliability of an application. The enhancement process showcases my skills in debugging, object serialization, software architecture, and testing—all critical competencies for software developers.

The Patient Management System represents a real-world application that addresses actual needs in the healthcare domain. By enhancing this system, I was able to apply software development principles to solve practical problems, demonstrating my ability to translate conceptual knowledge into functional solutions.

The artifact showcases my debugging skills through the identification and resolution of a critical NullPointerException in the EnhancedAppointment class. This error occurred during deserialization when the application attempted to load appointment data from persistent storage. By analyzing the class structure and serialization process, I determined that the composition field appointment was not being properly initialized after deserialization, causing method calls to fail with NullPointerExceptions.

**Object-Oriented Design**

The enhancements demonstrate my understanding of object-oriented design principles, particularly in the context of inheritance and composition. The original implementation of EnhancedAppointment had a flawed design that simultaneously used inheritance and composition with the Appointment class without properly managing the relationship between these approaches. My solution maintained the existing class structure while fixing the implementation to handle serialization correctly.

The improvements to the data persistence layer showcase my ability to implement robust storage and retrieval mechanisms. By ensuring proper serialization of complex objects and maintaining data integrity across application sessions, I demonstrated practical knowledge of data persistence techniques in Java applications.

The implementation of the enhanced search functionality demonstrates my ability to design and implement algorithms that meet specific requirements. The search function supports multiple search types (date, time, patient information) and employs pattern matching and parsing techniques to interpret user queries correctly.

The artifact was improved in several significant ways:

1. **Fixed Serialization Issues**: The EnhancedAppointment class was modified to properly handle serialization and deserialization by:
   * Marking the appointment field as transient to exclude it from serialization
   * Implementing a custom readObject method to reinitialize the field after deserialization
   * Using superclass methods instead of delegating to potentially null fields
2. **Enhanced Search Functionality**: Implemented a comprehensive search algorithm that can:
   * Detect search types automatically based on input format
   * Search across multiple patient and appointment fields
   * Handle date and time formats intelligently
3. **Improved Patient Data Management**: Added functionality to:
   * Create new patient records with extended information
   * Update existing patient records while maintaining referential integrity
   * Link patients to appointments through consistent ID mechanisms
4. **Robust Data Loading**: Enhanced the data loading process to ensure all data structures are properly initialized and populated before application use.

The enhancements to the Patient Management System align with several course outcomes I planned to meet:

1. **Employing strategies for building collaborative environments**: The improvements demonstrate my ability to understand and enhance existing code, a critical skill in collaborative development environments where developers often work with code written by others.
2. **Developing a security mindset that anticipates adversarial exploits**: By addressing serialization vulnerabilities, I demonstrated awareness of potential security issues in data persistence implementations.
3. **Designing and evaluating computing solutions**: The search functionality enhancement shows my ability to design algorithms that balance functionality, performance, and user needs.
4. **Using well-founded and innovative techniques in computing practices**: The implementation of custom serialization handling demonstrates advanced Java techniques to solve complex problems.

Beyond my initial plans, I also addressed outcomes related to software testing through the debugging process and fix verification. This unexpected focus on testing methodologies has enhanced my appreciation for test-driven development approaches.

The process of enhancing the Patient Management System provided several valuable learning experiences:

1. **Understanding Complex Class Relationships**: Working with the hybrid inheritance-composition structure in EnhancedAppointment deepened my understanding of object-oriented design patterns and their implications for serialization. I learned that when using both inheritance and composition with the same base class, special care must be taken to ensure proper object state management.
2. **Serialization Mechanics**: The debugging process enhanced my knowledge of Java's serialization mechanisms, particularly how transient fields and custom deserialization methods can be used to maintain object integrity across persistence boundaries.
3. **Search Algorithm Design**: Developing the search functionality taught me techniques for designing flexible algorithms that can adapt to different input types and search across multiple data fields efficiently.
4. **Test-Driven Debugging**: The process of fixing the NullPointerException reinforced the value of test-driven development. By focusing on making the test pass, I was able to systematically address the underlying issues.

Throughout the enhancement process, I encountered several challenges that tested my problem-solving abilities:

1. **Serialization Complexity**: The most significant challenge was understanding the root cause of the serialization issue. The error manifested as a NullPointerException during a unit test, but tracing this back to a serialization problem required deep analysis of the object lifecycle during persistence operations.
2. **Maintaining Backward Compatibility**: Ensuring that fixes did not break existing functionality was challenging. The hybrid nature of the EnhancedAppointment class made it difficult to modify without potentially affecting other system components.
3. **Search Algorithm Edge Cases**: Implementing a search algorithm that could intelligently handle various input formats (dates, times, text) presented challenges in pattern recognition and query interpretation. Ensuring the search worked consistently across all possible inputs required careful testing and refinement.
4. **Code Architecture Constraints**: Working within the existing code architecture imposed constraints that sometimes made ideal solutions impractical. I learned to balance theoretical best practices with pragmatic solutions that respected the existing codebase structure.

The enhancement of the Patient Management System represents a significant learning opportunity that allowed me to apply and extend my software development skills. By addressing complex issues in serialization, implementing advanced search capabilities, and improving data management functionality, I demonstrated practical application of software engineering principles to a real-world healthcare application.

The process of enhancing this artifact reinforced the importance of understanding existing code before making changes, systematic debugging techniques, and thorough testing to validate improvements. These skills—analyzing complex systems, identifying root causes of issues, and implementing effective solutions—are foundational to my development as a software engineer and will serve me well in future professional endeavors