Week 5: Self-Guided Plan & Submission Requirements

How to Perform the Activity (Self-Guided)

Phase 1: Mid-Project Review (Full Group Responsibility)

This phase is non-negotiable. You cannot build new modules on a shaky foundation.

Re-Run All Integration Tests: Gather the Integration Test Reports from Weeks 3 and 4 and re-run all tests (e.g., Procurement → Finance, HR → Finance). Confirm that all four core modules (1, 3, 5, 10) are still communicating without errors.

Audit and Debug: Systematically check the Git history and code of Modules 1, 3, 5, and 10 for any outstanding bugs or inefficiencies. All groups must dedicate time to fixing any issues identified.

Finalize Core APIs: Ensure that the APIs for Inventory, Procurement, Finance, and HR are stable and documented, as the new modules will rely entirely on **reading** data from them.

Phase 2: Development Launch (Modules 2 & 8)

The goal of this phase is to build the initial structure of these two new modules, focusing heavily on integrating them with the established core modules.

Module 8 (Sales and Customer Support) → Core Modules:

> **Database Setup:** Create core tables for Leads, SalesOrders, and Quotes.
>
> **Crucial Read Integration:** Implement API calls to **Module 1 (Inventory)** to check current stock levels before creating a sales order.
>
> **Crucial Write Integration:** Implement the logic to call the **Module 1 (Inventory)** API to decrement stock levels when a sale is finalized.

Module 2 (Customer Service) → Core Modules:

> **Database Setup:** Create core tables for Tickets, Issues, and Solutions.
>
> **Crucial Read Integration:** Implement API calls to **Module 8 (Sales)** (once built) to look up a customer's order history and status. Implement calls to **Module 5 (Finance)** to confirm a customer's payment status or invoice details.

What to Be Submitted & Format

Your submission for Week 5 must include two **Physical Printouts** and the **Digital Git Link**.

Submission Type	Deliverable	Description
Physical Printout	**1.Mid-Project Review Report**	A formal report summarizing the health of your existing system. This must include: (1) A statement confirming all old integration tests passed, and (2) A list of any bugs found in Modules 1, 3, 5, or 10, and a description of the fix applied.
Physical Printout	**2. Module 2 & 8 Integration Plan**	Documentation of the new database schemas for Modules 2 and 8, focusing on the new API calls they will make to *read* data from the core modules (1, 3, 5, 10).
Digital Submission	**3. Source Code (Git Link)**	A link to your team's Git repository. This must contain the initial code for Modules 2 and 8, as well as the committed fixes made to the older modules (1, 3, 5, 10).

ERP System Health Report

Group Number: 3

Date of Report: October 13, 2025

Confirmation of Integration Tests

All previous integration tests from Weeks 3 and 4 were successfully re-run without errors. These tests covered critical data flows, including:

- Procurement (Module 3) to Inventory (Module 1): Approved requisitions generating purchase orders, goods receipts updating inventory stock, stock-in transaction logs, low stock alert clearance, and rejected requisitions not affecting inventory.
- Procurement (Module 3) to Finance (Module 5): Purchase order creation, invoice generation, invoice status updates, updated status reflections, total invoice amounts divided by status, and detailed invoice information.
- Inventory (Module 1) to Finance (Module 5): Stock-in transactions increasing inventory assets, stock-out transactions decreasing inventory assets, and stock-out exceeding quantity being rejected.
- Human Resources (Module 10) to Finance (Module 5): Payroll data communication and processing.

All test cases passed as originally documented, demonstrating that the four core modules continue to communicate seamlessly. No regressions were observed, and the system remains stable for further development.

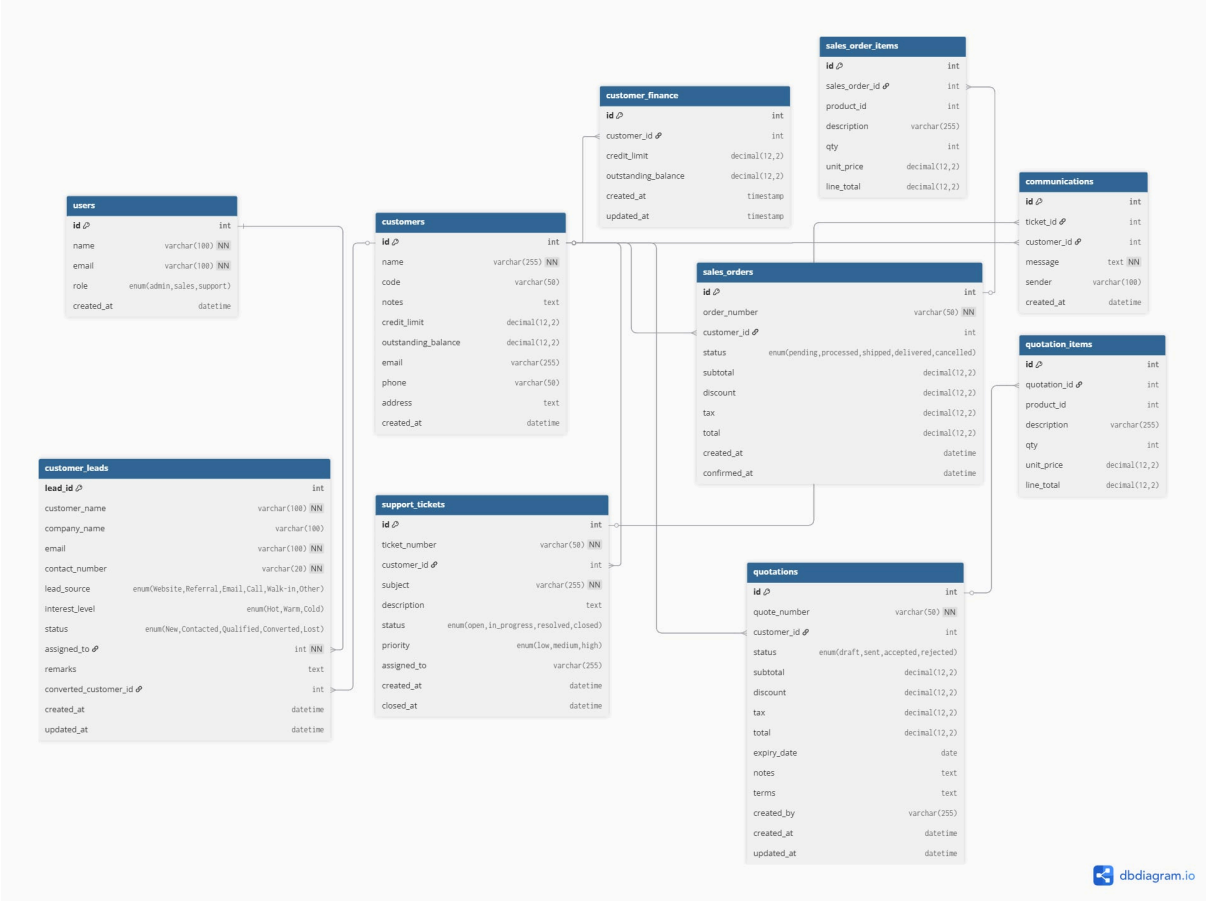
Identified bugs

During the audit of the Git history and codebase for Modules 1, 3, 5, and 10, no critical or major bugs were identified. A thorough review of commit history, database schemas, API endpoints, and integration logic confirmed that the system is free of outstanding issues.

Github Repositories Link: https://github.com/ElykRM/G3

Module 2 & 8 Integration Plan

Module 8 (Sales and Customer Support) - Database Schema & Integration



Database Schema: The schema for Module 8 is comprehensive and logically designed. Key tables include:

- customer_leads for tracking potential customers.
- customers as the central customer record.
- quotations and quotation_items for managing quotes.
- sales_orders and sales_order_items for final orders.
- support_tickets and communications for customer support.

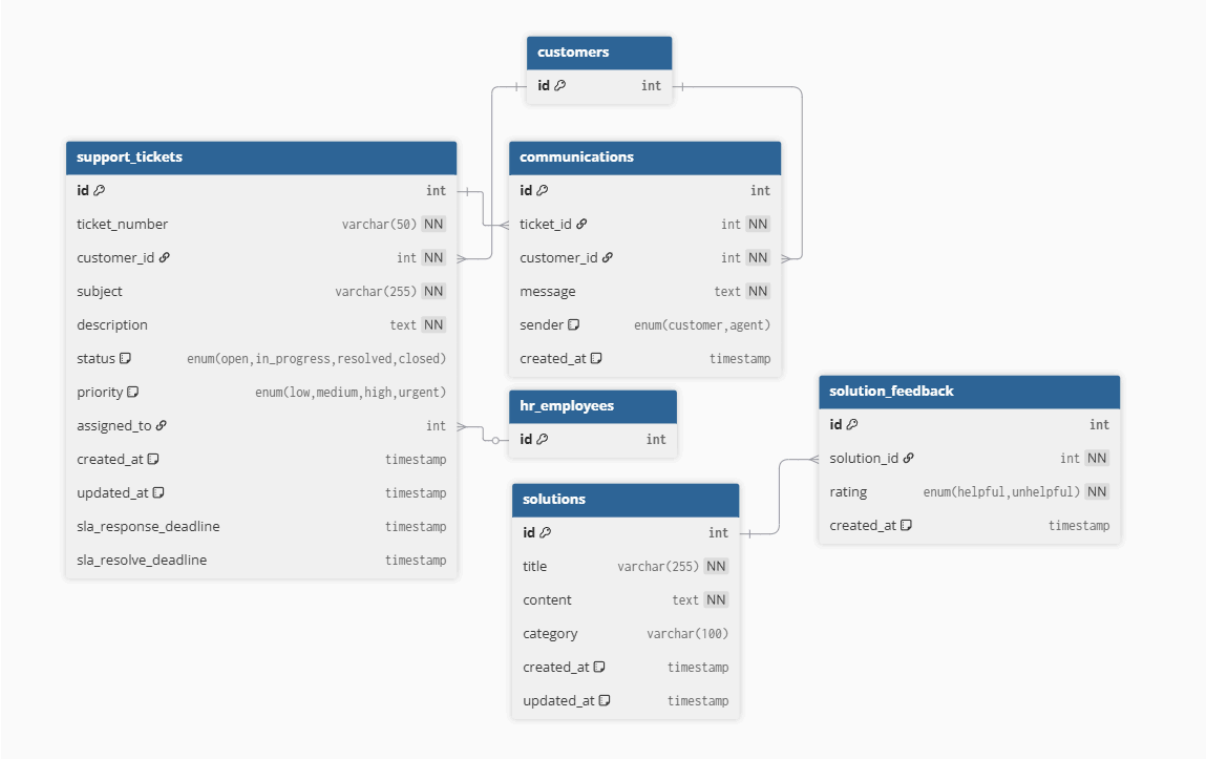
Integration Design (API Calls):

Read from Module 1 (Inventory): Before creating a sales order or quotation, Module 8 will call GET /api/inventory/products/{id} to check real-time stock levels and product availability.

Write to Module 1 (Inventory): Upon finalizing a sales order (status changed to 'processed'), Module 8 will call POST /api/inventory/stock/decrement to reduce the available stock quantity.

Read from Module 5 (Finance): Before confirming a sale for a customer, Module 8 will call GET /api/finance/customers/{id}/credit-status to verify the customer's credit limit and outstanding balance.

Module 2 (Customer Service) - Database Schema & Integration



Database Schema: The schema for Module 2 is focused and efficient. Key tables include:

- support_tickets for logging and tracking customer issues, enhanced with SLA deadline fields.
- communications to maintain a full history of customer-agent interactions.
- solutions and solution_feedback for managing a knowledge base.
- Integration Design (API Calls):

Read from Module 8 (Sales): To assist a customer, Module 2 will call GET /api/sales/customers/{id}/orders to retrieve the customer's complete order history and status, providing context for the support ticket.

Read from Module 5 (Finance): For billing-related inquiries, Module 2 will call GET /api/finance/customers/{id}/invoices to fetch invoice details and payment status.

Week 5 Group Submission Rubric

This rubric evaluates the overall project health, quality of documentation, and effectiveness of version control for the entire team's work in Week 5.

Integration Test Report - Mid-Project Review

Group Number: [3] **Date of Submission:** [14/10/25]

Category	Criteria	Points Possible
I. Mid-Project Review Report (Physical Printout)	**Total: 40 points**	
System Stability & Testing	**Excellent (18-20 points):** Report clearly states that all previous integration tests (Wks 3 & 4) were successfully re-run. No critical issues were found, demonstrating proactive maintenance. **Satisfactory (14-17 points):** Tests were re-run, but minor bugs were found. All issues were documented and promptly fixed within the week. **Needs Improvement (0-13 points):** Tests were not re-run, or major, system-breaking bugs were found in existing modules and were not resolved.	20
Bugs Found & Fixed	**Excellent (10 points):** Comprehensive summary of any bugs found in Modules 1, 3, 5, or 10, with clear commit references to the applied fixes. **Satisfactory (7-9 points):** Minor bugs were listed and fixed, but the documentation is vague or lacks clear links to the Git commits. **Needs Improvement (0-6 points):** Bugs were ignored, or the report fails to mention an audit of the existing codebase.	10
Formatting & Clarity	The report strictly adheres to all formatting guidelines (A4, 1-inch margins, Arial 11). Content is professionally written.	10

Category	Criteria	Points Possible
II. New Module Integration Plan (Physical Printout)	**Total: 30 points**	
Module 2 & 8 Schema	Clear, complete, and logically designed database schemas for the new Customer Service (2) and Sales (8) modules.	15
Integration Design	The document explicitly details the API endpoints (Read/Write) that Modules 2 and 8 will consume from the core modules (1, 3, 5, 10).	15
III. Digital Submission: Git Repository Health	**Total: 30 points**	
Code Organization	The repository is logically structured, making it easy to find code for Modules 1, 2, 3, 5, 8, and 10.	10
Version Control Quality	All members made consistent, meaningful contributions. Commit messages are descriptive and easy to follow throughout the week.	10
Integration Visibility	The committed code for Modules 2 and 8 clearly shows the implementation of the new integration calls as defined in the plan.	10
Total Group Score		**100**

Submission Format Requirement (Printout)

All printed documents must adhere to the project standard: **A4 bond paper**, **1-inch margin on all sides**, and **Arial 11 font**.

Individual Rubric for Module 2 (Customer Service)

This evaluates the student's ability to build the Customer Service module and use it to query information from other parts of the ERP.

Assigned Member's Full name: Kyle Raven P. Mabingnay Date Submitted: 14/10/25

Group Number: 3

Category	Criteria	Points Possible
I. Core Functionality	**Total: 40 points**	
Ticket Management CRUD	Did the student successfully implement the basic functions (Create, Read, Update, Delete) for customer service tickets?	20
Database Schema	Is the database design logical and complete for managing tickets, solutions, and employee assignment records?	20
II. Integration (Read Access)	**Total: 30 points**	
Sales Data Query (Read)	Did the student successfully implement API calls to query **Module 8 (Sales)** to retrieve a customer's order history or status based on their ID?	15
Core Data Query (Read)	Did the student successfully implement API calls to query **Module 5 (Finance)** for billing/payment status, or **Module 1 (Inventory)** for product status?	15
III. Project Management & Code Quality	**Total: 30 points**	
Version Control (Git)	Consistent use of Git with clear commit messages for the development of the new Module 2.	15
Code Quality	Clarity, readability, and adherence to team coding standards in the new module's code.	15
Total		**100**

Individual Rubric for Module 8 (Sales and Customer Support)

This evaluates the student's ability to build the Sales module and link it to Inventory and Finance for real-time validation and updates.

Assigned Member's Full name: Gabriel Ian De Leon Date Submitted: 14/10/25

Group Number: 3

Category	Criteria	Points Possible
I. Core Functionality	**Total: 40 points**	
Sales Order CRUD	Did the student successfully implement the basic functions (Create, Read, Update, Delete) for sales orders and quotes?	20
Database Schema	Is the database design logical and complete for managing sales orders, customer leads, and quotes?	20
II. Integration (Read & Write Access)	**Total: 30 points**	
Inventory Validation (Read/Write)	Did the student implement a crucial API call to **Module 1 (Inventory)** to check stock *before* confirming an order and *write* back to deduct stock upon finalization?	15
Finance/Customer Status Query (Read)	Did the student implement API calls to query **Module 5 (Finance)** to check a customer's current credit status or payment history?	15
III. Project Management & Code Quality	**Total: 30 points**	
Version Control (Git)	Consistent use of Git with clear commit messages for the development of the new Module 8.	15
Code Quality	Clarity, readability, and adherence to team coding standards in the new module's code.	15
Total		**100**