CS 499 Module One Assignment Template

Jasmine Morgan

CS 499 Computer Science Capstone

Date: January 11, 2026

GitHub Repository: https://github.com/Jmorgansings/CS360Undergrad

Complete this template by replacing the bracketed text with the relevant information.

I. **Self-Introduction:** Address all of the following questions to introduce yourself.

A. How long have you been in the Computer Science program?

I have been enrolled in the Computer Science program at Southern New Hampshire University since January 2024. Throughout my time in the program, I have completed foundational courses in programming, data structures, algorithms, database management, software engineering, mobile architecture, and security practices. Over these two years, I have progressed from learning basic programming fundamentals to developing complete, production-ready applications across multiple platforms.

B. What have you learned while in the program? List three of the most important concepts or skills you have learned.

During my time in the Computer Science program, I have acquired a comprehensive skill set that spans multiple domains of software development. Three of the most important concepts and skills I have learned include:

1. Mobile Application Development with Android and Kotlin

Through CS-360 Mobile Architecture and Programming, I gained hands-on experience building native Android applications using Kotlin and Java. I learned to architect mobile applications with proper separation of concerns, creating the Inv-Alert inventory management application with a DatabaseHelper class that encapsulates all data logic separate from the UI layer. This course taught me critical mobile development concepts including:

Activity Lifecycle Management: Understanding how Android manages activities and handling state preservation

Local Data Persistence: Implementing SQLite databases for offline-first mobile applications

User-Centered Design: Creating intuitive mobile interfaces that follow Material Design principles

Permission Handling: Requesting and managing runtime permissions (SMS notifications) while respecting user privacy

2. Secure Authentication and Data Protection

I developed a deep understanding of authentication security and data protection through multiple courses. In CS-360, I implemented a unified login/registration system that securely stores user credentials using hashing techniques. I learned critical security concepts including:

Password Security: Implementing proper password hashing and salting to protect user credentials

SQL Injection Prevention: Using parameterized queries with SQLite to prevent injection attacks

Input Validation: Sanitizing all user inputs to prevent malicious data from entering the system

Principle of Least Privilege: Only requesting permissions when needed and explaining why to users

This knowledge is critical for protecting user data and preventing unauthorized access in production systems, especially important in mobile applications where data is stored locally on devices.

3. Database Design and Relational Data Modeling

Working with SQLite in Android taught me how to design efficient, normalized database schemas with proper relationships and constraints. I learned to:

Design Relational Schemas: Creating multiple tables (users, inventory) with primary and foreign keys

Write Efficient Queries: Optimizing SELECT, INSERT, UPDATE, and DELETE operations for mobile performance

Implement CRUD Operations: Building a complete data access layer with all database operations

Manage Database Versioning: Handling schema migrations as the application evolves

Balance Normalization: Understanding when to normalize data versus denormalize for performance

This experience with relational databases provides a foundation that applies to both SQL and NoSQL systems, and taught me to think critically about data organization and access patterns.

Discuss the specific skills you aim to demonstrate through your enhancements to reach each of the course outcomes.

To reach each of the CS 499 course outcomes, I aim to demonstrate the following specific skills through my Inv-Alert mobile application enhancements:

For Course Outcome 1 (Employ strategies for building collaborative environments):

Clear Technical Documentation: Comprehensive README files with setup instructions, architecture diagrams, and API documentation

Professional Code Comments: Well-commented Kotlin code explaining complex logic and design decisions

Modular Architecture: Separating concerns (UI, business logic, data access) to enable team collaboration

Version Control Best Practices: Using Git with clear, descriptive commit messages following conventional commit standards

Code Review Readiness: Writing code that is easy for other developers to understand and extend

For Course Outcome 2 (Design, develop, and deliver professional-quality communications):

User-Friendly Interfaces: Intuitive mobile UI with clear visual feedback and error messages

Professional Documentation: Well-structured enhancement narratives and technical explanations

Visual Communication: Flowcharts and diagrams illustrating algorithms and system architecture

Accessible Design: Implementing accessibility features for users with disabilities

Clear Error Messaging: User-facing messages that are helpful without exposing technical details

For Course Outcome 3 (Design and evaluate computing solutions using algorithmic principles):

Search and Filter Algorithms: Implementing efficient inventory search with multiple filter criteria

Sorting Algorithms: Multiple sort options (by name, quantity, date) with performance analysis

Data Structure Selection: Choosing appropriate data structures (Lists, HashMaps) for specific use cases

Algorithm Complexity Analysis: Understanding and documenting time/space complexity (Big O notation)

Performance Optimization: Using database indexing and caching to improve query performance

For Course Outcome 4 (Demonstrate ability to use well-founded and innovative techniques):

Modern Android Architecture: Implementing MVVM (Model-View-ViewModel) architecture pattern

Room Database Migration: Migrating from raw SQLite to Room Persistence Library for better maintainability

LiveData and ViewModel: Using Android Jetpack components for reactive UI updates

RecyclerView Optimization: Implementing ViewHolder pattern for efficient list rendering

Material Design 3: Applying modern design principles and components

For Course Outcome 5 (Develop a security mindset):

SQL Injection Prevention: Using parameterized queries and prepared statements

Secure Credential Storage: Implementing proper password hashing (bcrypt or Android Keystore)

Input Validation: Comprehensive validation on all user inputs at multiple layers

Permission Management: Requesting minimum necessary permissions with clear justification

Data Encryption: Encrypting sensitive data stored locally on the device

Secure Coding Practices: Following OWASP Mobile Top 10 security guidelines

How do the specific skills you will demonstrate align with your career plans related to your degree?

My career goal is to become a Mobile Application Developer or Full Stack Software Engineer with expertise in creating secure, user-centered applications across multiple platforms. The skills I will demonstrate through these enhancements directly align with this career path for several reasons:

Mobile-First Development Skills:

Modern software development increasingly prioritizes mobile platforms. By enhancing the Inv-Alert Android application, I demonstrate proficiency in:

Native Android development with Kotlin, the industry-standard language for Android apps

Understanding mobile-specific concerns like battery life, offline functionality, and limited screen space

Working within platform constraints and following platform-specific design guidelines (Material Design)

Cross-Platform Transferable Skills:

While this project focuses on Android, the architectural patterns and principles I'm demonstrating transfer to other platforms:

The MVVM architecture pattern is used in iOS (SwiftUI), Android (Jetpack), and web frameworks (React, Vue)

Database design principles apply whether using SQLite, Room, Core Data (iOS), or cloud databases

Security practices like input validation and secure storage are universal across all platforms

Software Engineering Best Practices:

Employers seek developers who can write maintainable, scalable code:

Separation of concerns and modular architecture make codebases easier to maintain and extend

Comprehensive testing and documentation reduce technical debt

Following design patterns makes code more predictable and easier for teams to collaborate on

Security-Conscious Development:

With increasing cybersecurity threats, especially in mobile applications that handle sensitive user data:

Understanding common vulnerabilities (OWASP Mobile Top 10) makes me a more valuable developer

Implementing security from the ground up rather than as an afterthought reduces risk

Knowledge of secure coding practices is increasingly required for developer positions

Problem-Solving and Innovation:

The ability to identify gaps in existing applications and propose practical solutions demonstrates:

Critical thinking skills essential for senior development roles

Understanding of user needs and business requirements

Capacity to work independently and make architectural decisions

These skills directly prepare me for roles such as Mobile Application Developer, Android Developer, Software Engineer, or Full Stack Developer at companies ranging from startups to large enterprises.

How does this contribute to the specialization you are targeting for your career?

This ePortfolio contributes significantly to my specialization in mobile application development with an emphasis on security and user experience by demonstrating:

1. Mobile-Specific Expertise

By enhancing a complete Android application, I showcase specialized knowledge in:

Native Mobile Development: Deep understanding of Android platform APIs, lifecycle management, and platform constraints

Mobile UI/UX Patterns: Implementing touch-friendly interfaces, responsive layouts for various screen sizes, and mobile-specific navigation patterns

Offline-First Architecture: Designing applications that work seamlessly without network connectivity using local storage

Mobile Performance Optimization: Understanding mobile-specific performance concerns like memory constraints, battery usage, and app startup time

2. Security Specialization in Mobile Context

Mobile applications present unique security challenges that this portfolio addresses:

Local Data Security: Protecting sensitive data stored on devices that can be lost or stolen

Authentication on Mobile: Implementing secure login that balances security with user convenience (biometric authentication, token storage)

Permission Model: Understanding Android's runtime permission system and requesting only necessary permissions

Secure Communication: Planning for secure data transmission when connecting to backend services

3. Full Application Lifecycle Understanding

This project demonstrates end-to-end mobile development skills:

Database Layer: Designing and implementing local data persistence with SQLite/Room

Business Logic Layer: Creating a clean separation between data access and UI with repository pattern

Presentation Layer: Building intuitive user interfaces with Material Design components

Testing Strategy: Implementing unit tests, integration tests, and UI tests for mobile applications

4. Modern Development Practices

The enhancements showcase current industry-standard practices:

Kotlin Language Features: Using modern language features like coroutines, null safety, and extension functions

Android Jetpack: Implementing recommended architecture components (ViewModel, LiveData, Room, Navigation)

Dependency Injection: Using modern patterns for testable, maintainable code

Version Control: Professional Git workflow with feature branches and pull requests

5. Bridging Mobile and Full Stack

This specialization positions me uniquely in the job market:

Mobile + Backend: Understanding how mobile apps consume REST APIs prepares me for full stack roles

Cross-Platform Thinking: Architectural patterns learned here apply to iOS, web, and cross-platform frameworks

Cloud Integration Ready: The enhanced local database can be extended to sync with cloud backends (Firebase, AWS, Azure)

Versatile Skill Set: Ability to work on mobile apps, web applications, or integrated solutions

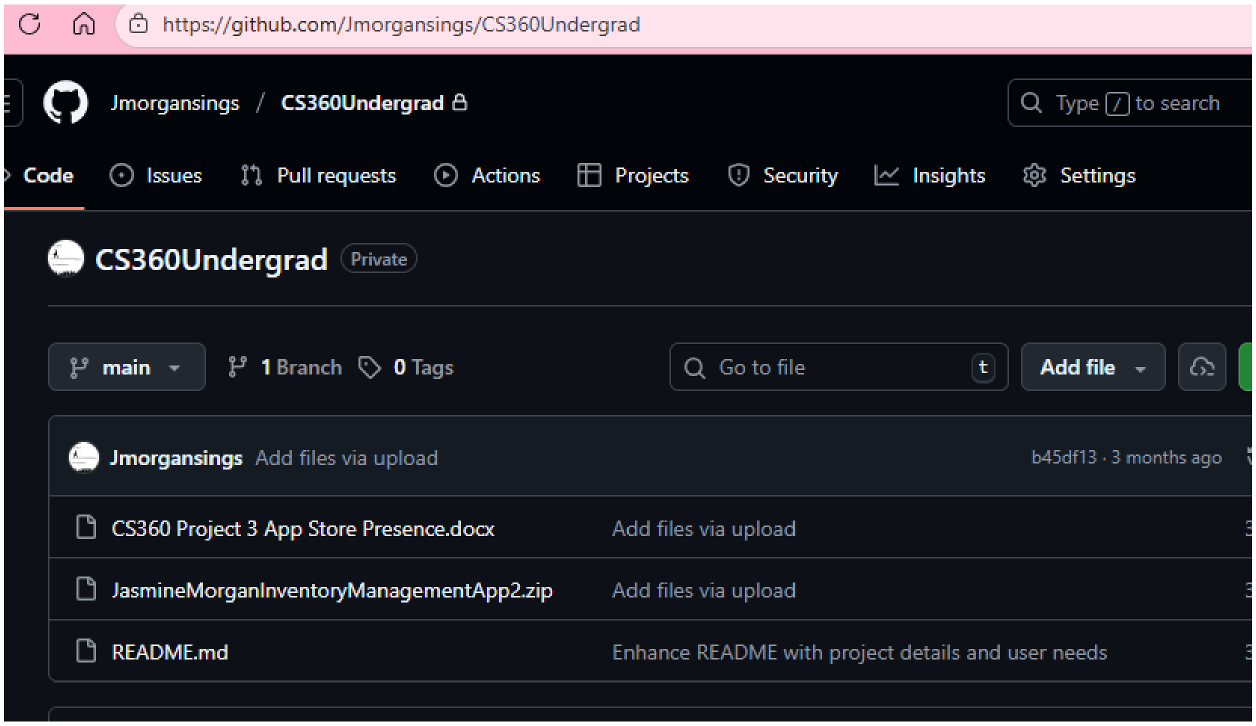
This specialized portfolio makes me a strong candidate for roles that specifically seek mobile developers with security awareness, or for full stack positions where mobile expertise is a differentiator. It demonstrates not just coding ability, but the capacity to make architectural decisions, consider security implications, and create applications that users will trust with their data.

II. ePortfolio Set Up:

A. Submit a **screen capture** of your ePortfolio GitHub Pages home page that clearly shows your URL.
   i. You already have a repository in GitHub where you uploaded projects in previous courses. Your ePortfolio will reside in GitHub but can link to work at other sites, such as Bitbucket.

B. Use the GitHub Pages link in the Resource section for directions on:
   i. How to create your GitHub website and publish code to GitHub Pages
   ii. Issues, such as adding links to other sites

C. Paste a screenshot of your GitHub Pages home page with your URL clearly showing in the space below.


https://github.com/Jmorgansings/CS360Undergrad

Jmorgansings Add files via upload b45df13 · 3 months ago

CS360 Project 3 App Store Presence.docx Add files via upload

JasmineMorganInventoryManagementApp2.zip Add files via upload

README.md Enhance README with project details and user needs

III. Enhancement Plan:

A. **Category One:** Software Engineering and Design

i. **Select an artifact** that is **aligned with the** software engineering and design **category** and explain its origin. Submit a file containing the code for the artifact you choose with your enhancement plan.
https://github.com/Jmorgansings/CS360Undergrad/blob/main/JasmineMorganInventoryManagementApp2.zip

I will use one artifact for all three enhancement categories:
Artifact: Inv-Alert Mobile Inventory Management Application
Origin: CS-360 Mobile Architecture and Programming (2024)
Platform: Native Android application developed in Kotlin and Java
GitHub Repository: https://github.com/jmorgansings/cs360undergrad

Note: Your artifact may be work from the following courses:

- IT 145: Foundation in Application Development
- CS 250: Software Development Lifecycle
- CS 260: Data Structures and Algorithms
- IT 315: Object Oriented Analysis and Design
- CS 320: Software Testing, Automation, and Quality Assurance

- CS 330: Computational Graphics and Visualization
- CS 340: Advanced Programming Concepts
- CS 350: Emerging Systems Architectures and Technologies
- CS 360: Mobile Architecture and Programming
- IT 365: Operating Environments
- IT 380: Cybersecurity and Information Assurance
- CS 405: Secure Coding
- CS 410: Reverse Software engineering
- IT 340: Network and Telecommunication Management
- IT 380: Cybersecurity and Information Assurance

ii. **Describe** a practical, well-illustrated **plan** for enhancement in alignment with the category, including a pseudocode or flowchart that illustrates the planned enhancement.

: Inv-Alert is an Android inventory management application designed for small businesses and hobbyists who need to track inventory levels and receive low-stock alerts. The current implementation includes

User Authentication System: Unified login/registration flow where users can create accounts or log in: with existing credentials on a single screen

SQLite Database: Two-table relational database (users and inventory) with proper primary/foreign key relationships

DatabaseHelper Class: Separation of concerns with all data access logic encapsulated in a dedicated helper class, independent of UI components

CRUD Operations: Full create, read, update, and delete functionality for inventory items

GridView Display: Inventory items displayed in a grid layout for efficient browsing

SMS Notification Permission: Runtime permission handling for sending low-stock alerts via SMS

Material Design UI: User interface following Android design guidelines with FloatingActionButton for adding new items

For this category of enhancement, consider improving a piece of software, transferring a project into a different language, reverse engineering a piece of software for a different operating system, or expanding a project's complexity. These are just recommendations. Consider being creative and proposing an alternative enhancement to your instructor.

Think about what additions to include to complete the enhancement criteria in this category. Since one example option is to port to a new language, that is the kind of scale that is expected. This does not mean you need to port to a new language but instead have an equivalent scale of enhancement. Underlying expectations of any enhancement include fixing errors, debugging, and cleaning up comments, but these are not enhancements themselves.

iii. Explain how the planned enhancement will **demonstrate** specific **skills** and align with course outcomes.

a. Identify and describe the specific skills you will demonstrate that align with the course outcome.

Rationale for Single Artifact Selection

I chose to enhance a single artifact across all three categories because:

1. Comprehensive Mobile Application: The Inv-Alert app already touches all three enhancement domains (software design, algorithms, databases), making it ideal for demonstrating interconnected improvements across the entire application stack

2. Real-World Development Scenario: Enhancing one cohesive codebase reflects actual industry practices where improvements span multiple architectural layers rather than being isolated to single components

3. Clear Growth Opportunities: The application has substantial room for enhancement in each category:

- Software Engineering: Migrate to modern MVVM architecture, implement Material Design 3, improve UX

- Algorithms: Add search/filter/sort functionality, implement efficient data structures, optimize performance

- Databases: Migrate to Room Database, add data validation, implement complex queries and relationships

4. Demonstrates Holistic Thinking: This approach showcases my ability to see mobile applications as integrated systems where architectural decisions in one layer affect others, demonstrating end-to-end development expertise

5. Sufficient Complexity: The application is complex enough to support meaningful enhancements without requiring separate artifacts, while remaining focused enough to complete within the capstone timeline.

b. Select one or more of the course outcomes below that your enhancement will align with.

Design and evaluate computing solutions that solve a given problem using algorithmic principles and computer science practices and standards appropriate to its solution while managing the trade-offs involved in design choices
Course Outcomes:

1. Employ strategies for building collaborative environments that enable diverse audiences to support organizational decision-making in the field of computer science.

2. Design, develop, and deliver professional-quality oral, written, and visual communications that are coherent, technically sound, and appropriately adapted to specific audiences and contexts.
3. Design and evaluate computing solutions that solve a given problem using algorithmic principles and computer science practices and standards appropriate to its solution while managing the trade-offs involved in design choices.
4. Demonstrate an ability to use well-founded and innovative techniques, skills, and tools in computing practices for the purpose of implementing computer solutions that deliver value and accomplish industry-specific goals.
5. Develop a security mindset that anticipates adversarial exploits in software architecture and designs to expose potential vulnerabilities, mitigate design flaws, and ensure privacy and enhanced security of data and resources.

B. **Category Two:** Algorithms and Data Structures

i. **Select an artifact** that is **aligned with the** algorithms and data structures **category** and explain its origin. Submit a file containing the code for the artifact you choose with your enhancement plan. You may choose work from the courses listed under Category One.

I will use one artifact for all three enhancement categories:

Artifact: Inv-Alert Mobile Inventory Management Application

Origin: CS-360 Mobile Architecture and Programming (2024)

Platform: Native Android application developed in Kotlin and Java

GitHub Repository: https://github.com/jmorgansings/cs360undergrad

ii. **Describe** a practical, well-illustrated **plan** for enhancement in alignment with the category, including a pseudocode or flowchart that illustrates the planned enhancement.

kotlin

```
// ViewModel for Inventory Management

CLASS InventoryViewModel(private repository: InventoryRepository) EXTENDS ViewModel


  // LiveData for reactive UI updates

  PRIVATE val _inventoryItems = MutableLiveData<List<InventoryItem>>()

  PUBLIC val inventoryItems: LiveData<List<InventoryItem>> = _inventoryItems


  PRIVATE val _errorMessage = MutableLiveData<String>()
```

```
PUBLIC val errorMessage: LiveData<String> = _errorMessage


PRIVATE val _isLoading = MutableLiveData<Boolean>()

PUBLIC val isLoading: LiveData<Boolean> = _isLoading


INIT:

   loadInventoryItems()

END INIT


// Load all inventory items

METHOD loadInventoryItems():

   SET _isLoading.value = true


   viewModelScope.launch {

      TRY:

         SET items = repository.getAllItems()

         SET _inventoryItems.value = items

         SET _isLoading.value = false

      CATCH exception:

         SET _errorMessage.value = "Failed to load inventory: ${exception.message}"

         SET _isLoading.value = false

   }

END METHOD


// Add new inventory item with validation

METHOD addInventoryItem(name: String, quantity: Int, description: String):

   IF name.isBlank():

      SET _errorMessage.value = "Item name cannot be empty"
```

```
        RETURN

    IF quantity < 0:
        SET _errorMessage.value = "Quantity must be non-negative"
        RETURN

    SET _isLoading.value = true

    viewModelScope.launch {
        TRY:
            SET newItem = InventoryItem(
                name = name.trim(),
                quantity = quantity,
                description = description.trim(),
                dateAdded = System.currentTimeMillis()
            )
            repository.insertItem(newItem)
            loadInventoryItems() // Refresh list
        CATCH exception:
            SET _errorMessage.value = "Failed to add item: ${exception.message}"
        FINALLY:
            SET _isLoading.value = false
    }
END METHOD

// Update inventory item
METHOD updateInventoryItem(item: InventoryItem):
    SET _isLoading.value = true
    viewModelScope.launch {
        TRY:
            repository.updateItem(item)
            loadInventoryItems() // Refresh list
        CATCH exception:
            SET _errorMessage.value = "Failed to update item: ${exception.message}"
        FINALLY:
            SET _isLoading.value = false
    }
END METHOD

// Delete inventory item
METHOD deleteInventoryItem(itemId: Int):
    viewModelScope.launch {
        TRY:
            repository.deleteItem(itemId)
            loadInventoryItems() // Refresh list
        CATCH exception:
            SET _errorMessage.value = "Failed to delete item: ${exception.message}"
    }
END METHOD

// Search inventory by name
METHOD searchInventory(query: String):
    SET _isLoading.value = true

    viewModelScope.launch {
          TRY:
             SET filteredItems = repository.searchItems(query)
             SET _inventoryItems.value = filteredItems
             SET _isLoading.value = false
          CATCH exception:
             SET _errorMessage.value = "Search failed: ${exception.message}"
             SET _isLoading.value = false
        }
    END METHOD

END CLASS

// Repository Pattern for Data Abstraction
CLASS InventoryRepository(private dao: InventoryDao)

    // Suspend functions for async operations
    SUSPEND METHOD getAllItems(): List<InventoryItem>:
        RETURN dao.getAllItems()
    END METHOD

    SUSPEND METHOD searchItems(query: String): List<InventoryItem>:
        RETURN dao.searchByName("%$query%")
    END METHOD

    SUSPEND METHOD insertItem(item: InventoryItem):
        dao.insertItem(item)
    END METHOD
```

```
  SUSPEND METHOD updateItem(item: InventoryItem):
    dao.updateItem(item)
  END METHOD

  SUSPEND METHOD deleteItem(itemId: Int):
    dao.deleteItem(itemId)
  END METHOD

  SUSPEND METHOD getLowStockItems(threshold: Int): List<InventoryItem>:
    RETURN dao.getItemsWithQuantityLessThan(threshold)
  END METHOD

END CLASS

// Activity using ViewModel (simplified)
CLASS InventoryActivity EXTENDS AppCompatActivity

  PRIVATE lateinit viewModel: InventoryViewModel

  METHOD onCreate(savedInstanceState: Bundle):
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_inventory)

    // Initialize ViewModel
    SET viewModel = ViewModelProvider(this).get(InventoryViewModel::class.java)

    // Observe LiveData
    viewModel.inventoryItems.observe(this) { items ->
```

```
        updateRecyclerView(items)
      }

      viewModel.errorMessage.observe(this) { message ->
        IF message != null:
          showSnackbar(message)
      }

      viewModel.isLoading.observe(this) { isLoading ->
        IF isLoading:
          showProgressBar()
        ELSE:
          hideProgressBar()
      }

      setupUI()
    END METHOD

END CLASS
```

Enhancement 1.2: Material Design 3 UI Overhaul with Accessibility

Enhancement 1.2: Material Design 3 UI Overhaul with Accessibility

For this category of enhancement, consider improving the efficiency of a project or expanding the complexity of the use of data structures and algorithms for your artifact. These are just recommendations. Consider being creative and proposing an alternative enhancement to your instructor. Note: You only need to choose one type of enhancement per category.

Think about what additions to include to complete the enhancement criteria in this category. Since one example option is to port to a new language, that is the kind of scale that is expected. Perhaps you might increase the efficiency and time complexity of an algorithm in an application and detail the logic of the increased time complexity. Remember, you do not need to port to a new language but instead have an equivalent scale of enhancement. Underlying expectations of any enhancement include fixing errors, debugging, and cleaning up comments, but these are not enhancements themselves.

iii. Explain how the planned enhancement will **demonstrate** specific **skills** and align with course outcomes.

a. Identify and describe the specific skills you will demonstrate to align with the course outcome.

I chose to enhance a single artifact across all three categories because:

1. Comprehensive Mobile Application: The Inv-Alert app already touches all three enhancement domains (software design, algorithms, databases), making it ideal for demonstrating interconnected improvements across the entire application stack

2. Real-World Development Scenario: Enhancing one cohesive codebase reflects actual industry practices where improvements span multiple architectural layers rather than being isolated to single components

3. Clear Growth Opportunities: The application has substantial room for enhancement in each category:

- Software Engineering: Migrate to modern MVVM architecture, implement Material Design 3, improve UX

- Algorithms: Add search/filter/sort functionality, implement efficient data structures, optimize performance

- Databases: Migrate to Room Database, add data validation, implement complex queries and relationships

4. Demonstrates Holistic Thinking: This approach showcases my ability to see mobile applications as integrated systems where architectural decisions in one layer affect others, demonstrating end-to-end development expertise

5. Sufficient Complexity: The application is complex enough to support meaningful enhancements without requiring separate artifacts, while remaining focused enough to complete within the capstone timeline.

b. Select one or more of the course outcomes listed under Category One that your enhancement will align with.

6. Demonstrate an ability to use well-founded and innovative techniques, skills, and tools in computing practices for the purpose of implementing computer solutions that deliver value and accomplish industry-specific goals.

C. **Category Three: Databases**

i. **Select an artifact** that is **aligned with the** databases **category** and explain its origin. Submit a file containing the code for the artifact you choose with your enhancement plan. You may choose work from the courses listed under Category One.

I will use one artifact for all three enhancement categories:

Artifact: Inv-Alert Mobile Inventory Management Application

Origin: CS-360 Mobile Architecture and Programming (2024)

Platform: Native Android application developed in Kotlin and Java

GitHub Repository: https://github.com/jmorgansings/cs360undergrad

ii. **Describe** a practical, well-illustrated **plan** for enhancement in alignment with the category, including a pseudocode or flowchart that illustrates the planned enhancement.

kotlin

```
// RecyclerView Adapter with Material Design 3
CLASS InventoryAdapter EXTENDS RecyclerView.Adapter<InventoryViewHolder>

  PRIVATE var items: List<InventoryItem> = emptyList()

  METHOD setItems(newItems: List<InventoryItem>):
    // Use DiffUtil for efficient updates
    SET diffCallback = InventoryDiffCallback(items, newItems)
    SET diffResult = DiffUtil.calculateDiff(diffCallback)

    SET items = newItems
    diffResult.dispatchUpdatesTo(this)
  END METHOD

  METHOD onCreateViewHolder(parent: ViewGroup, viewType: Int): InventoryViewHolder:
    SET view = LayoutInflater.from(parent.context)
```

```
        .inflate(R.layout.item_inventory_card, parent, false)
      RETURN InventoryViewHolder(view)
    END METHOD

    METHOD onBindViewHolder(holder: InventoryViewHolder, position: Int):
      SET item = items[position]
      holder.bind(item)
    END METHOD

END CLASS

// ViewHolder with Accessibility
CLASS InventoryViewHolder(itemView: View) EXTENDS RecyclerView.ViewHolder(itemView)

    PRIVATE val nameText: TextView = itemView.findViewById(R.id.itemName)
    PRIVATE val quantityText: TextView = itemView.findViewById(R.id.itemQuantity)
    PRIVATE val card: MaterialCardView = itemView.findViewById(R.id.itemCard)
    PRIVATE val editButton: MaterialButton = itemView.findViewById(R.id.editButton)
    PRIVATE val deleteButton: MaterialButton = itemView.findViewById(R.id.deleteButton)

    METHOD bind(item: InventoryItem):
      SET nameText.text = item.name
      SET quantityText.text = "Qty: ${item.quantity}"

      // Color code for low stock
      IF item.quantity < LOW_STOCK_THRESHOLD:
        SET card.setCardBackgroundColor(colorLowStock)
        SET card.strokeColor = colorWarning
```

```
ELSE:
    SET card.setCardBackgroundColor(colorNormal)
    SET card.strokeColor = colorPrimary

// Accessibility: Content descriptions
SET card.contentDescription =
    "${item.name}, quantity ${item.quantity}" +
    IF (item.quantity < LOW_STOCK_THRESHOLD) " - Low stock warning" ELSE ""

SET editButton.contentDescription = "Edit ${item.name}"
SET deleteButton.contentDescription = "Delete ${item.name}"

// Accessibility: Minimum touch targets (48dp)
SET editButton.minimumWidth = 48dp
SET editButton.minimumHeight = 48dp
SET deleteButton.minimumWidth = 48dp
SET deleteButton.minimumHeight = 48dp

// Click listeners
SET editButton.setOnClickListener {
    onEditClicked(item)
    // Announce action for screen readers
    itemView.announceForAccessibility("Editing ${item.name}")
}

SET deleteButton.setOnClickListener {
    onDeleteClicked(item)
    itemView.announceForAccessibility("${item.name} deleted")
    }
  END METHOD

END CLASS
```

Enhancement 1.3: Comprehensive Error Handling and User Feedback

Description: Implement robust error handling throughout the application with clear, user-friendly feedback for all operations and error scenarios.

Technical Implementation:

Create sealed classes for Result types (Success, Error, Loading)

Implement Snackbar notifications for user feedback

Add loading states with progress indicators

Implement retry mechanisms for failed operations

Log errors for debugging while showing user-friendly messages

Handle network errors, database errors, and validation errors separately

Pseudocode:

kotlin

```
// Sealed class for operation results
SEALED CLASS Result<out T>
  DATA CLASS Success<T>(val data: T) EXTENDS Result<T>
  DATA CLASS Error(val exception: Exception, val message: String) EXTENDS Result<Nothing>
  OBJECT Loading EXTENDS Result<Nothing>
END CLASS

// Enhanced Repository with Result types
```

```
CLASS InventoryRepository(private dao: InventoryDao)

  SUSPEND METHOD getAllItems(): Result<List<InventoryItem>>:
    RETURN TRY:
      SET items = dao.getAllItems()
      Result.Success(items)
    CATCH exception: Exception:
      Log.e(TAG, "Failed to load items", exception)
      Result.Error(exception, "Unable to load inventory items")
  END METHOD

  SUSPEND METHOD addItem(item: InventoryItem): Result<Long>:
    RETURN TRY:
      // Validate before adding
      IF item.name.isBlank():
        RETURN Result.Error(
          IllegalArgumentException(),
          "Item name cannot be empty"
        )

      IF item.quantity < 0:
        RETURN Result.Error(
          IllegalArgumentException(),
          "Quantity cannot be negative"
        )

      SET id = dao.insertItem(item)
      Result.Success(id)
```

```
        CATCH exception: SQLiteConstraintException:
            Log.e(TAG, "Database constraint violated", exception)
            Result.Error(exception, "An item with this name already exists")
        CATCH exception: Exception:
            Log.e(TAG, "Failed to add item", exception)
            Result.Error(exception, "Unable to add item. Please try again.")
    END METHOD

END CLASS

// ViewModel handling results
CLASS InventoryViewModel(private repository: InventoryRepository) EXTENDS ViewModel

    PRIVATE val _uiState = MutableLiveData<InventoryUiState>()
    PUBLIC val uiState: LiveData<InventoryUiState> = _uiState

    METHOD loadInventory():
        SET _uiState.value = InventoryUiState.Loading

        viewModelScope.launch {
            WHEN (val result = repository.getAllItems()):
                IS Result.Success:
                    SET _uiState.value = InventoryUiState.Success(result.data)

                IS Result.Error:
                    SET _uiState.value = InventoryUiState.Error(
                        message = result.message,
                        canRetry = true
            )
        }
    END METHOD

    METHOD addItem(name: String, quantity: Int):
        SET _uiState.value = InventoryUiState.Loading

        viewModelScope.launch {
            SET item = InventoryItem(name = name, quantity = quantity)

            WHEN (val result = repository.addItem(item)):
                IS Result.Success:
                    SET _uiState.value = InventoryUiState.ItemAdded(item.name)
                    loadInventory() // Refresh list

                IS Result.Error:
                    SET _uiState.value = InventoryUiState.Error(
                        message = result.message,
                        canRetry = false
                    )
        }
    END METHOD

    METHOD retryLastOperation():
        loadInventory() // Retry loading
    END METHOD

END CLASS
```

```
// UI State sealed class
SEALED CLASS InventoryUiState
  OBJECT Loading EXTENDS InventoryUiState
  DATA CLASS Success(val items: List<InventoryItem>) EXTENDS InventoryUiState
  DATA CLASS ItemAdded(val itemName: String) EXTENDS InventoryUiState
  DATA CLASS Error(val message: String, val canRetry: Boolean) EXTENDS InventoryUiState
END CLASS


// Activity handling UI states
CLASS InventoryActivity EXTENDS AppCompatActivity


  METHOD observeUiState():
    viewModel.uiState.observe(this) { state ->
      WHEN (state):
        IS InventoryUiState.Loading:
          showProgressBar()
          hideErrorView()


        IS InventoryUiState.Success:
          hideProgressBar()
          hideErrorView()
          updateRecyclerView(state.items)


        IS InventoryUiState.ItemAdded:
          hideProgressBar()
          showSuccessSnackbar("${state.itemName} added successfully")
```

```
            IS InventoryUiState.Error:
                hideProgressBar()
                showErrorSnackbar(state.message, state.canRetry)
    }
END METHOD

METHOD showSuccessSnackbar(message: String):
    Snackbar.make(rootView, message, Snackbar.LENGTH_SHORT)
        .setBackgroundTint(colorSuccess)
        .setTextColor(colorOnSuccess)
        .show()
END METHOD

METHOD showErrorSnackbar(message: String, canRetry: Boolean):
    SET snackbar = Snackbar.make(rootView, message, Snackbar.LENGTH_LONG)
        .setBackgroundTint(colorError)
        .setTextColor(colorOnError)

    IF canRetry:
        snackbar.setAction("RETRY") {
            viewModel.retryLastOperation()
        }

    snackbar.show()

    // Announce error to screen reader
    rootView.announceForAccessibility(message)
END METHOD
```

END CLASS

For this category of enhancement, consider adding more advanced concepts of MySQL, incorporating data mining, creating a MongoDB interface with HTML/JavaScript, or building a full stack with a different programming language for your artifact. These are just recommendations; consider being creative and proposing an alternative enhancement to your instructor. Note: You only need to choose one type of enhancement per category.

Think about what additions to include to complete the enhancement criteria in this category. Since one example option is to port to a new language, that is the kind of scale that is expected. Perhaps you might increase the efficiency and time complexity of an algorithm in an application and detail the logic of the increased time complexity. Remember, you do not need to port to a new language but instead have an equivalent scale of enhancement. Underlying expectations of any enhancement include fixing errors, debugging, and cleaning up comments, but these are not enhancements themselves.

iii. Explain how the planned enhancement will **demonstrate** specific **skills** and align with course outcomes.

   a. Identify and describe the specific skills you will demonstrate that align with the course outcome.

These Software Engineering and Design enhancements demonstrate the following professional competencies:

1. Modern Mobile Architecture: Implementing MVVM pattern with Android Jetpack components shows understanding of industry-standard architectural patterns that promote testability, maintainability, and separation of concerns

2. User-Centered Design: Creating accessible interfaces that work for all users, including those with disabilities, demonstrates commitment to inclusive design and compliance with accessibility standards (WCAG 2.1)

3. Material Design Expertise: Applying Material Design 3 principles shows knowledge of platform-specific design guidelines and modern UI/UX best practices

4. Error Handling and Resilience: Implementing comprehensive error handling with user-friendly feedback demonstrates defensive programming and production-ready code quality

5. Code Organization: Separating concerns across multiple layers (View, ViewModel, Repository, DAO) shows understanding of clean architecture principles

6. Reactive Programming: Using LiveData and coroutines demonstrates knowledge of modern Android development patterns for asynchronous operations

7. Professional Communication: Writing code that is self-documenting with clear naming conventions and proper accessibility labels for screen readers

Course Outcome Alignment:

Course Outcome 1 (Employ strategies for building collaborative environments):

Modular Architecture: MVVM separation allows multiple developers to work on different layers simultaneously without conflicts

Clear Documentation: Content descriptions and well-named classes/methods enable team understanding

Testable Code: ViewModels can be unit tested independently, facilitating code review and quality assurance

Repository Pattern: Abstracts data sources, allowing easy swapping of implementations for testing or future changes

Course Outcome 2 (Design, develop, and deliver professional-quality communications):

User Feedback: Clear, actionable error messages and success confirmations guide users effectively

Accessibility: Screen reader support ensures all users can interact with the application

Visual Communication: Material Design 3 provides consistent, professional visual language

Code as Communication: Well-structured architecture communicates design intent to other developers

Course Outcome 5 (Develop a security mindset):

Input Validation: Validating data in ViewModel before database operations prevents malformed data

Error Messages: User-friendly messages don't expose system internals or sensitive information

Separation of Concerns: Isolating data access reduces attack surface for injection vulnerabilities

Lifecycle Awareness: Proper handling of Android lifecycle prevents memory leaks and data exposure

b. Select one or more of the course outcomes listed under Category One that your enhancement will align with.

Design, develop, and deliver professional-quality oral, written, and visual communications that are coherent, technically sound, and appropriately adapted to specific audiences and contexts.

IV. ePortfolio Overall Skill Set

A. Accurately describe the **skill set** to be illustrated by the **ePortfolio overall**.

i. Skills and outcomes planned to be illustrated in the code review

These enhancements demonstrate:

Software Architecture: Designing scalable, maintainable service patterns and component hierarchies

User Experience Design: Creating intuitive, user-friendly interfaces with clear feedback

Accessibility Standards: Implementing WCAG guidelines for inclusive web applications

Code Reusability: Building reusable components and services

Defensive Programming: Anticipating and handling error conditions gracefully

Course Outcomes Alignment

Course Outcome 1 (Employ strategies for building collaborative environments):

Well-documented code with clear comments and README updates

Component-based architecture enables team collaboration

Consistent coding standards and naming conventions

Course Outcome 2 (Design, develop, and deliver professional-quality communications):

User-facing error messages are clear, professional, and actionable

Error messages don't expose sensitive system information

Form validation enforces business rules and data integrity

ii. Skills and outcomes planned to be illustrated in the narratives

Each enhancement narrative will demonstrate:

Problem Identification: Clearly articulating the specific technical debt or missing functionality in the current implementation

Solution Design: Explaining architectural decisions (MVVM, Repository pattern, Room) with trade-off analysis

Implementation Details: Documenting code transformations with before/after comparisons showing measurable improvements

Algorithm Analysis: For Category Two, include Big O complexity analysis comparing current vs. enhanced approaches

Database Design: For Category Three, document schema normalization, relationship diagrams, and query optimization strategies

Testing and Validation: Describing unit tests, UI tests, and manual testing procedures to verify enhancements

Reflection: Discussing implementation challenges (e.g., database migration complexity) and solutions discovered

Course Outcome Alignment: Explicitly mapping each enhancement to specific CS 499 course outcomes with concrete examples

Security Considerations: Highlighting security improvements in each enhancement category

Future Improvements: Identifying potential next steps (e.g., cloud sync, barcode scanning, analytics dashboard)

iii. Skills and outcomes planned to be illustrated in the professional self-assessment

The professional self-assessment will showcase:

Self-Awareness: Understanding of my evolution from beginner Android developer to practitioner of modern mobile architecture patterns

Career Vision: Clear articulation of career goals in mobile application development with security specialization

Program Reflection: Synthesis of learning throughout the CS program, connecting CS-360 mobile development to broader computer science principles

Professional Identity: Positioning myself as a mobile-first developer who understands full-stack principles and can work across platforms

Continuous Learning: Commitment to staying current with Android development trends (Jetpack Compose, Kotlin coroutines, modern architecture)

Industry Readiness: Demonstrating skills directly applicable to junior/mid-level mobile developer positions