



Module Code & Module Title
CS5002NT– Software Engineering

Assessment Type
20% Group Coursework

Semester
2024-2025 Autumn

Group Name: Group D
Group members

London Met ID	Student Name
23049202	Ashika Nepal
23049235	Dipin Karki
23049226	David Basnet
23049216	Bidur Siwakoti
23049253	Ishwor Dhakal

Assignment Due Date: 2025/01/08
Assignment Submission Date:2025/01/08
Word Count:10418

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.

Group L2C1 GD



Islington College,Nepal

Document Details





Submission ID	trn:oid:::3618:78334022	85 Pages
Submission Date	Jan 8, 2025, 11:47 PM GMT+5:45	9,070 Words
Download Date	Jan 8, 2025, 11:49 PM GMT+5:45	55,717 Characters
File Name	very very final software	
File Size	66.9 KB	

Page 1 of 85 - Cover Page




19% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Match Groups

-  **163** Not Cited or Quoted 19%
Matches with neither in-text citation nor quotation marks
-  **6** Missing Quotations 0%
Matches that are still very similar to source material
-  **0** Missing Citation 0%
Matches that have quotation marks, but no in-text citation
-  **0** Cited and Quoted 0%
Matches with in-text citation present, but no quotation marks

Top Sources

- 4%  Internet sources
- 1%  Publications
- 19%  Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- 163Not Cited or Quoted 19%

Matches with neither in-text citation nor quotation marks
- 6 Missing Quotations 0%

Matches that are still very similar to source material
- 0 Missing Citation 0%

Matches that have quotation marks, but no in-text citation
- 0 Cited and Quoted 0%

Matches with in-text citation present, but no quotation marks

Top Sources

- 4%

Internet sources
- 1%

Publications
- 19%

Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Submitted works	
islingtoncollege on 2025-01-02		1%
2	Submitted works	
islingtoncollege on 2024-12-24		1%
3	Submitted works	
islingtoncollege on 2024-12-24		1%
4	Submitted works	
islingtoncollege on 2024-12-24		1%
5	Submitted works	
islingtoncollege on 2024-12-24		1%
6	Submitted works	
islingtoncollege on 2024-12-24		1%
7	Submitted works	
islingtoncollege on 2024-12-24		0%
8	Submitted works	
islingtoncollege on 2024-12-24		0%
9	Internet	
www2.aston.ac.uk		0%
10	Submitted works	
islingtoncollege on 2024-12-24		0%

islingtoncollege on 2024-12-24 0%

13 Submitted works

islingtoncollege on 2024-12-24 0%

14 Submitted works

islingtoncollege on 2024-12-24 0%

15 Submitted works

islingtoncollege on 2024-12-24 0%

16 Submitted works

islingtoncollege on 2024-12-24 0%

17 Submitted works

islingtoncollege on 2024-12-24 0%

18 Internet

www.coursehero.com 0%

19 Submitted works

Asia Pacific University College of Technology and Innovation (UCTI) on 2021-12-02 0%

20 Submitted works

islingtoncollege on 2024-12-24 0%

21 Submitted works

islingtoncollege on 2024-12-24 0%

22 Submitted works

islingtoncollege on 2024-12-24 0%

23 Submitted works

Gusto University on 2024-11-07

24 Submitted works

Table of Contents

1. Introduction to the Report.....	1
2. Business Case	2
2.1 Problem Statement	2
2.2 Objectives	3
2.3 Scope.....	3
2.4 Benefits	4
2.5 Costs and Risks	5
2.6 Timelines.....	7
3. Software Requirement Specification (SRS)	8
3.1 Purpose of SRS	8
3.2 Intended Audience of SRS	8
3.3 Definitions, Acronyms and Abbreviations	9
3.4 Functional Requirements	9
3.4.1 Inventory Tracking	10
3.4.2 Order Management	10
3.4.3 QR Code Scanning.....	11
3.4.4 User Role Management.....	11
3.4.5 Reporting and Analysis.....	12
3.4.6 Integration with ERP	12
3.4.7 Notification System.....	13
3.4.8 Multi-Warehouse Support	13
3.4.9 Audit Trial	14
3.4.10 Backup and Recovery	14
3.5 External Interface Requirements.....	15

3.5.1	Assumptions made during Build	16
3.5.2	Assumptions made to use software	16
3.6	Other non-functional requirements	16
3.7	Other requirements	18
4.	Group Tasks.....	21
4.1	Environmental Module Specification	21
4.1.1	Level 0 DFD.....	21
4.1.2	Level 1 DFD.....	23
4.1.3	Level 2 DFD.....	25
4.2	Internal Module Specification	28
4.2.1	Entity-Relationship Diagram	28
4.2.2	Data Dictionary	29
4.2.3	Process Specification	35
4.3	Design Specification.....	38
4.4	Progress Logs	39
4.5	Group Responsibilities	44
5.	Individual Task: Bidur Siwakoti.....	45
5.1	Environmental Model Specification	45
5.1.1	Context Diagram/ Level 0 DFD.....	45
5.2	Internal Model Specification	46
5.2.1	Level 1 DFD.....	46
5.2.2	Level 2 DFD.....	47
5.3	Design Specification.....	48
5.3.1	Structure Charts	48
5.4	Module Specification	49

6. Individual Task: Ishwor Dhakal.....	53
Feature:Report Preparation.....	53
6.1 Environmental Module Specification	53
Context level DFD.....	53
6.2 Internal Module Specification	54
Level 2 DFD	55
6.3 Design Specification.....	56
6.3.1 Module Specification	57
7. Individual Task: David Basnet.....	60
7.1 Environmental Module Specification	60
7.2 Internal Module Specification	61
Level 2 DFD	63
7.3 Design Specification.....	64
7.4 Module Specification	65
8. Individual Task: Ashika Nepal	70
Feature: Dispatch Order.....	70
8.1 Environmental Module Specification	70
Context level DFD.....	70
8.2 Internal Module Specification	71
Level 1 DFD	71
Level 2 DFD	72
8.3 Design Specification.....	73
8.4 Module Specification	74
9. Individual Task: Dipin Karki.....	77
Feature: Payment.....	77

9.1	Environmental Module Specification	77
	Context level DFD.....	77
9.2	Internal Module Specification	78
9.3	Design Specification.....	80
9.4	Module specification:.....	81
10.	Conclusion.....	83
11.	References	84

Tables of Figures:

Figure 1 Context Level DFD (Overall)	21
Figure 2 Level 1 DFD (Overall)	23
Figure 3 User Registration (Overall DFD)	25
Figure 4 Report Generation (Overall DFD).....	26
Figure 5 Purchase Order (Overall DFD).....	27
Figure 6 Entity-Relationship Diagram.....	28
Figure 7 Structure Chart (Overall	38
Figure 8: Context Level Diagram (Purchase Order).	45
Figure 9: Level 1 DFD (Purchase Order).....	46
Figure 10: Level 2 DFD (Purchase Order).....	47
Figure 11: Structure Chart (Purchase Order).	48
Figure 12 Context level DFD (Report Preparation)	53
Figure 13 Level 1 DFD (Report Preparation).....	54
Figure 14 Level 2 DFD (Report Preparation).....	55
Figure 15 Structure Chart (Report Preparation)	56
Figure 16 Context level DFD (Real-time Stock Updates)	61
Figure 17 Level 1 DFD (Real-time Stock Updates)	62
Figure 18 Level 2 DFD (Real-time Stock Updates)	63
Figure 19 Structure Chart (Real-time Stock Updates).....	64
Figure 20 Context level DFD (Dispatch Order).....	70
Figure 21 Level 1 DFD (Dispatch Order).....	71
Figure 22 Level 2 DFD (Dispatch Order).....	72
Figure 23 Structure Chart (Dispatch Order)	73
Figure 24 Context Level DFD (Payment)	77
Figure 25: Level 1 DFD (Payment).....	78
Figure 26: Level 2 DFD (Payment).....	79
Figure 27 Structure Chart (Payment)	80

Table of Tables:

Table 1 Scope of Inventory Management System.....	3
Table 2 Estimated Costs for Inventory Management System.....	5
Table 3 Timelines for Inventory Management System	7
Table 4 Inventory Tracking (Functional Requirement)	10
Table 5 Order Management (Functional Requirement)	10
Table 6 QR Code Scanning (Functional Requirement)	11
Table 7 user role management (Functional Requirement)	11
Table 8 Reporting and Analysis (Functional Requirement)	12
Table 9 Integration with ERP (Functional Requirements).....	12
Table 10 Notification System (Functional Requirements).....	13
Table 11 Multi-Warehouse Support (Functional Requirements).....	13
Table 12 Audit Trail (Functional Requirements)	14
Table 13 Backup and Recovery (Functional Requirements)	14
Table 14 All Non-functional Requirements	16

1. Introduction to the Report

The Inventory Management System (IMS) aims at managing the operations of the warehouse successfully and efficiently with the help of fundamental procedures including inventory control, order processing and supply chain management. To enhance the relationship between customers and the warehouses in a way to improve customer's satisfaction by boosting order processing and delivery efficiency. IMS thus aims to capture the customer's attention by satisfying the needs of both customers and warehouses. The technology is very adaptable and enables the warehouses to monitor the customer's orders and inventory levels. It also identifies the best locations to store and distribute the products. This feature facilitates the warehouse's efficient operations, particularly when dealing with complicated processes. IMS ensures that there is no waste and that what is needed at the right time by using resources properly. One of the major objectives of the system is to integrate with the various teams within the warehouse. This lowers the potential for errors and enhances the cooperation between the different branches of the company. The approach makes the supply chain and demand clear by providing the real-time information on orders and supplies. IMS removes most risks and challenges that are associated with the warehouse management by providing clarity and real-time tracking.

All the features of the IMS have been developed with careful planning and analysis to make sure that warehouses could use the system. Addressing the system reduces the chances of making mistakes and fixes issues from the past. This enhancement thus promotes better decision-making, proper utilization of resources and improved customer satisfaction. Thus, by utilizing advanced tools and technologies, the system provides improved performance and efficient work. In addition to enhancing warehouse management capacity, IMS also improves the customer satisfaction by ensuring reliable information and fast delivery. It works well for warehouses seeking to be sustainable in the market because of its emphasis on analysis and continuous improvement.

2. Business Case

Global Technology Corporation is currently working on a project to implement an innovative Inventory Management System (IMS) to enhance its warehouse functions. The goal of this initiative is to solve the problems and challenges that occurred during the implementation of IMS as an outcome of poor risk management processes, project planning and documentation. Drawing lessons from experience is at the core of this new initiative, which emphasizes the importance of solid project management strategies and clearly defined goals to deliver tangible outcomes that are in line with the company's overachieving objectives. This document outlines the problem statement, objectives, scope, benefits, estimated costs, potential risks, and a detailed timeline for the project.

2.1 Problem Statement

During the previous phase, global tech corporations faced with difficulties with the inventory management system due to crucial project management aspects like:

- **Undefined scope and goals:** No detailed documentation outlining the different characteristics expected from the system. Integration problem or end goals which meant a lot of misunderstanding and wastage of effort.
- **Insufficient resource management:** Budgets, timetables, and manpower were misallocated, resulting in delays and cost overruns that had impact on other necessary operations.
- **Poor communication structures:** Information was shared throughout teams in an unexpected manner, resulting in disjointed activities and missed deadlines.
- **Disregarded of Risk Management:** No preparation was made in identifying possible risks of the system, including integration problems that caused interrupts in operations.

2.2 Objectives

The following IMS implementation project has specified the following goals:

- **Establishing a new IMS:** creating and implementing an IMS with specific characteristics, integration points and aims that fulfil the operational requirements of the company.
- **Timelines and cost-effectiveness:** complete the project in scheduled time and within budget to enhance resource efficiency.
- **Improving the efficiency of the operations of warehouse:** downtime will be reduced; resources will be maximized and improvement in performance of the warehouse.
- **Enhancement of interdependence and inter-project communications:** exchange of information and collaboration efforts between the various department and project teams
- **Risk management and improved integration:** Explore the identification and handling of potential risk as an opportunity so that improvements can be built into the process.

2.3 Scope

The scope of the IMS project is as follows:

Table 1 Scope of Inventory Management System

Components	In Scope	Out Scope
User Access	Admin and customer (buyer) registration and login functionality.	External user roles beyond admin and buyers
Purchase Orders	Add and view purchase orders from suppliers.	Vendor management or contract handling.
Report Generation	Sales and purchase reports, profit-loss analytics for admins and buyers.	Advanced predictive analytics.

Sales Management	Viewing sales orders, delivery methods, and dispatching details.	Integration with third party logistics
Product Management	Adding, viewing, and managing product inventory	Managing product design or manufacturing details.
Payment System	Price comparison, secure payment options, and data storage.	Integration with multiple external payment gateways.

2.4 Benefits

It is anticipated that the following benefits will follow through after the successful completion of the IMS:

It is anticipated that the following benefits will follow through after the successful installation of the IMS:

- **Decreased operational inefficiencies:** The IMS will assist in the integration of the workflow and the standardization of all warehouse processes that will enhance the minimum errors and waste while increasing efficiency by 30%, optimizing overall productivity.
- **Cost Savings:** Reducing inefficiencies and preventing resource wastage will save the company up to \$100,000 annually, representing an estimated 20% decrease in operational costs.
- **Enhanced customer satisfaction:** efficient order transactions will be faster and more accurate inventory management, which improves customer satisfaction levels by 30% due to quicker response time and improved accuracy.
- **Improved team collaboration:** People in various industries can work collaboratively while communication and interaction are encouraged which aims to expect improvement by 20% for better interaction and coordinated efforts.

- **Reduced risk of downtimes:** System unavailability is expected to be extremely low by 90%, as project management and risk management strategies is improved.
- **Gained Insights:** By emphasizing project output and challenges that have arisen, following initiatives can more effectively organized to capitalize on best practices and scale and the improvement made which is expected to lead by 40% improvement.

2.5 Costs and Risks

Estimated Costs:

Table 2 Estimated Costs for Inventory Management System

Cost Item	Description	Estimated Cost (NPR)
IMS Software Development	Designing and developing the custom IMS solution, including programming, system architecture, and integration with existing systems.	NPR 15,600,000
Hardware and Infrastructure	Purchase of necessary hardware (e.g., servers, network upgrades) to support the IMS.	NPR 5,600,000
Training and Documentation	Conducting training sessions for staff.	NPR 3,000,000
System Testing and Quality Assurance	Performing system testing and quality assurance to ensure the IMS works properly.	NPR 1,800,000
Deployment and Setup	Costs associated with system deployment, including temporary	NPR 3,600,000

	staffing, hardware installation, and setup.	
Ongoing Maintenance and Support	Annual maintenance, including software updates, bug fixes, and technical support.	NPR 1,800,000

Risk and Mitigation Strategies

Delays in requirements to timeline due to lack of clarity: There may be unintended consequences in terms of timeline due to incomplete or unclear requirements.

Mitigation: Solicit and document clear project requirements at the beginning of the project and at any other significant project milestones.

Scope expansion: Budget deficits and time lags could arise from alterations to project scope.

Mitigation: Adopt proper change management procedures to review and authorize any changes made to the scope.

Change Management: Some stakeholders are likely to not support the new system owing to lack of appreciation for it or due to its complexity.

Mitigation: Training should be made available, and the personnel should be brought on board with the system in a gradual manner.

System problems: Operational difficulties may be encountered when integrating and after the system has been implemented.

Mitigation: Plan for adequate testing including deployment of technical support expertise to mitigate expected technical challenges.

2.6 Timelines

The timelines of the IMS project are as follows:

Table 3 Timelines for Inventory Management System

Month	Activity	Out Scope
Month 1	Gathering requirements and project scope documentation.	With the support of stakeholder workshops, the project's goals and scope will be defined.
Month 3	Design and development of the system	IMS will be developed, and the progress will be assessed to meet the above objectives.
Month 4	Integration and Testing	The issues identified will be fixed and IMS will be linked with the Warehouse Management System.
Month 5	Training and Final Adjustments	It also involves training the stakeholders and the system must be enhanced based on the feedback of stakeholders.
Month 6	Final deployment and analysis after implementation	Implement IMS into practise and access for data that will be useful for upcoming projects.

3. Software Requirement Specification (SRS)

The software Requirement Specification (SRS) is a document that has all the information about the system's functional and non-functional requirements. It provides a common platform for understanding between the stakeholders including the clients, developers, project managers and testers as to what the project objectives, details and limitations are. The SRS provides a strong foundation for planning and decision-making by minimizing the possibility of misinformation, outlining the development process, allowing testing and decreasing related risks. (Barney, 2023)

3.1 Purpose of SRS

The purpose of SRS is to:

- Clearly outline the goals and scope of the system.
- To define the planned functionality and design of the system.
- Help in efficiency allocating resources and avoid delays or additional expenses.
- Improve communication by providing a common reference for everyone involved.
- Enable the identification of potential risks in course of the activity and the development of strategy to address them.
- To serve as a basis for guiding and ensuring the quality testing assurance procedures to guarantee that the system operates as intended.
- To serve as a reference for the system's development and for the support.

3.2 Intended Audience of SRS

The SRS is intended for various stakeholders:

Project Managers: To plan, monitor, and manage the project.

Developers: To meet the needs for the system's installation.

Testers/QA Teams: To create test data and check the feasibility of the system.

Clients/Stakeholders: To make sure that their requirements and expectations are in line with what has been documented.

System Architects: To formulate the system architecture that would meet these requirements.

Maintenance Teams: To refer during future enhancement or when troubleshooting.

Regulatory Authorities: To confirm with the set standards and legal requirements.

3.3 Definitions, Acronyms and Abbreviations

Definitions:

System Downtime: A situation where the system is unavailable for use.

Stakeholders: These are the people or organizations that may have an interest in this project.

Acronyms and Abbreviations:

SRS: Software Requirements Specification

IMS: Inventory Management System

QA: Quality Assurance

UI: User Interface

WCAG: Web Content Accessibility Guidelines

GDPR: General Data Protection Regulation

API: Application Programming Interface

3.4 Functional Requirements

This section outlines the functional requirements of the system, categorized unique IDs, functions, descriptions, and their priority levels.

3.4.1 Inventory Tracking

Table 4 Inventory Tracking (Functional Requirement)

ID	Function name	Description	Priority
FR-3.4.1.1	Real-time level tracking	Tracks the real time inventory levels which updates whenever items are added or removed.	High
FR-3.4.1.2	Location and Status Update	Monitor inventory locations and product status for better visibility.	High

3.4.2 Order Management

Table 5 Order Management (Functional Requirement)

ID	Function Name	Description	Priority
FR-3.4.2.1	Order Automation	Automates the process of creating and updating the customers and vehicle's orders	High
FR-3.4.2.2	Order Tracking	Allows the real time tracking of orders from placement until fulfilment of orders	High

3.4.3 QR Code Scanning

Table 6 QR Code Scanning (Functional Requirement)

ID	Function Name	Description	Priority
FR-3.4.3.1	Product Location Management	Use QR code to quickly locate and update the details of product	Medium
FR-3.4.3.2	Inventory Accuracy	Helps in error free inventory management using QR code scanning	medium

3.4.4 User Role Management

Table 7 user role management (Functional Requirement)

ID	Function Name	Description	Priority
FR-3.4.4.1	Role Creation	Allow administrators to create, edit, and delete user roles.	High
FR-3.4.4.2	Permission Assignment	Grant permissions to different roles to control access to different system features	High

3.4.5 Reporting and Analysis

Table 8 Reporting and Analysis (Functional Requirement)

ID	Function Name	Description	Priority
FR-3.4.5.1	Inventory Reporting	Generates report on different metrics like inventory management and order performance	Medium
FR-3.4.5.2	Business Analytics	Provide analytics on product turnover and operational efficiency	Medium

3.4.6 Integration with ERP

Table 9 Integration with ERP (Functional Requirements)

ID	Function Name	Description	Priority
FR-3.4.6.1	Data Sharing	Shares inventory and order data with ERP system flawlessly	High
FR-3.4.6.2	Real-time synchronization	Ensure real-time synchronization between the system and ERP.	High

3.4.7 Notification System

Table 10 Notification System (Functional Requirements)

ID	Function Name	Description	Priority
FR-3.4.7.1	Stock Level Alerts	Notify users if there is low stock for any items to avoid disruption	Medium
FR-3.4.7.2	System Error Alerts	Alert users about the errors related to order processing or system malfunctions	Medium

3.4.8 Multi-Warehouse Support

Table 11 Multi-Warehouse Support (Functional Requirements)

ID	Function Name	Description	Priority
FR-3.4.8.1	Centralized Inventory Management	With the centralized management, it tracks inventory across multiple warehouses	Medium
FR-3.4.8.2	Warehouse Transfer Support	Support inter-warehouse transfers and inventory balancing	Medium

3.4.9 Audit Trial

Table 12 Audit Trial (Functional Requirements)

ID	Function Name	Description	Priority
FR-3.4.9.1	Activity Logging	Logs each inventory adjustment with details related to user and timestamp	Low
FR-3.4.9.2	Compliance Reporting	It provides exportable audit for review purpose	Low

3.4.10 Backup and Recovery

Table 13 Backup and Recovery (Functional Requirements)

ID	Function Name	Description	Priority
FR-3.4.10.1	Automated Backups	Schedule automated backups to prevent data loss.	High
FR-3.4.10.2	Data Recovery Protocols	Develop strong data recovery techniques for quick restoration	High

3.5 External Interface Requirements

This section identifies the interfaces through which the system will interact guarantee the effectiveness of the system.

User Interface

- The system is expected to have a good interface in the form of web-based interface to control and manage inventory.
- Mobile friendly interface to enable access using mobile phones and other smart devices such as tablets.
- Support for multiple languages is required.

Hardware Interface

- Co-operable with barcode/QR code scanners for the inventory input.
- RFID (Radio Frequency Identification) integration with the reader's functions if applicable.
- For data consolidation, it links required equipment like desktops.

Software Interface

- Data synchronization with ERP systems such as SAP or Oracle.
- Assistance of APIs link to e-commerce or third-party logistics platforms.
- Suitable for storing data in database like MySQL or PostgreSQL.

Communication Interface

- Secure data transmission and reception using HTTPS and APIs must be provided by the system.
- Using email and SMS gateways to send notifications and alerts.
- Verify scalability compatible with cloud platforms.

3.5.1 Assumptions made during Build

- Cloud-based operations will have reliable internet connectivity.
- Hardware, including barcode scanners, will be operable and pre-configured.
- Each of the legacy data will be delivered in a migration-compatible format.
- End users will receive sufficient guidance and documentation.
- The system will not function in areas with major environment restrictions like (no extreme heat or cold).

Cloud-based operations

3.5.2 Assumptions made to use software

- Users can operate the interface with the basic knowledge of technology.
- The system will be compliance with rules and policies of the organization.
- The performance of system's regular updates and maintenance.
- Customers will have access to compatible devices like (cell phones, computers).
- The company will allocate a dedicated team for system monitoring and support.

3.6 Other non-functional requirements

Table 14 All Non-functional Requirements

ID	Category	Description
NFR-3.5.1	Performance	The page should load in less than 3 seconds to ensure better user experience
NFR-3.5.2	Security	All the transmission of data should be encrypted using standard protocols like HTTPS.

NFR-3.5.3	Usability	The interface should be user-friendly, suitable for users with minimal technical knowledge.
NFR-04	Reliability	System should ensure 99.9% uptime excluding planned maintenance
NFR-05	Scalability	The system must be suitable to accommodate growth in users, inventory items, and data volumes without significant reconfiguration.
NFR-06	Maintainability	The system should have easy maintenance with clear documentation, modular design, and regular software updates.
NFR-07	Compliance	It should follow the local and international regulations on data protection such as GDPR or Nepal's Data Protection Act.
NFR-08	Accessibility	Users with disabilities should be able to access the system, following WCAG 2.1 guidelines
NFR-09	Interoperability	The system should be able to integrate with the various

		other business platforms with the help of APIs.
NFR-10	Environmental	The system should be designed in such a way that it should consume less energy and encourage sustainable usage practises.

3.7 Other requirements

Legal Requirements

Data Protection Compliance: The system should follow the data protection laws and regulations which include Nepal's Data protection Act. This means that user data privacy, data storage and management of user's consent are well protected.

Rights to intellectual Property: Unless otherwise specified, Global Tech Corporation will own all software, code and other sorts of intellectual property developed during this project.

Database Requirements

To store structured data: The system will include Relational Database Management Systems (RDBMS) like PostgreSQL, MySQL and others. The type of database should be selected according to the expected volume of data that will be stored, and the frequency of data should be taken into consideration.

Data Integrity: The database should also have measures such as Atomicity, Consistency, Isolation and Durability to make sure that data is not damaged or deleted during transaction.

Backup and recovery: Regular backups of generic databases are recommended, and a recovery plan for disasters that explains how to restore data in the case of a system failure, should be in order.

Data Retention Policy: Based on legal requirements, the system must also confirm to data retention policies, that define how long different data types should be stored on file and when they should be backed up or deleted.

Scalability: The database should be created in a manner that it can accommodate a lot of data and be scalable as the need arises, such as using indexes, partitions, and other measures to enhance the efficiency of the database.

Hardware and Environmental Requirements

Server Requirements: The system should be hosted on servers which meet the stated hardware conditions that includes CPU speed, memory and disk space for the efficient operation of the system.

Environmental Conditions: The servers and hardware should be used in as such environment where there is a guaranteed and stable supply of power together with adequate cooling and physical security.

Third-Party Integrations

Payment Gateway Integration: If required, the system should also have the capability of connecting with other third-party payment gateways for handling the order transactions and security features for managing the financial information.

Cloud Integration: If the system utilizes cloud infrastructure (e.g., AWA, Azure), proper cloud configurations, data synchronization, and access controls must be established.

4. Group Tasks

4.1 Environmental Module Specification

Data Flow Diagram

A data Flow Diagram (DFD) is a method for showing the flow of data through a system and how transactions take place between systems. It identifies the sources and flow of data within the system. A DFD, often known as 'Bubble chart', is a technique that can help in understanding the operation of a system. (Lindemulder, 2024)

4.1.1 Level 0 DFD

The DFD illustrates the Inventory Management System communicating with warehouse staff, suppliers, Customers and Admin as well as the exchange of data between the systems to support the overall process. It provides as a way of representing processes as updating inventory levels, order processing, customer interaction and administrative tasks. The Following is the Context level DFD of an Inventory Management System with four different entities:

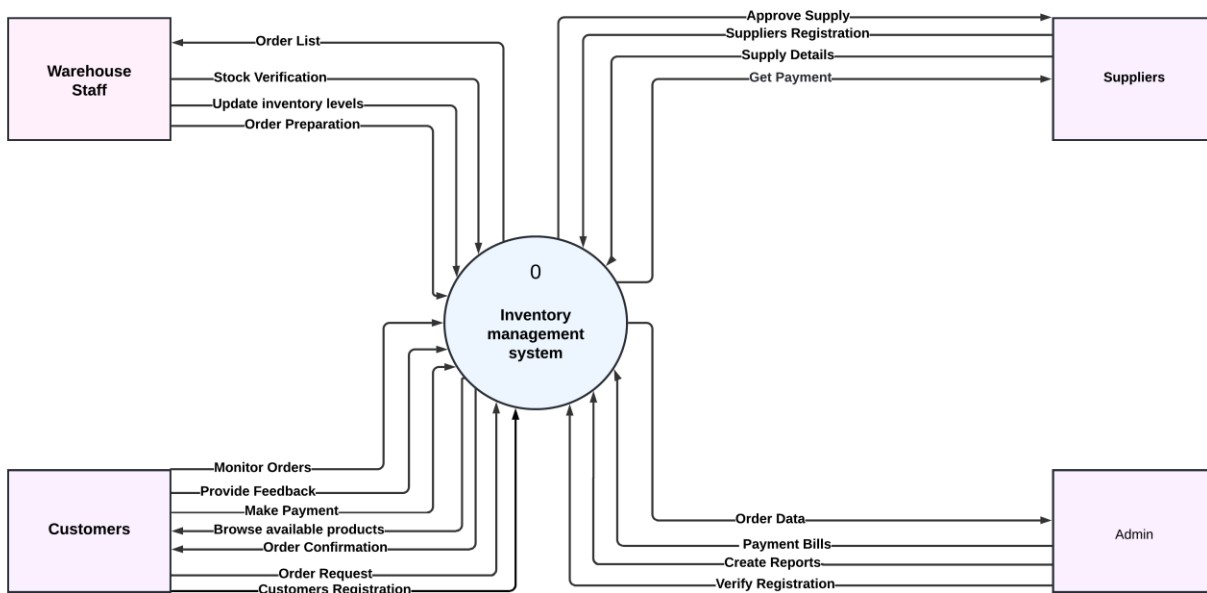


Figure 1 Context Level DFD (Overall)

Warehouse Staff: It includes employees in the warehouse are responsible of processing orders, verifying stocks, and updating inventory.

Customers: It includes customers who register, browse products, place orders, pay, and provide feedback.

Suppliers: it involves registering, providing supply information and receiving payment

Admin: Admin handles order information, verifies the information provided by the customers, handles payment and generate the reports.

Customers relations, ordering, delivery, and inventory control are all supported by the system.

4.1.2 Level 1 DFD

This data flow diagram also shows how an inventory management system works, with data flow between suppliers, customers, employees, and admin. Suppliers can register an account, deliver goods, and get payment, customers can create an account, search for products, order products, and make payment. While the admin handles data management, payment, and reporting, the staff manages packing, order dispatch, and updates the Inventory.

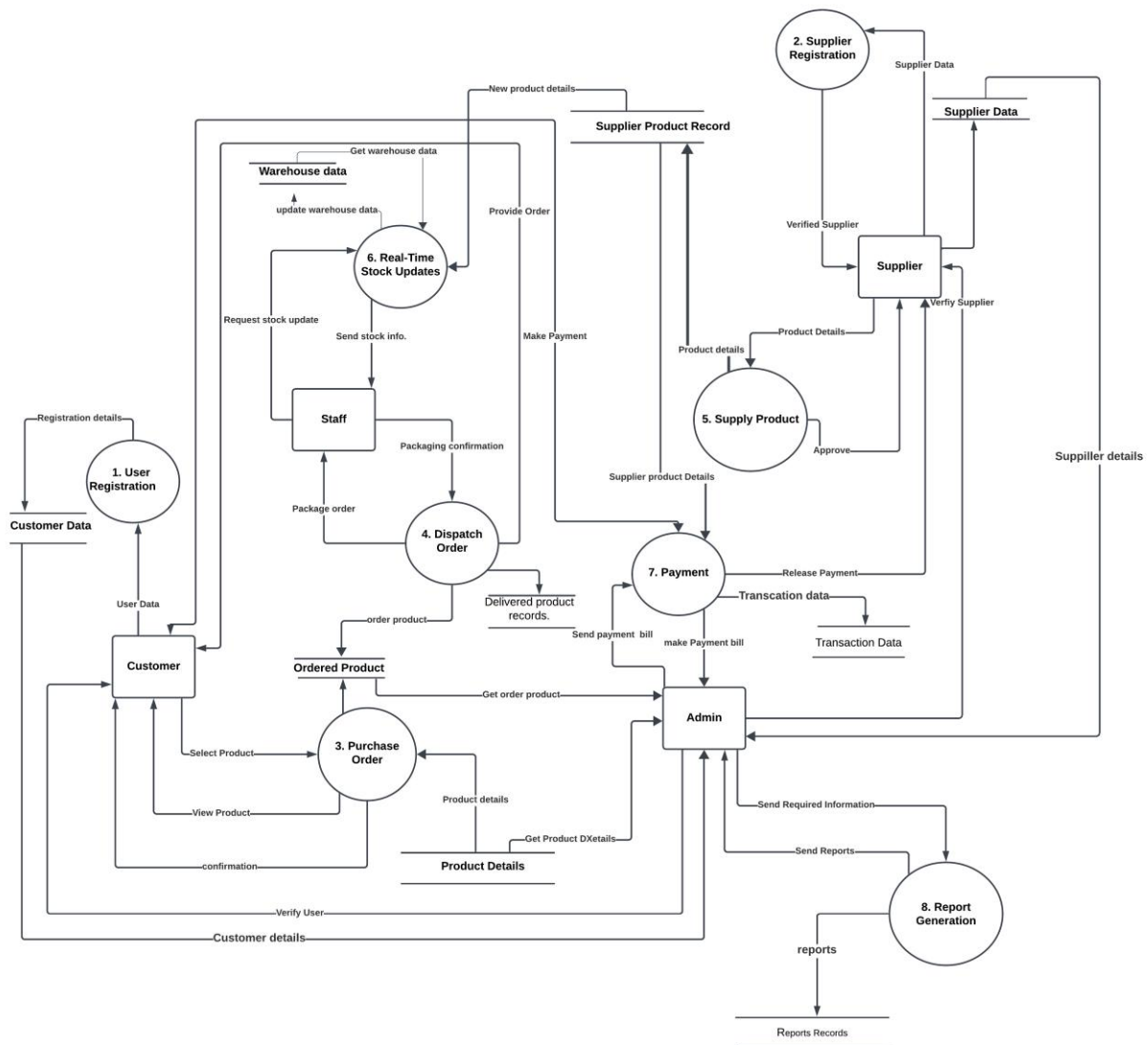


Figure 2 Level 1 DFD (Overall)

User Registration: customer must sign in to place an order.

Supplier Registration: Before supplying the products, suppliers are also ensured that they are verified.

Purchase Order: customers select and requests for the products they need.

Dispatch Order: Employees are involved in completing and delivering of customer requests.

Supply Products: the approved products are received from the suppliers.

Real-time stock Updates: the warehouse inventory is also practised in real-time

Payment: customers make the payment, and suppliers are provided with the payment.

Report Generation: depending on the transactions and products involved, the admin generates the report.

4.1.3 Level 2 DFD

Here the step shows the data flow for the module user registration. Firstly, user must enter the customer details and admin of the system will verify the customer details. After verifying the customer details, admin will notify the verification of the customer details to the customer

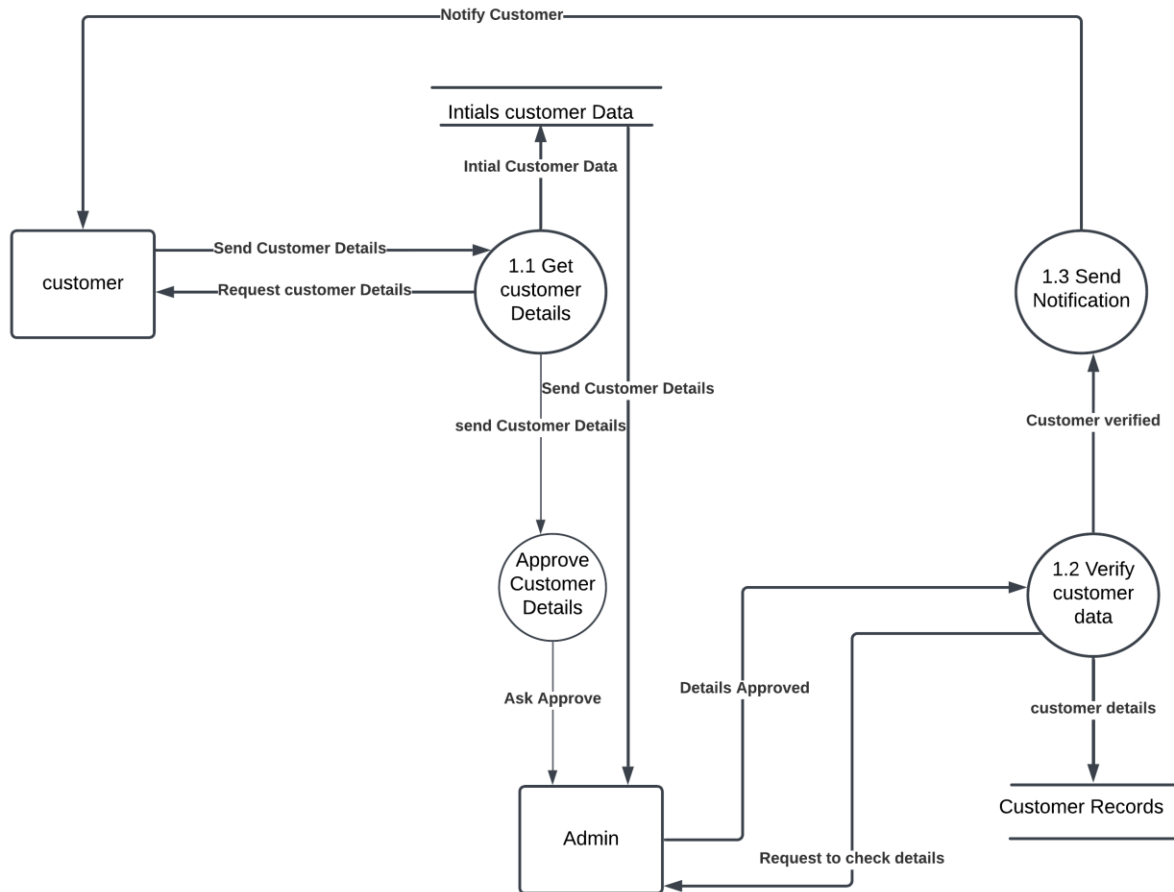


Figure 3 User Registration (Overall DFD)

Here the step how the data flow for the module purchase order. Firstly, the product available in the warehouse is displayed. Customers have a choice to select the product and can request the order. After requesting the order, order is confirmed by staff and billing process will be started.

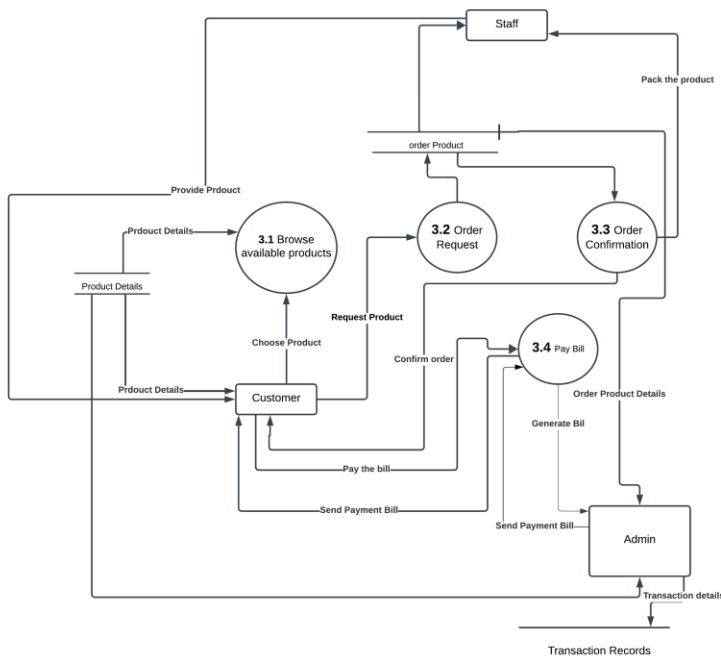


Figure 4 Report Generation (Overall DFD)

Here the step how the data flow for the module report generation. Firstly, the data is collected and then data is filtered as needed by the customer or a admin then a sample report is formed. After the analysis of the sample report the final report is prepared.

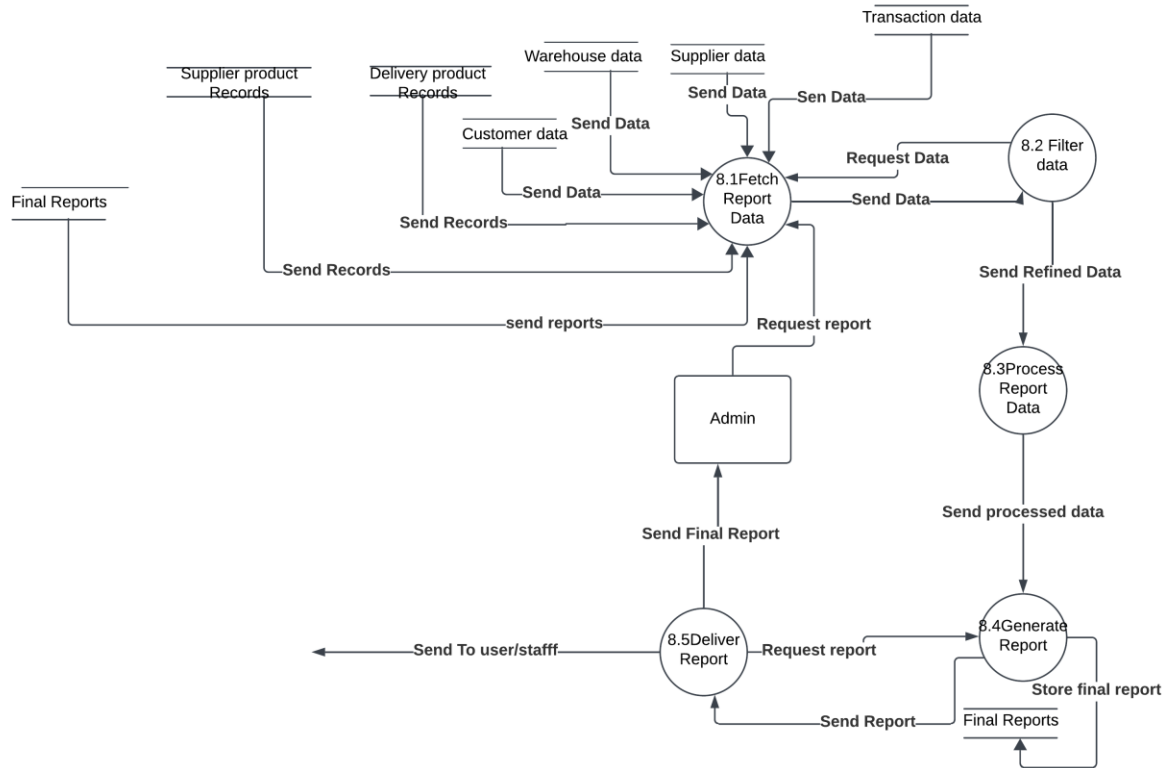


Figure 5 Purchase Order (Overall DFD)

4.2 Internal Module Specification

4.2.1 Entity-Relationship Diagram

An Entity-Relationship Diagram (ERD) is a type of diagram that is applied in a database design to show the relationship between various data types. It also facilitates in identifying some crucial characteristics of a system, the entities and their characteristics, therefore, improving the quality of the design process. ERDs makes the complex data relationships easier to understand so that the database is accurate, easily accessible and can be developed in a way that the organization requires. (Jacqueline Biscobing, 2024)

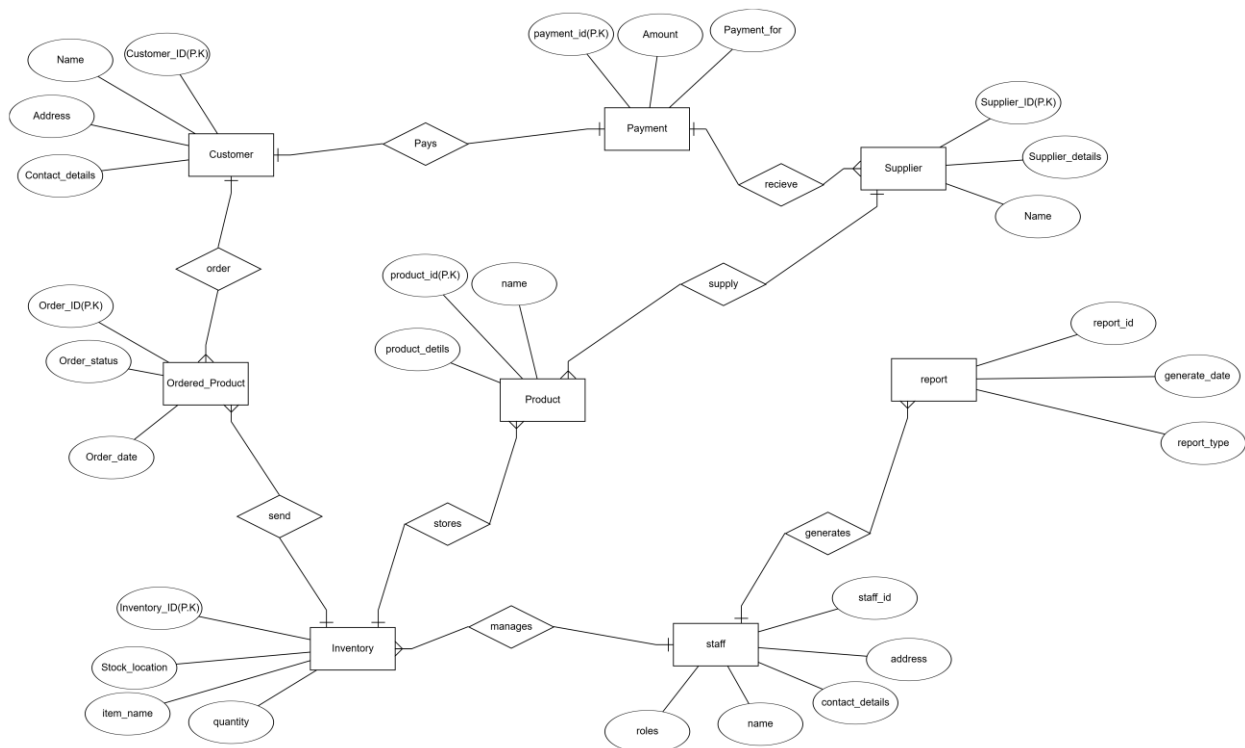


Figure 6 Entity-Relationship Diagram

4.2.2 Data Dictionary

A data dictionary is a structured repository of metadata that provides a comprehensive description of the data used. In early 1990s data dictionary are developed for managing the databases. Today, data dictionaries have transformed from single file catalogues into comprehensive metadata repositories, enabling advanced data analytics. It provides detailed descriptions of the data, including its structure, types, relationships and usages. The primary purpose of a data dictionary is to ensure consistency, standardization and clarity in how the data is defined, stored and used across the organization. The components of data dictionary include Data Element Name, Data Type, Domain Value, Description, Source, Date Created, Last Updated, Approve By, Owner, Relationships and Validation Rules. (Chia, 2024). Some of the benefits of using data dictionary include improved data quality, easier maintenance, easily searchable, better communication.

The data dictionary for the Inventory Management System:

- **Customer**

Customer = Customer_details + Customer_Data

Customer_Data = { Customer_Details }*

Customer_Details = Customer_ID + Name + Address + Contact_details

Name = Customer_First_Name + Customer_Last_Name + (Customer_Middle_Name)

Customer_ID = Number

Customer_First_Name = Varchar2

Customer_Middle_Name = Varchar2

Customer_Last_Name = Varchar2

Address = Varchar2

Contact_details = Varchar2

Input Parameters: Customer_ID, Name, Address, Contact_details

Output Parameters: Customer details such as ID, Name, Address, and Contact information

Called By: Order Module, Payment Module

Calls: Main

- **Supplier**

Supplier = supplier_details + Supplier_data

Supplier_data= {supplier_details}*

Supplier_details = supplier_ID + Name + Supplier_details

Name = Supplier_First_Name + Supplier_Last_Name + (Supplier_Middle_Name)

Supplier_details = Address + Contact_Number + (Email)

Supplier_ID = Number

Supplier_First_Name = Varchar2

Supplier_Middle_Name = Varchar2

Supplier_Last_Name = Varchar2

Address = varchar2

Contact_Number = Varchar2

Email = Varchar2

Input Parameters: Supplier_ID, Name, Address, Contact_Number, Email

Output Parameters: Supplier details such as ID, Name, Contact information

Called By: Inventory Module, Purchase Module

Calls: Main

- **Staff**

Staff= Staff_details + Staff_Records

staff_Data = {staff_Details }*

staff_Details = staff_ID + Name + Address + Contact_details + Roles

Name = staff_First_Name + staff_Last_Name + (staff_Middle_Name)

Roles = ["Inventory Manager", "Warehouse Manager", "Packing Clerk",]

staff_ID = Number

staff_First_Name = Varchar2

staff_Middle_Name = Varchar2

staff_Last_Name = Varchar2

Address = Varchar2

Contact_details = Varchar2

Roles = Varchar2

Input Parameters: Staff_ID, Name, Address, Contact_details, Roles

Output Parameters: Staff details like roles and contact information

Called By: Report Module, Dispatch Module, Real Time Stock Update Module

Calls: Main

- **Ordered_Product**

ordered_product = order_details + order_product

order_product = {order_details}*

order_details = Order_ID + Order_status + Order_date

Order_status = ["Processing", "Packing", "On a way", "Delivered"]

Order_ID = Number

Order_status = varchar2

Order_date = Date

Input Parameters: Order_ID, Order_status, Order_date

Output Parameters: Ordered product details including status and dates

Called By: Customer Module, Order Product

Calls: Main

- **Product**

product = product_details + product_details_table

product_details_table = {product_details}*

product_details = product_id + name + product_details

product_id = Number

name = varchar2

product_details = varchar2

Input Parameters: Product_ID, Name, Product_details

Output Parameters: Product information such as names and descriptions

Called By: Order Module, Purchase Order Module

Calls: Main

- **Payment**

payment = payment_details + transaction_data

transaction_data = {payment_details}*

payment_details = payment_id + Amount + Payment_for

payment_id = Number

Amount = Number

Payment_for = varchar2

Input Parameters: Payment_ID, Amount, Payment_for

Output Parameters: Transaction details such as amount and payment ID

Called By: Customer Module, Admin

Calls: Main

- **Report**

report = report_details + reports_records

reports_records = {report_details}*

report_details = report_id + generate_date + report_type

Report_type = ["Performance Report", "Supply Report", "Transaction Report", "Overall Report"]

report_id = Number

generate_date = Date

report_type = varchar2

Input Parameters: Report_ID, Generate_date, Report_type

Output Parameters: Generated report with type and timestamp

Called By: Inventory Module, Payment Module, Admin

Calls: Main

- **Inventory**

inventory = inventory_details + warehouse_data

warehouse_data = {inventory_details}*

inventory_details = Inventory_ID + Stock_location + item_name + quantity

Inventory_ID = Number

Stock_location = varchar2

item_name = varchar2

quantity = Number

Input Parameters: Inventory_ID, Stock_location, Item_name, Quantity

Output Parameters: Inventory levels, stock location, and item details

Called By: Supplier Module, Order Module,

Calls: Main

4.2.3 Process Specification

A process Specification or a process document is a thorough description of a task that must be performed in an organization. It offers a systematic account of the activities to be completed, the guidelines to follow and expectations to be fulfilled. This therefore ensures that everyone is positioned to perform the task in the right manner. It enhances the consistency, performance and reliability of the work being done. The document provides a roadmap of the process to achieve certain goals and objectives. This is since it improves the quality of outcomes that are produced. (Guerrero, 2021)

Process Specification of Inventory Management System

1. **Process Name:** Get Customer Details

Process Number: 1.1

Inputs: Send Customer Details (Name, Contact, Address)

Outputs: Request Customer Details, Store Initials of Customers, Send Initials customer details to admin

Process Logic

- Customer sends its information to the system
- System checks the accuracy and makes sure all the information is provided
- Stores the information collected from the customer in a temporary data store
- Generates the initials of the customer from the data
- Sends the initial customer data to the admin for additional review

2. **Process Name:** Order Request

Process Number: 3.2

Inputs: Request Product

Outputs: Store ordered Product

Process Logic

- The product is selected and requested by the customer
- Selected Product is stored by the system as an order request
- Sends the order request to the admin for processing
- Provide notifications to all the staff for packaging.
- Order details are stored in the system.

3. Process Name: Order Confirmation**Process Number:** 3.3**Inputs:** Process the order request from data store**Outputs:** Pack the Product, Confirm Order**Process Logic**

- The admin verifies and approves the order requests.
- After approval, orders are updated in the system.
- , the staff put the product into packet for delivery
- Ordered product's status are updated.
- After the order is processed, the system informs the customer about the product

4. Process Name: Pay Bill**Process Number:** 4.4**Inputs:** Send Payment Bill**Outputs:** Generate Bill, Pay the Bill**Process Logic**

- Admin creates a bill for payment for the product that were ordered
- The bill produced is send to the customer.
- Customers receives the bill for the system and pays the bill
- The system updates the transaction details.
- Payment confirmation is made with the customer.

5. Process Name: Generate Report**Process Number:** 8.4**Inputs:** Request Report, Send Processed Data**Outputs:** Send Report, Store Final Report**Process Logic**

- After system receives a data processed, the admin requests report information to generate the report
- The processed data is arranged by the system according to the defined parameters

- The final report is then generated
- Stores report in the database for future use and reviews
- Admin sends the detailed bill to the customers.

4.3 Design Specification Structure Chart

Structure chart is the visual representation of structure or modules in hierarchical forms. it usually provides the detailed breakdown of the entire system into the lowest possible useable module and provide the detailed explanation of each process and sub-process. The key components of structure chart are Modules or process, sub-process, control flow, Data flow. conditional and iterative process are often used in structure charts. Structure charts are primarily used during the design phase of a project to define and document's the system architecture and its functional components. generally, there are two types of structure chart they are transform centred structure and transaction centred structure. (GeeksForGeeks, 2024)

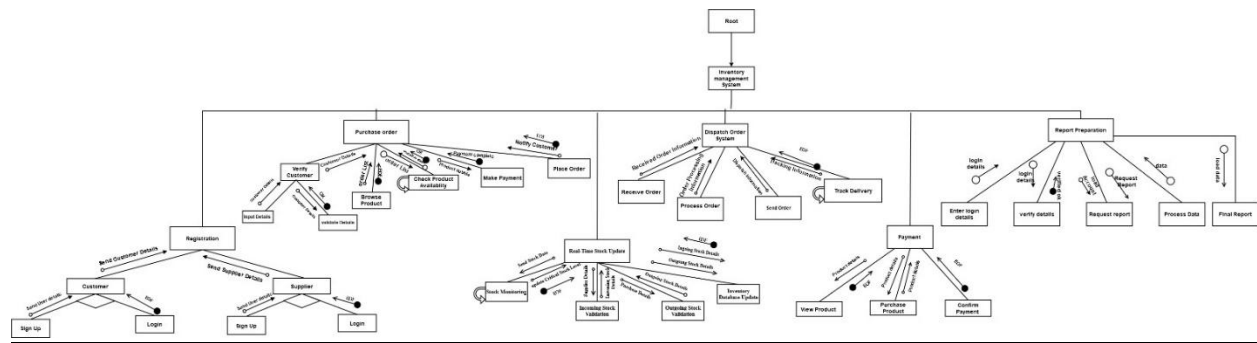


Figure 7 Structure Chart (Overall)

4.4 Progress Logs

Logbook Entry Sheet

Meeting No: 01

Date: 2024-12-09

Start Time: 9:00 pm

Finish Time: 10:00 pm

Items Discussed:

Project scope and agenda

Identification of budget constraints and risk

Achievements:

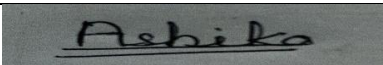


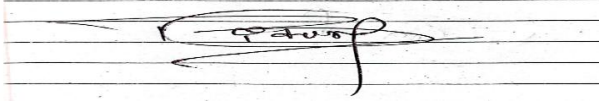

Actual project scope

Problems (if any):

No clear allocation of resources.

Tasks for Next Meeting:

Design of the system

Student's Name	Student's Signature
Ashika Nepal	
Dipin Karki	
Ishwor Dhakal	
Bidur Siwakoti	
David Basnet	

Logbook Entry Sheet

Meeting No: 02**Date: 2024-12-22**

Start Time: 9:00 pm

Finish Time: 11:00 pm

Items Discussed:

Design of the system

Data flow Diagram

Achievements:

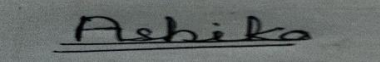


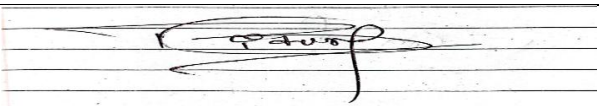

Approval of Data flow Diagram

Problems (if any):

No proper concept of Data flow Diagram

Tasks for Next Meeting:

Progress Check

Student's Name	Student's Signature
Ashika Nepal	
Dipin Karki	
Ishwor Dhakal	
Bidur Siwakoti	
David Basnet	

Logbook Entry Sheet

Meeting No: 03**Date: 2024-12-25**

Start Time: 8:00 pm

Finish Time: 10:00 pm

Items Discussed:

Progress logs

Data Dictionary and process specifications

Achievements:

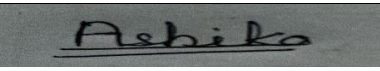
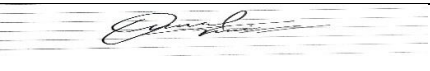

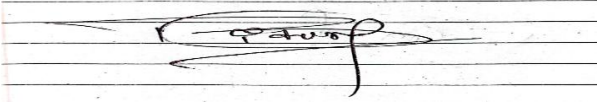
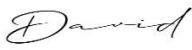
Actual project scope

Problems (if any):

No clear topic on Data dictionary

Tasks for Next Meeting:

Testing project

Student's Name	Student's Signature
Ashika Nepal	
Dipin Karki	
Ishwor Dhakal	
Bidur Siwakoti	
David Basnet	

Logbook Entry Sheet**Meeting No: 04****Date: 2024-12-29**

Start Time: 9:00 pm

Finish Time: 10:00 pm

Items Discussed:

Testing the project

Achievements:

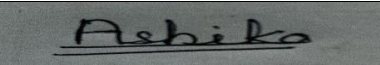


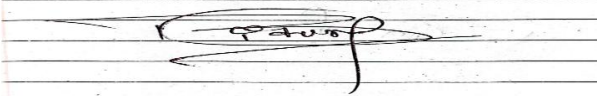

Project under proper scope

Problems (if any):

Final documentation

Tasks for Next Meeting:

Finalizing Project

Student's Name	Student's Signature
Ashika Nepal	
Dipin Karki	
Ishwor Dhakal	
Bidur Siwakoti	
David Basnet	

Logbook Entry Sheet

Meeting No: 05

Date: 2025-01-05

Start Time: 9:30 pm

Finish Time: 11:00 pm

Items Discussed:

Overall project review

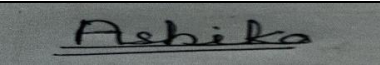


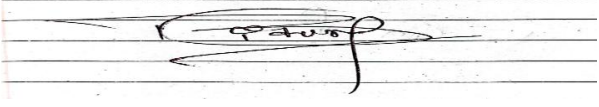

Finalization of report

Achievements:

Inventory Management System Project

Problems (if any):

No Problem

Student's Name	Student's Signature
Ashika Nepal	
Dipin Karki	
Ishwor Dhakal	
Bidur Siwakoti	
David Basnet	

4.5 Group Responsibilities

Member	Tasks
Bidur Siwakoti	Individual Task i. Purchase Order Group Task i. DFD ii. E-R Diagram, iii. Data Dictionary
Dipin Karki	Individual Task i. Payment Group Task i. SRS ii. Context Diagram, iii. Progress Logs
Ishwor Dhakal	Individual Task i. Report Preparation Group Task i. Business Case ii. Progress Logs iii. DFD
David Basnet	Individual Task i. Payment Group Task i. Structure Chart ii. DFD iii. E-R Diagram
Ashika Nepal	Individual Task i. Dispatch Order Group Task i. Business Case ii. SRS iii. DFD iv. Process Specification

5. Individual Task: Bidur Siwakoti

Feature: Purchase Order

The purchase order module is a critical and most important component of Inventory Management System designed for Global Tech Corporation. This module focuses on creating and managing purchase order. It allows customer to place order. The module also validates order information, check stock availability, calculates the total cost, and records the order in the database. It is an integral part of Inventory Management System as it connects customer demands with the inventory system, ensuring smooth order processing and tracking for both customer and administrators. By automating this process, it reduces errors, enhances customer satisfaction and optimizes resources utilizations.

5.1 Environmental Model Specification

5.1.1 Context Diagram/ Level 0 DFD.

The context diagram or level 0 DFD provides a high-level overview of the purchase order module, showing its interaction with external entities show as in picture. It focuses on the interaction between the primary entities like customer, Staff and Admin and the central purchase order Process.

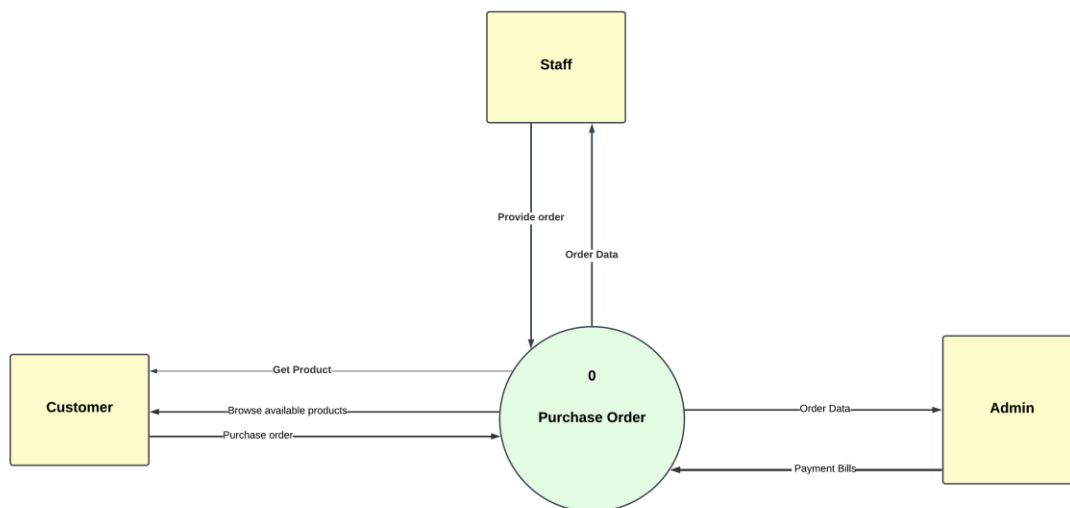


Figure 8: Context Level Diagram (Purchase Order).

5.2 Internal Model Specification

5.2.1 Level 1 DFD

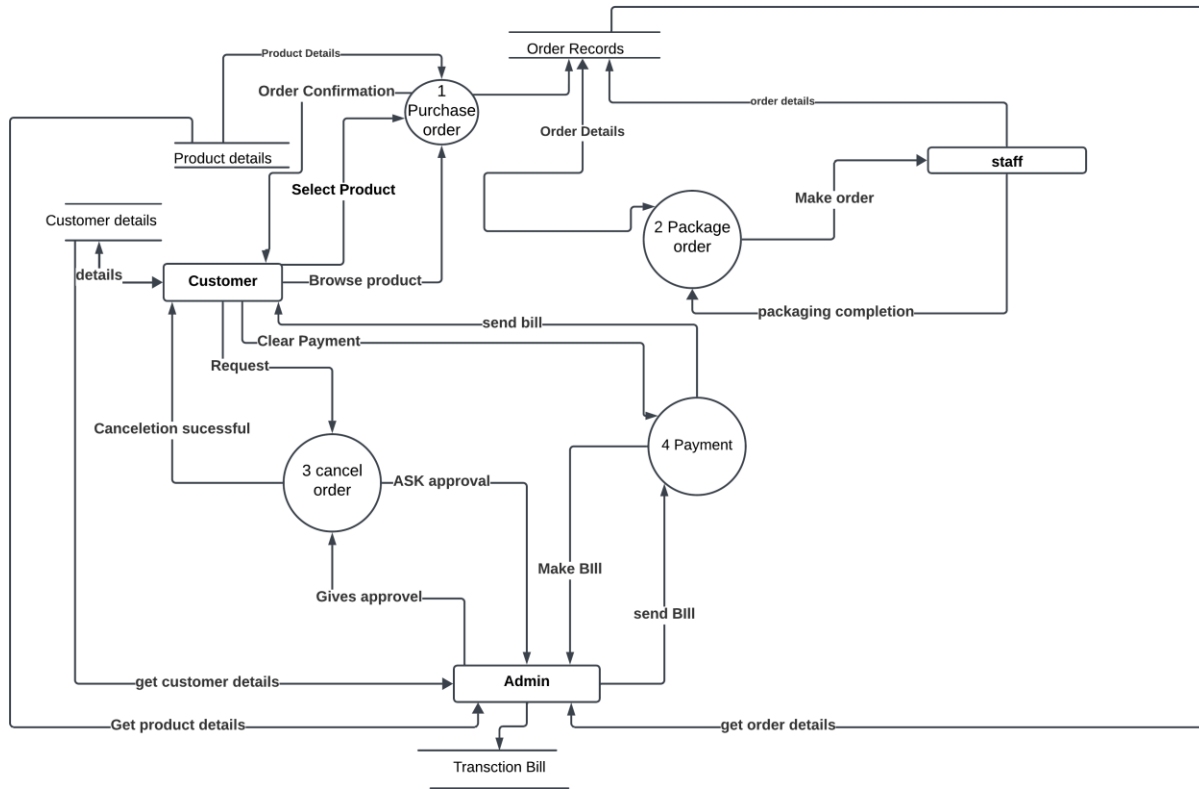


Figure 9: Level 1 DFD (Purchase Order).

The level 1 Data flow diagram provides a more detailed breakdown of the purchase order process by dividing it into four sub process: Purchase order, Package Order, Cancel Order and Payment. The process begins when a customer browses available product and select the desired items. Once the item is selected the system provides confirmation to the customer. After the confirmation, the order details are forwarded to staff. The staff packages the order and update the system with packaging completion status. After than customer clear their payment based on the bill sent by the admin. Once payment is received, the transaction is recorded, and a confirmation is sent to the customer. If customer request to cancel the order the system, ask approval to the admin. Once the admin approves the cancellation process, the customer is notified of the successful cancellation.

5.2.2 Level 2 DFD

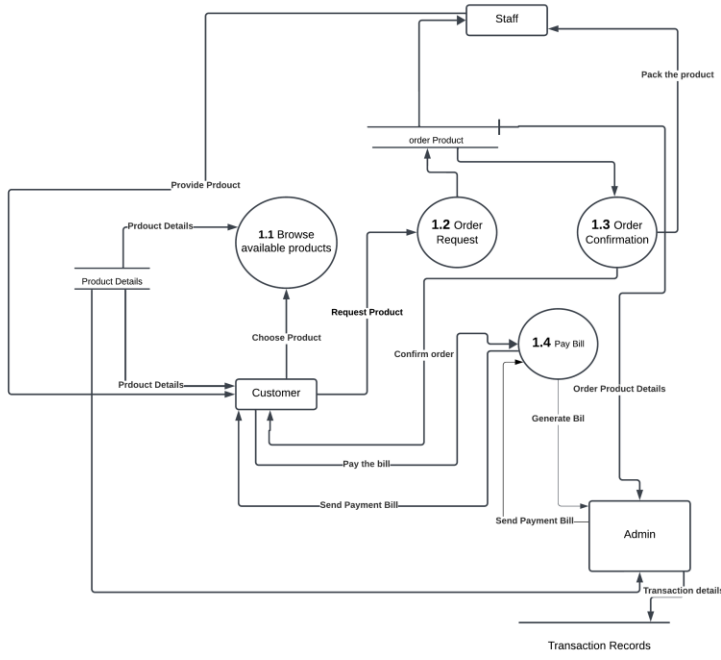


Figure 10: Level 2 DFD (Purchase Order).

The level 2 DFD breaks down the purchase order sub process further to give a more detailed view of how specific tasks are performed with the purchase order system. The process is further divided into four different sub processes: Browse Available Product, Order Request, Order Confirmation and Pay Bill. The customer interacts with the system to browse product and select the product, which initiates the next step. Once the customer selects the product the customer request is processed by creating an order record. Then the system forwards the order information to another process. The system confirms the order and provides the confirmation message to the customer. Then the admin generates a bill for the confirmed order. The system sends the bills to the customer, customer pays the bill, and the payment is recorded in the system. At last, the transaction records are updated and stored for future references.

5.3 Design Specification

5.3.1 Structure Charts

The structure chart for the **Purchase Order** module provides a hierarchical breakdown of its components. It visually represents the relationships between the main module and its sub-modules:

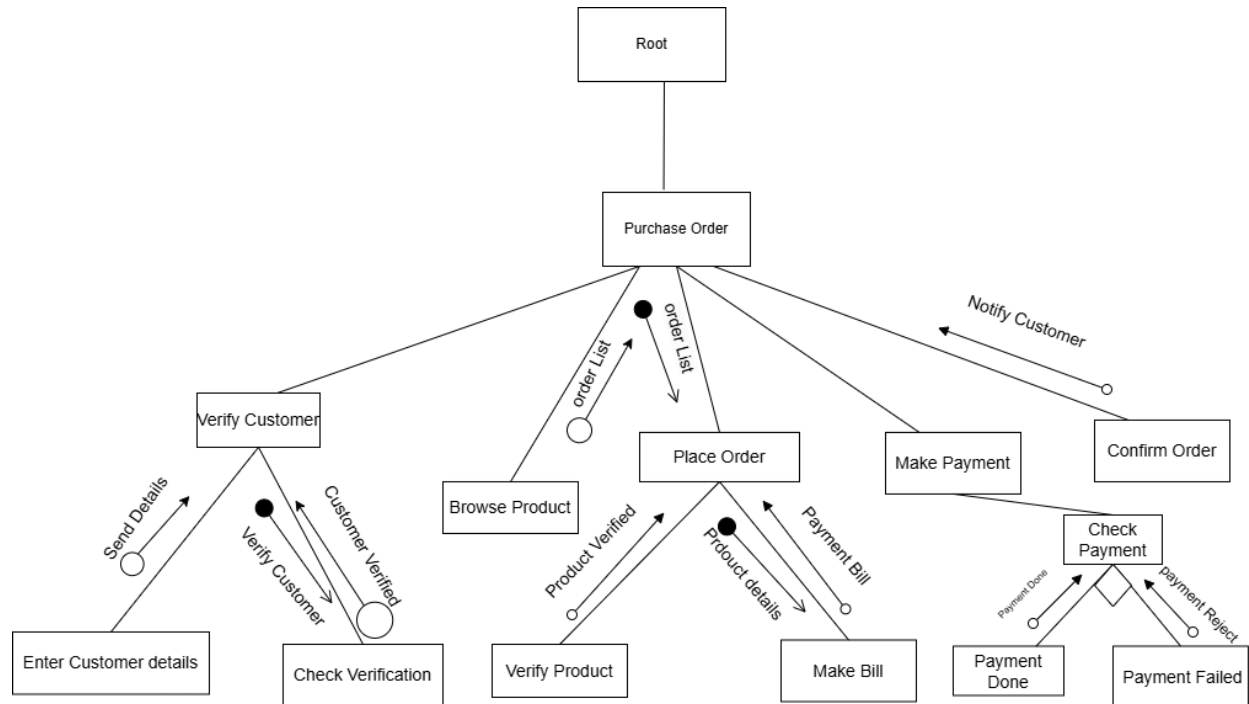


Figure 11: Structure Chart (Purchase Order).

5.4 Module Specification

A module specification is a detailed document that outlines the key information about an individual module within a system or project. It defines the module's purpose, expected outcomes, and technical components. (Geeks for Geeks, 2022).

Module Name: Purchase Order

Purpose: To handle the creation, validation and processing of purchase order. This module ensures that purchase orders are accurately recorded, and all required information is verified before processing.

Pseudocode:

DO

/* Verify customer */

VAR customerID = userDB.getCustomerDetails()

IF customerID is valid **THEN**

/* Get and validate order details */

VAR orderItems = getOrderItems()

DO

FOR EACH item **IN** orderItems

IF item is invalid **THEN**

DISPLAY "Invalid item detected. Please check you cart an remove invalid items"

EXIT DO

END IF

END FOR

END DO

/* Check item availability */

DO

FOR EACH item **IN** orderItems

VAR availableStock = PrdouctDB.checkItemAvailability(item.itemID)

IF availableStock < item.quantity **THEN**

```
        DISPLAY "selected items is out of stock or insufficient quantity."
    EXIT DO
END IF
END FOR
END DO
/* Calculate total amount */
VAR totalAmount = 0
DO
    FOR EACH item IN orderItems
        VAR itemTotal = item.quantity * item.price
        totalAmount = totalAmount + itemTotal
    END FOR
END DO
/* Ask for delivery location */
VAR deliveryLocation = INPUT ("Enter delivery Location")
IF deliveryLocation is invalid THEN
    DISPLAY "Invalid delivery location.."
    EXIT DO
END IF
/* Proceed for payment */
DISPLAY "Choose a payment method: 1. E-wallet 2. Online Banking 3. Cash on
Delivery (COD)"
VAR paymentOption = INPUT("Enter the payment Method")
IF paymentOption = "E-wallet" THEN
    VAR eWalletPaymentStatus = payment.processEwalletPayment(totalAmount)
    IF eWalletPaymentStatus = "Success" THEN
        DISPLAY "Payment Successful via E-wallet"
    ELSE
        DISPLAY "Payment failed. Please try again."
        EXIT DO
    END IF
```

```
ELIF paymentOption = "Online Banking" THEN
    VAR onlineBankingStatus = payment.ProcessOnlineBankingPayment(totalAmount)
    IF onlineBankingStatus = "Success" THEN
        DISPLAY "Payment Successful via Online Banking"
    ELSE
        DISPLAY "Payment failed. Please try again later."
    EXIT DO
END IF
ELIF paymentOption = "COD" THEN
    DISPLAY "You have selected COD and You need to pay before taking the product"
ELSE
    DISPLAY "Invalid payment option. Please try again."
    EXIT DO
END IF
/* Confirm the order */
VAR purchaseOrderID = getOrderID()
SAVE purchaseOrderID, customerID, orderItems, deliveryLocation, totalAmount,
paymentOption TO database
DISPLAY "Purchase Order Created Successfully!"
DISPLAY "Order ID: " + purchaseOrderID
ELSE
    DISPLAY "Invalid customer. Please register or log to purchase the product."
END IF
END DO
```

Input Parameters: CustomerID, OrderItems[], deliveryLocation, PaymentOption

Output Parameters: PurchaseOrderID, confirmationMessage, PaymentStatus

Global Variables: userDB, ProductDB, Payment

Local Variables: AvailableStock, itemTotal, totalAmount, PaymentOptions, PurchaseOrderID

Calls: getCustomerDetails(), getOrderItems(), checkItemAvailability(),
generateOrderID(), getDeliveryLocation(), SaveToDatabase()

Called_By: Main

6. Individual Task: Ishwor Dhakal

Feature: Report Preparation

6.1 Environmental Module Specification

Context level DFD

The level 0 DFD provides a high-level overview of the system with the focus of entities. It focuses on the central process (like Report Preparation in this DFD) and the flow between customers and admin.

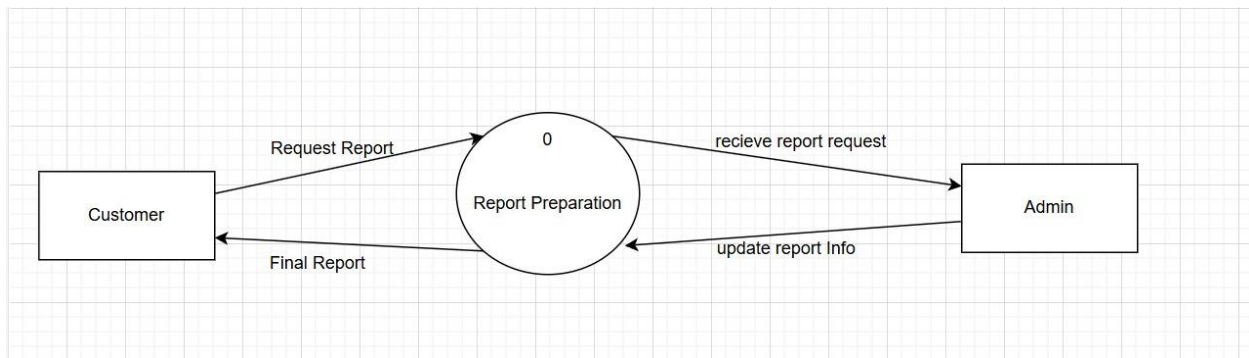


Figure 12 Context level DFD (Report Preparation)

6.2 Internal Module Specification

Level 1 DFD

The Level 1 DFD breaks the central process that is defined in Level 0 into subprocesses. It shows the subprocess like Request Report, Process Data and Final Report. At this level data stores are introduced as there are stores like Inventory Records, Supply Records, Order Records and Report with the data flows with each other's.

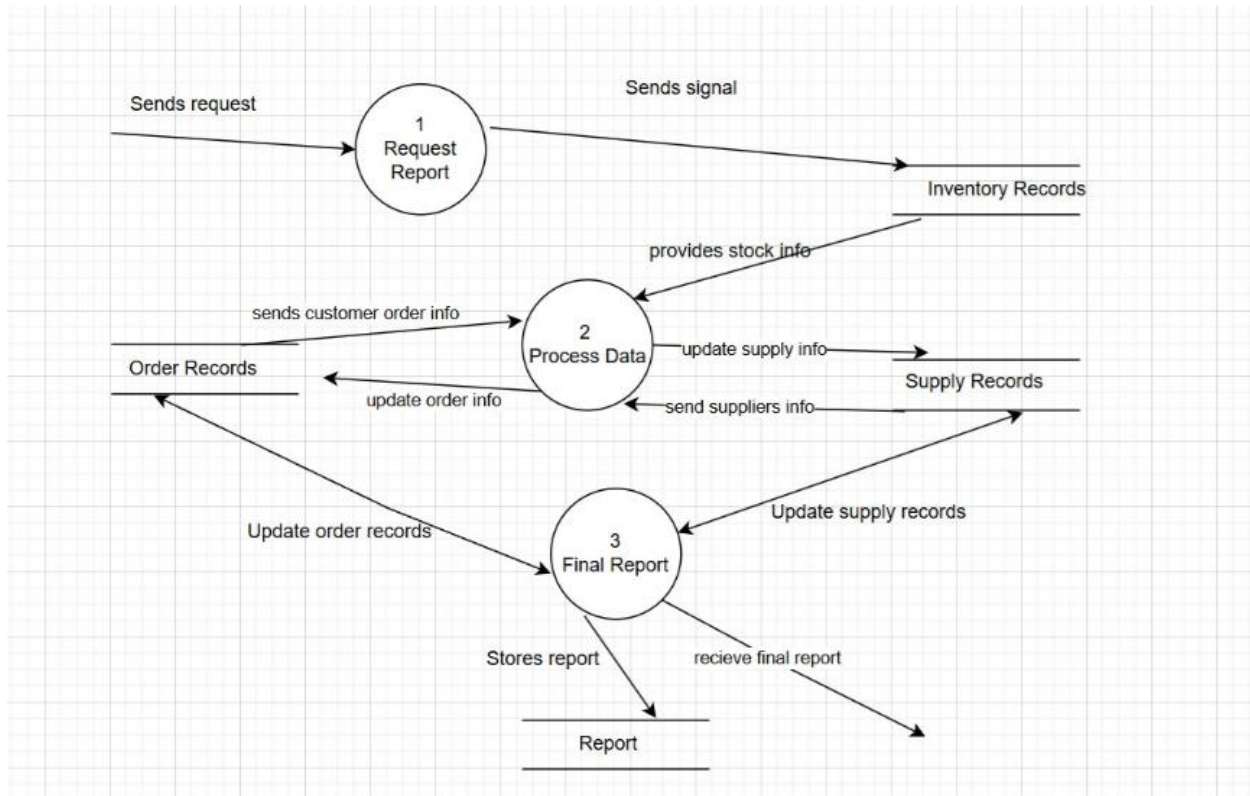


Figure 13 Level 1 DFD (Report Preparation)

Level 2 DFD

The level 2 DFD more break down the subprocesses of level 1 such as Request Report is divided into valid request, log Request and final valid request showing the data flow with the processes and data stores that stores the required data. This level shows the system in the detail way how the flow goes and how it is completed.

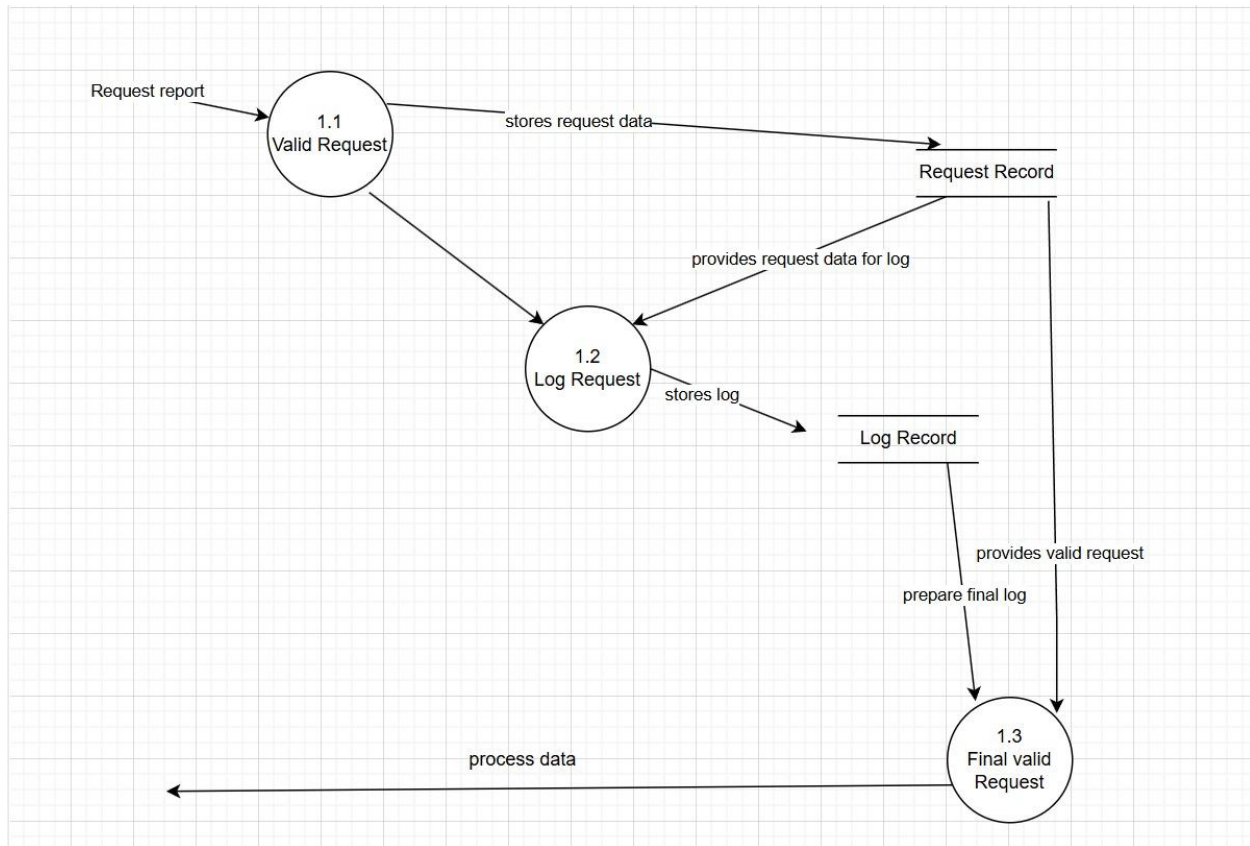


Figure 14 Level 2 DFD (Report Preparation)

6.3 Design Specification

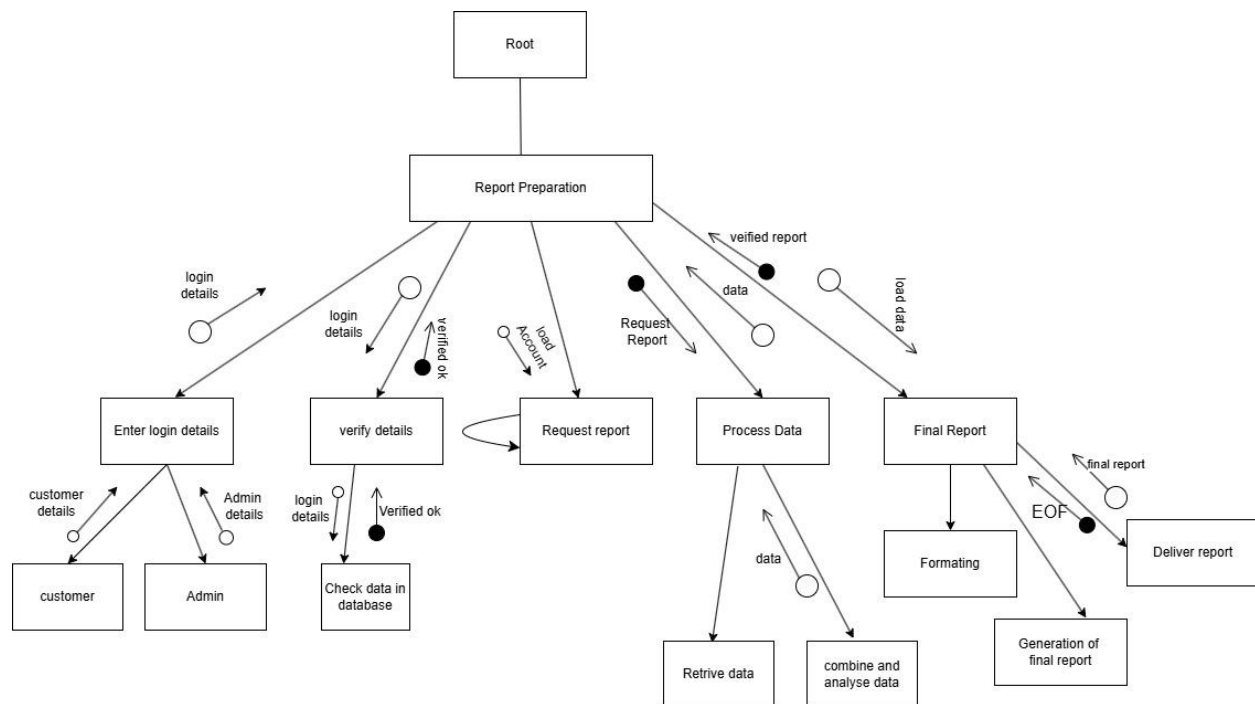


Figure 15 Structure Chart (Report Preparation)

6.3.1 Module Specification

Module Name: Report Preparation

Purpose: To make report generation faster, smoother and well-organized.

Pseudocode:

FUNCTION GenerateReport

/* Getting AdminId from Session */

VAR adminId = session.getAdminId()

IF data **IS NOT EMPTY THEN**

VAR reportType = **INPUT** ("Enter the report type: ")

/* Validate Report Type */

IF reportType == "Purchase Report" **THEN**

VAR startDate = **INPUT** ("Enter the start date")

VAR endDate = **INPUT** ("Enter the end date")

/* Fetch Purchase Data */

VAR purchaseData = Database.PurchaseReport.getPurchaseData(startDate, endDate)

/* Generate and Print Purchase Report */

VAR purchaseReport = ReportGenerator.generateReport(purchaseData)

PRINT (purchaseReport)

ELSE IF reportType == "Sales Report" **THEN**

```
VAR startDate = INPUT ("Enter the start date")

VAR endDate = INPUT ("Enter the end date")

/* Fetch Sales Data */

VAR salesData = Database.SalesReport.getSalesData(startDate, endDate)

/* Generate and Print Sales Report */

VAR salesReport = ReportGenerator.generateReport(salesData)

PRINT (salesReport)

ELSE IF reportType == "Customer Report" THEN

    VAR startDate = INPUT ("Enter the start date")

    VAR endDate = INPUT ("Enter the end date")

    /* Fetch Customer Data */

    VAR customerData = Database.CustomerReport.getCustomerData(startDate,
endDate)

    /* Generate and Print Customer Report */

    VAR customerReport = ReportGenerator.generateReport(customerData)

    PRINT (customerReport)

ELSE

    PRINT ("Invalid report type")

END IF

ELSE

    PRINT ("Data is empty")

END IF

END FUNCTION
```

Input Parameters: reportType, startDate, endDate

Output Parameters: purchaseReport, salesReport, customerReport, Invalid message

Global Variables: adminID

Local Variables: reportType, startDate, endDate, purchaseData, salesData, customerData, purchaseReport, salesReport, customerReport

Calls: session.getAdminID(), Database.PurchaseReport.getPurchaseData(), Database.SalesReport.getSalesData(), Database.CustomerReport.getCustomerData(), ReportGenerator.generateReport(), PRINT()

Called By: Admin

7. Individual Task: David Basnet

Feature: Real-time Stock Updates

The Real-time Stock Update is a module that specifies the stock handling processes. This module contains stock update, stock information and stock depreciation in warehouse. A data store is used to have information about stock. A supplier adds stocks which is further updated in the data store. A customer orders/purchase a stock after viewing the information about the stock in the data store and as the purchase is successful the stock from the data store updated. Use of the data store in this specific process is very essential as its secure and has less risk of data loss. Stock information can be updated via data store by admin.

7.1 Environmental Module Specification

Context Diagram is a process in which interaction between the entities are overviewed in high level method. As shown in the figure, Real-Time Stock Update is the process and the primary entities such as customer, staff and supplier have interaction between the process.

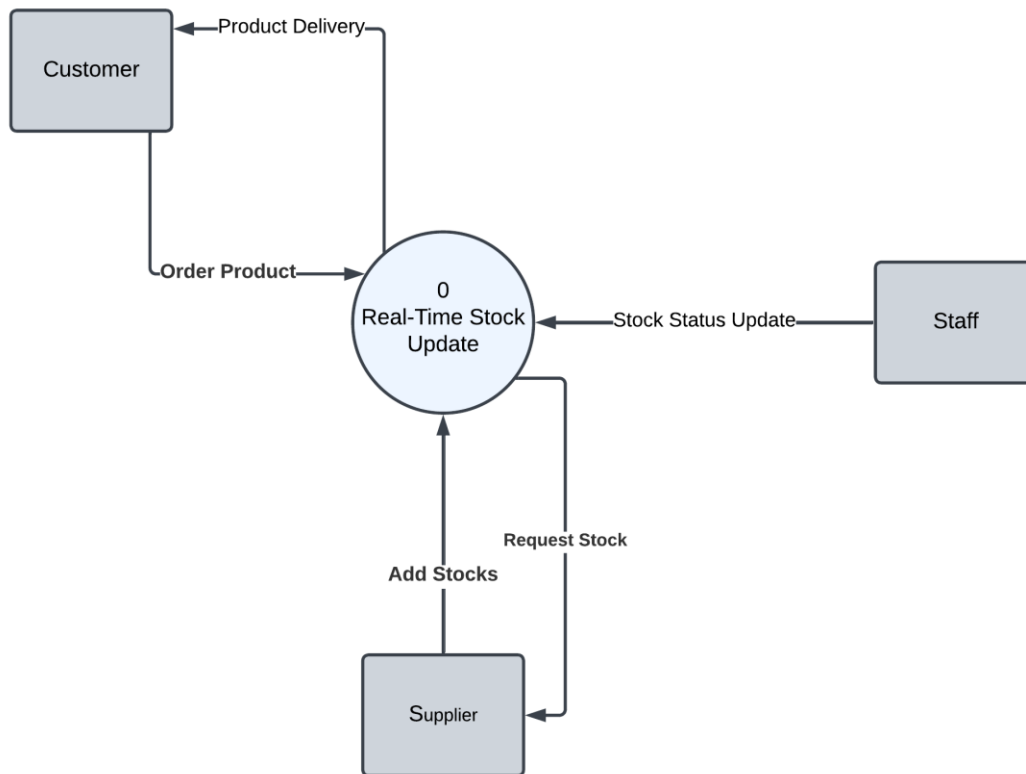


Figure 16 Context level DFD (Real-time Stock Updates)

7.2 Internal Module Specification

Level 1 DFD

The Level 1 DFD of Real-Time Stock Update is a detailed breakdown process of Context diagram/ Level 0 DFD. In this process we have 4 sub-processes: Order Management, Stock Request, Stock Checking and Stock Update. In the sub-process:

Order Management: The order placed by customer is updated in the stock data store which lead to depreciation of stock in the inventory.

Stock Request: This Sub-process is essential to request the ordered stock by the customer to the supplier.

Stock Checking: In this process the no of stocks in the stock data store is checked by staff and notified. This process contains all the details about the outgoing, incoming and removal of stock by customer and suppliers.

Stock Update: The deduction or addition of stock is verified by this process and is saved in the data store named “Stock Details”.

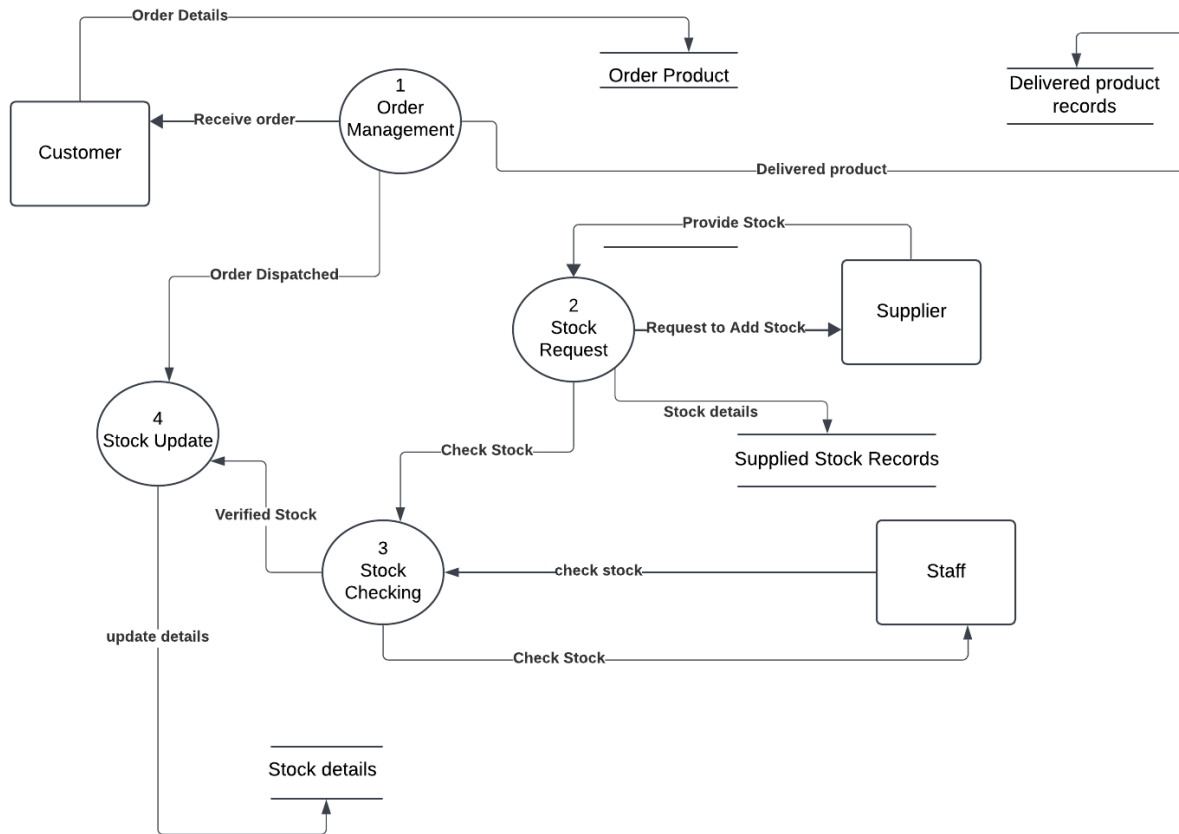


Figure 17 Level 1 DFD (Real-time Stock Updates)

Level 2 DFD

The level 2 DFD is a breakdown of the sub-process of level 1 DFD named “Stock Update”. This breakdown structure describes the stock update ways such as stock import or export. After the incoming or outgoing of the referred stock is done the stock details is updated to Real-Time detail. The stock update is then sent to a transaction record which holds the recent details of the incoming, outgoing and sold stock for future use. By this mean, data is secured and stock is updated time to time with the real world detail.



Figure 18 Level 2 DFD (Real-time Stock Updates)

7.3 Design Specification

The structure chart for the module “Real-Time Stock Update” is a breakdown method of its components making it easier to acquire the function of the module. The relation between the main and the sub modules are shown in the below breakdown structure called Structure Chart.

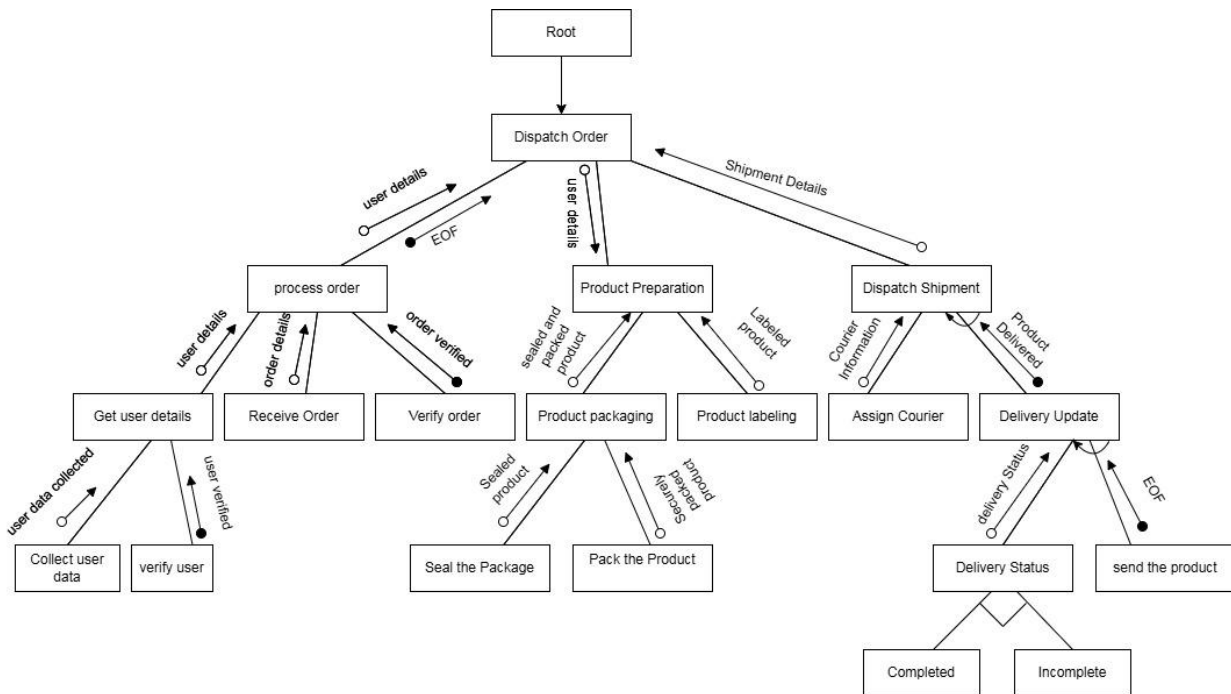


Figure 19 Structure Chart (Real-time Stock Updates)

7.4 Module Specification

A module specification contains information about an individual module. It details the aims and learning outcomes of the module, in addition to the teaching, learning and assessment methods. It includes a summary of content and a summary of methods and frequency of teaching.

Module Name: Real-Time Stock Updates

Purpose: To handle stock updates according to the orders and arrival of stocks. This module specifies the real-time update of stock in the data store when ordered or placed to be ordered by customers/suppliers.

Pseudocode:

DO

/* Authenticate the supplier and process stock supply */

VAR supplierID = getSupplierDetails()

IF supplierID is valid **THEN**

VAR newStockDetails = getSupplierStock()

DO

FOR EACH stockItem **IN** newStockDetails

/* Ensure stock data is valid */

IF stockItem.itemID is invalid **OR** stockItem.quantity <= 0 **THEN**

logManager.logError("Invalid stock provided by the supplier.", stockItem)

DISPLAY "Invalid stock provided by the supplier. Please check and try again."

EXIT DO

END IF

END FOR

END DO

/* Update the stock in the warehouse */

DO

FOR EACH stockItem **IN** newStockDetails

VAR currentStock = getWarehouseStock(stockItem.itemID)

VAR updatedStock = currentStock + stockItem.quantity

UPDATE warehouseStock **SET** stock = updatedStock **WHERE** itemID = stockItem.itemID

logManager.logActivity("Stock updated for item.", stockItem)

END FOR

END DO

/* Log the supplier's stock update for tracking */

SAVE supplierID, newStockDetails **TO** WarehouseRecord

DISPLAY "Warehouse stock updated successfully."

ELSE

DISPLAY "Supplier validation failed. Access denied."

logManager.logError("Supplier validation failed.", supplierID)

EXIT DO

END IF

```
/* Process customer orders */
```

```
VAR customerID = getCustomerDetails()
```

```
IF customerID is valid THEN
```

```
VAR orderDetails = getCustomerOrder()
```

```
DO
```

```
FOR EACH orderItem IN orderDetails
```

```
/* Check if there is enough stock for the order */
```

```
VAR availableStock = getWarehouseStock(orderItem.itemID)
```

```
IF availableStock < orderItem.quantity THEN
```

```
DISPLAY "We're sorry, but there's insufficient stock for: " + orderItem.itemID
```

```
logManager.logError("Insufficient stock for order item.", orderItem)
```

```
EXIT DO
```

```
END IF
```

```
END FOR
```

```
END DO
```

```
/* Deduct the ordered stock from the warehouse */
```

```
DO
```

```
FOR EACH orderItem IN orderDetails
```

```
VAR currentStock = getWarehouseStock(orderItem.itemID)
```

```
VAR updatedStock = currentStock - orderItem.quantity
```

```
UPDATE warehouseStock SET stock = updatedStock WHERE itemID = orderItem.itemID
```

```
DISPLAY "Processed item: " + orderItem.itemID
```

```
    logManager.logActivity("Order processed for item.", orderItem)
```

```
END FOR
```

```
END DO
```

```
/* Notify staff to package and prepare the order for dispatch */
```

```
VAR packagingStatus = notifyStaffForPackaging(customerID, orderDetails)
```

```
IF packagingStatus = "Success" THEN
```

```
    DISPLAY "Your order is being packaged and will be dispatched soon."
```

```
ELSE
```

```
    DISPLAY "There was an issue preparing your order for dispatch."
```

```
EXIT DO
```

```
END IF
```

```
/* Save the order details for future reference */
```

```
SAVE customerID, orderDetails TO TransactionLog
```

```
DISPLAY "Thank you for your purchase! Your order has been successfully processed."
```

```
ELSE
```

```
    DISPLAY "Customer validation failed. Please register or log in to continue."
```

```
    logManager.logError("Customer validation failed.", customerID)
```

```
EXIT DO
```

```
END IF
```

```
END DO
```

Input Parameters: supplierID, newStockDetails[], customerID, orderDetails[]

Output Parameters: stockUpdatedMessage, orderProcessedMessage

Global Variables: logManager

Local Variables: currentStock, updatedStock, orderDetails, packagingStatus

Calls: getSupplierDetails(), getSupplierStock(), getWarehouseStock(), updateWarehouseStock(),
getCustomerDetails(), getCustomerOrder(), notifyStaffForPackaging(), saveTransactionLog()

Called_By: Main

8. Individual Task: Ashika Nepal

Feature: Dispatch Order

8.1 Environmental Module Specification

Context level DFD

This Context level Data Flow Diagram (DFD) shows how a dispatch order system depicts with the following entities:

- Customers: order requests are placed by customer, receives order confirmation and the delivery status as well.
- Inventory System provides information on current inventory and the availability of an item as well as receives requests for order allocation and inventory inspections.
- Logistics: produces tracking and delivery status information after processing the dispatch details and delivery address.

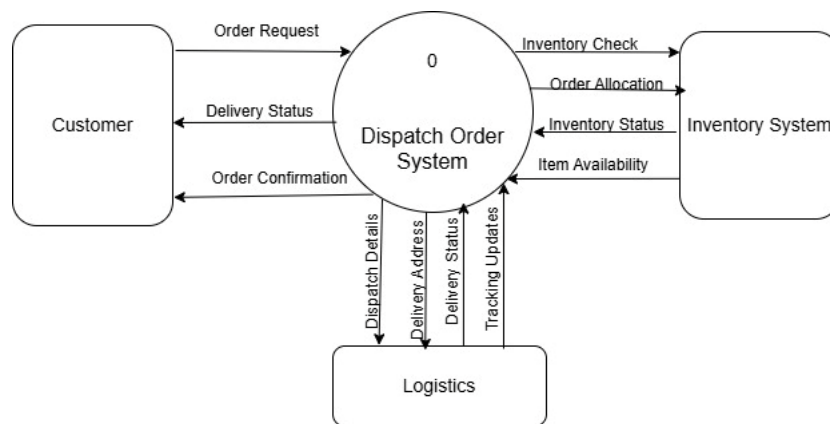


Figure 20 Context level DFD (Dispatch Order)

8.2 Internal Module Specification

Level 1 DFD

The level 1 Data Flow Diagram (DFD) for a dispatch order system shows how the system operates in the following way:

- Customer: orders, order confirmations, and delivery updates are received.
- Receive Order receives and verifies the receipt from customer
- Check Inventory: ensures that the items listed in the inventory system are in stock.
- Process Order: complete the order and generate shipment information by processing the order.
- Process Shipment: With the help of logistics delivery and information, products are transported to the respective destinations.
- Track Delivery: to verify the status of the inventory shipment, system provides information on the inventory.
- Logistics: organize the shipping information and track the shipment. It displays how the orders are received, processed, and inventory is checked before sending and the order tracking process is completed.

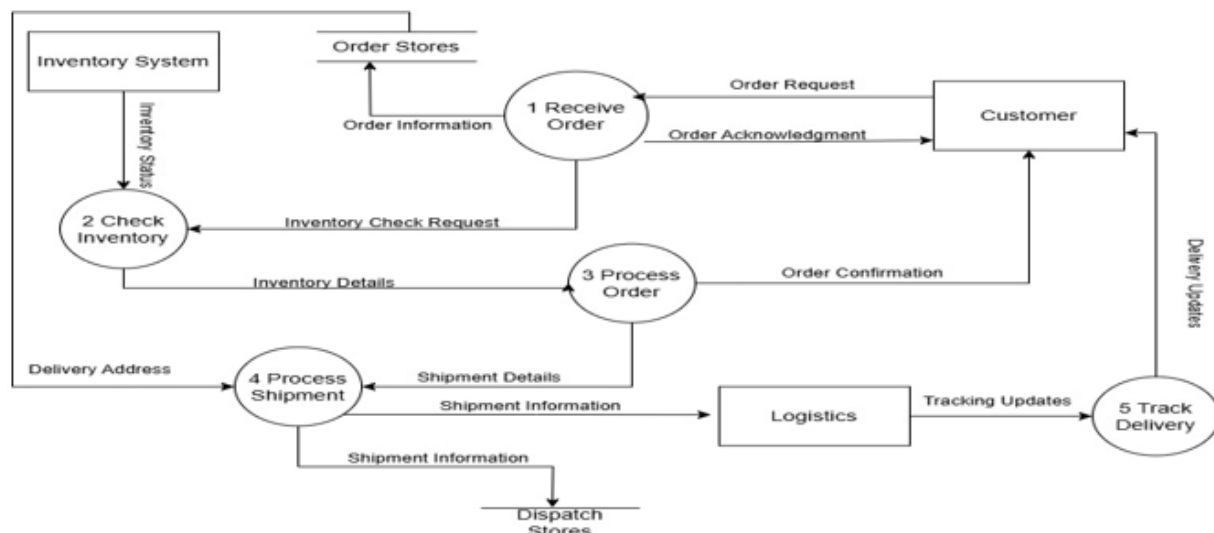


Figure 21 Level 1 DFD (Dispatch Order)

Level 2 DFD

The level 2 Data Flow Diagram (DFD) shows the details about process shipment, process.

- Prepare Shipment: shipping information is generated using the shipment information and the delivery address.
- Generates shipping label: here, the shipment details and the delivery address are stored to generate shipment information.
- Update Shipment Status: it also helps in monitoring on the system and shipment information.
- Update Shipment Status: It monitors the shipment details and updates the status with the help of logistics team.
- Dispatch Shipment: using shipping label and the updated status, system forwards the shipment.

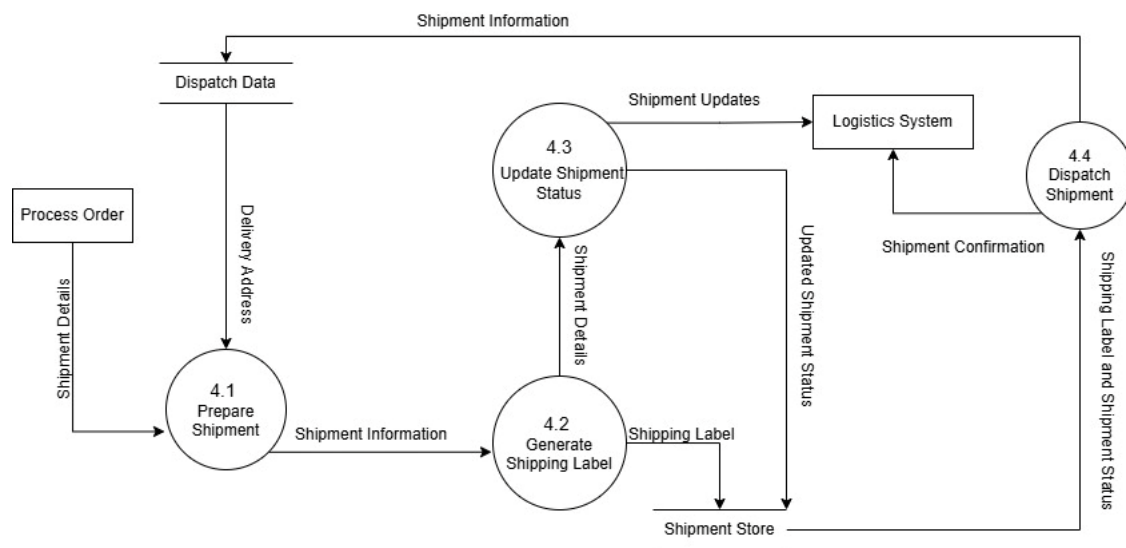


Figure 22 Level 2 DFD (Dispatch Order)

8.3 Design Specification

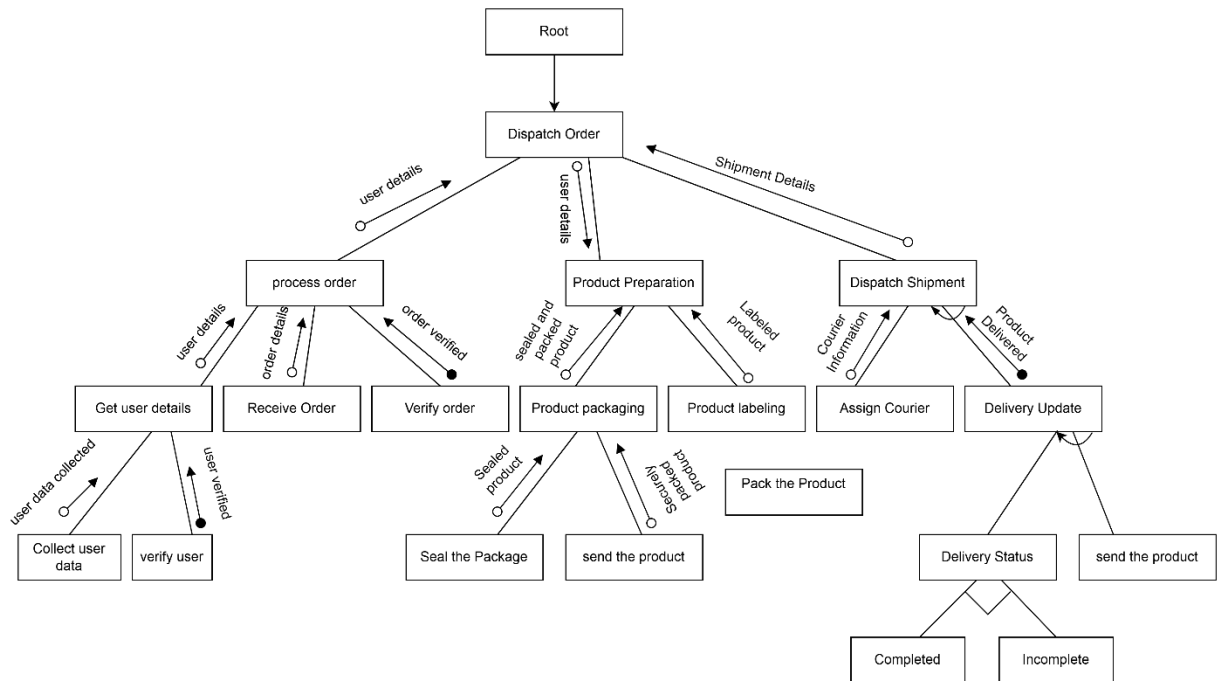


Figure 23 Structure Chart (Dispatch Order)

8.4 Module Specification

Module Name: Dispatch Order

Purpose: To manage and automate the process of receiving, processing, and tracking orders.

Pseudocode: Detailed steps for the main processes.

Pseudocode

START DispatchOrder

// Step 1: Receive Order

/* verifies whether the order information received is correct*/

FUNCTION ReceiveOrder (OrderInfo):

VALIDATE OrderInfo

IF OrderInfo IS VALID **THEN**

 /*sends the customer with a receipt*/

CALL SendAcknowledgment(CustomerID, "Order Received")

ELSE

PRINT ("Invalid Order")

RETURN

END IF

END FUNCTION

// Step 2: Process Order

FUNCTION ProcessOrder (OrderDetails):

 /* generates an order confirmation, provide it to the customer and create the shipment details for the order*/

VAR ShipmentDetails = **CALL** PrepareShipmentDetails (OrderDetails)

CALL GenerateOrderConfirmation (CustomerID, OrderDetails, ShipmentDetails)

CALL NotifyCustomer (CustomerID, "Order Confirmed")

END FUNCTION

// Step 3: Track Delivery

FUNCTION TrackDelivery(OrderID):

/* It retrieves the tracking information for the specific orderID and forward the same to customer as the latest tracking information*/

VAR TrackingUpdates = **CALL** ReceiveTrackingUpdates(OrderID)

CALL Notify Customer (CustomerID, TrackingUpdates)

END FUNCTION

// Main System Flow

/* receives the order and forwards it to ReceiveOrder to check on the validity of the order*/

INPUT OrderInfo

CALL ReceiveOrder (OrderInfo)

/* if OrderInfo is valid then the order is processed through the ProcessOrder function */

IF OrderInfo IS VALID **THEN**

CALL ProcessOrder (OrderInfo)

END IF

OrderID = OrderInfo.ID

/* the TrackDelivery function is used to track the order delivery.

CALL TrackDelivery(OrderID)

END DispatchOrder

Input variables: Order Information, Tracking Information.

Output variables: Order Confirmation, Tracking Updates, Notifications.

Global Variables: None specified.

Local Variables: OrderDetails, ShipmentDetails, TrackingUpdates.

Calls: SendAcknowledgement, PrepareShipmentDetails, GenerateOrderConfirmation, NotifyCustomer, ReceiveTrackingUpdates, ReceiveOrder, ProcessOrder, TrackDelivery

Called By: Main

9. Individual Task: Dipin Karki

Feature: Payment

In this system make payment function is very important. In this function all the required details for payment are included. The make payment function plays very important role to make accurate financial operation. Also, this function handles all the details of the payment related.

Environmental model specification:

9.1 Environmental Module Specification

Context level DFD



Figure 24 Context Level DFD (Payment)

In the above diagram the context level of the payment function is included. In that diagram there are 2 entities customer and management system which are engaged to make payment process. The diagram provides the process of how the payment work is done.

9.2 Internal Module Specification

Level 1 DFD

Level 1 DFD gives a more information about the system as comparing to the level 0. It breaks the level 0 information and gives the main process and shows how all the data are flow among them.

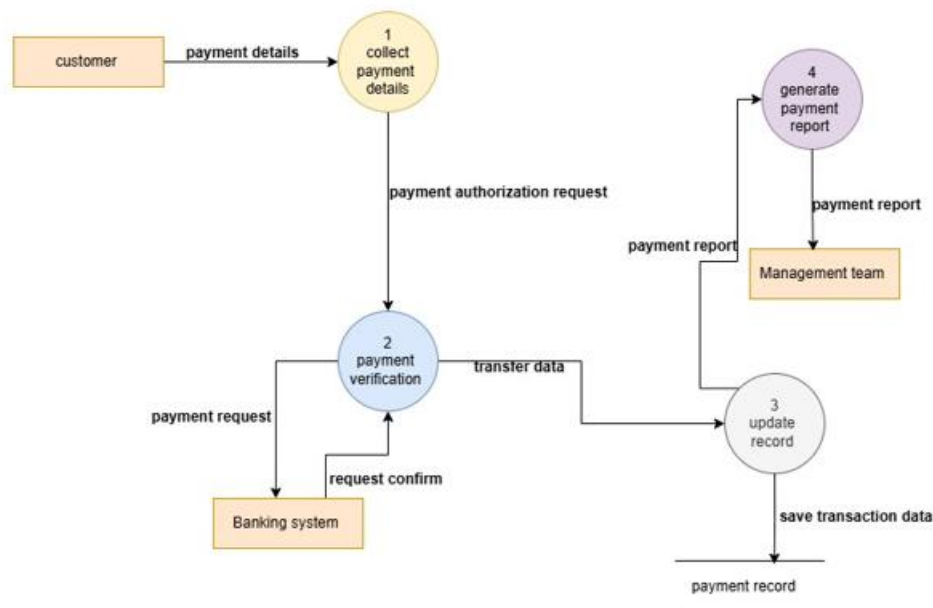


Figure 25: Level 1 DFD (Payment).

Collect payment details: In this process customer gives all the details like, amount, invoice reference etc.

Payment verification: In this process they collect all the details of payment and verify. After verifying all the details, they sent to the next further process.

Update record: After finishing all the transaction of the payment the bank sent all the payment details for update and record.

Generate payment report: After finishing all the process of payment the management team generate payment reports to handover the customer.

Level 2 DFD:

In the level 2 DFD it breaks down all the process from level 1 into many details process. For the payment system, all the details process of payment and proper operations are focused on level 2 DFD.

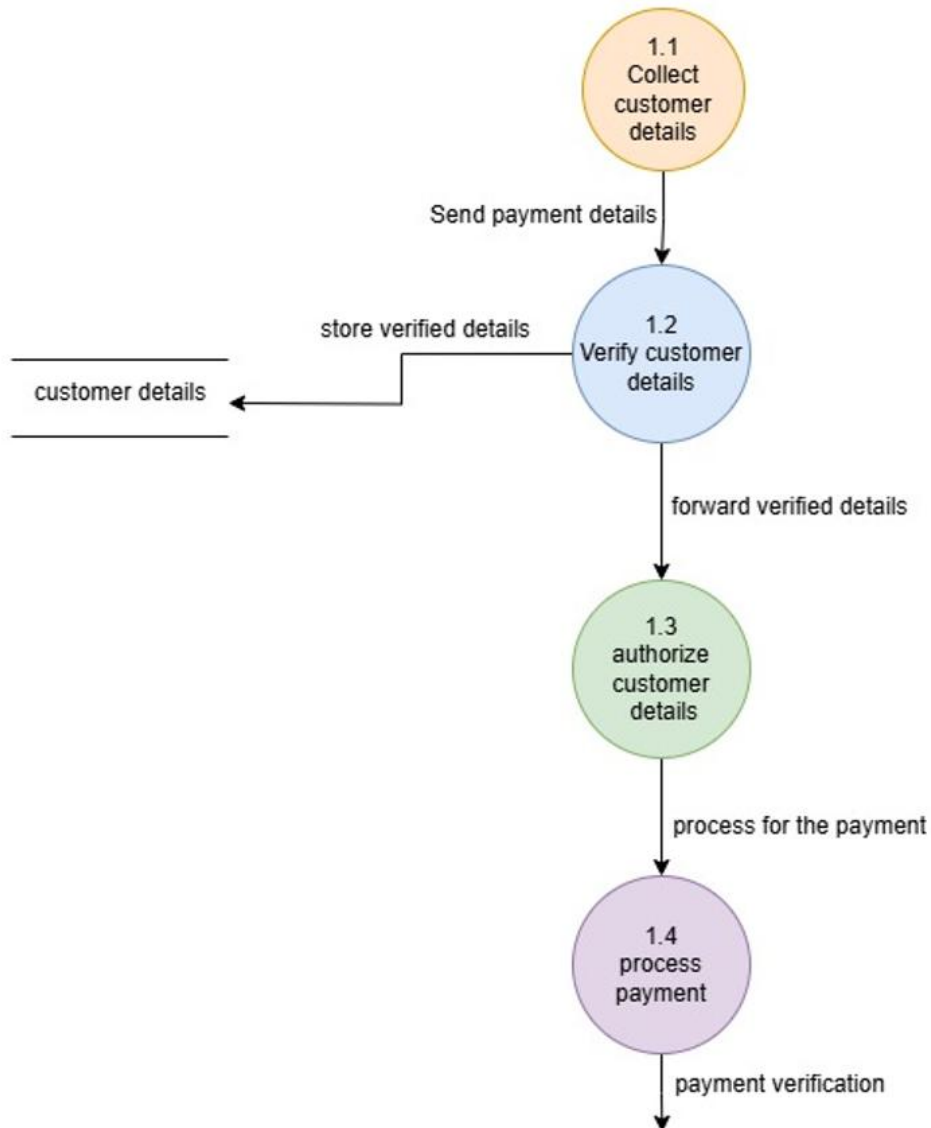


Figure 26: Level 2 DFD (Payment)

Collect payment details: In this process they gather all the information about collecting payment details. This process makes sure that all the payment process is done and accurate without any mistake and they step up for the verification process.

Verify payment details: After collect all the payment details the details is verified in this process then after it will be sent on further next process.

Process payment: In this process all the verified details are collected, and payment will be done and it will send for generating invoice.

Generate invoice: In this process after completing all the payment process the report or invoice will generate for prove and it will be sent to costumer.

9.3 Design Specification

Structure chart

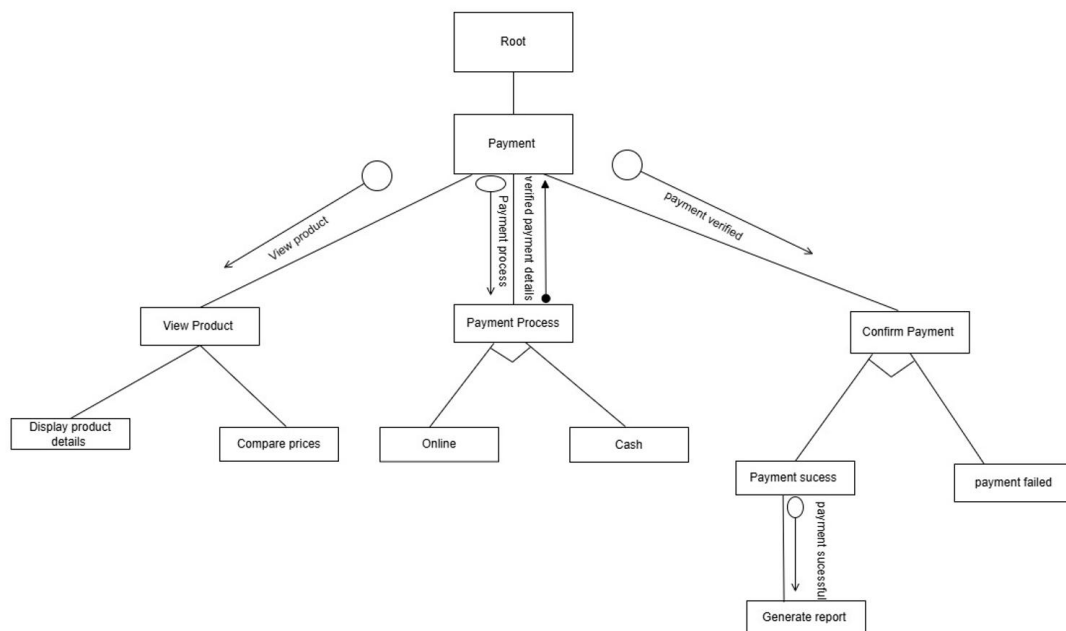


Figure 27 Structure Chart (Payment)

9.4 Module specification:

Module Name: Make payment.

Purpose: The main purpose of the make payment system in the Global Tech Corporation's new Inventory Management System (IMS) is to make payment process simple, easy and accurate.

Pseudocode:

Retrieving the user ID of the staff initiating the payment feature.

```
VAR userId = session.getUserId()
```

Requesting payment method and payment details from the staff.

```
VAR paymentMethod = Input("Select a payment method (Credit Card, Bank Transfer, etc.):")
```

```
VAR paymentDetails = Input("Enter payment details (e.g., account number):")
```

```
// Fetching the necessary amounts from inventory purchase and subscription records.
```

```
VAR inventoryCost = Database.inventoryPurchase.getCostDetails(userId)
```

```
VAR subscriptionFee = Database.systemSubscription.getSubscriptionFee(userId)
```

```
VAR totalPayment = inventoryCost + subscriptionFee
```

```
// Verifying the validity of the payment details.
```

```
VAR isDetailsVerified = Gateway.validatePaymentDetails(paymentDetails)
```

```
// Checking if the details are valid.
```

```
IF (isDetailsVerified == "True") THEN
```

```
    // Recording the payment in the database.
```

```
VAR paymentReference = Database.paymentHistory.recordTransaction(userId,  
paymentMethod, totalPayment)
```

```
// Checking the payment status.
```

```
VAR paymentStatus =  
Database.paymentHistory.getTransactionStatus(paymentReference)
```

```
IF (paymentStatus == "Success") THEN
```

```
    PRINT("Payment processed successfully.")
```

```
    // Generating and displaying the invoice.
```

```
    VAR invoiceDetails =  
Database.paymentHistory.generateInvoice(paymentReference)
```

```
    PRINT("Invoice generated successfully. Invoice ID: " + invoiceDetails)
```

```
ELSE
```

```
    PRINT("Payment failed. Please check the transaction details and try again.")
```

```
END IF
```

```
ELSE
```

```
    PRINT("Payment details verification failed. Please recheck the information provided.")
```

```
END IF
```

Input Parameters: paymentMethod, paymentDetails

Output Parameters: paymentReference, invoiceDetails, isDetailsVerified

Global Variables: Database

Local Variables: userId, inventoryCost, subscriptionFee, totalPayment

Calls: getUserId(), getCostDetails(), getSubscriptionFee(), validatePaymentDetails(),
recordTransaction(), getTransactionStatus(), generateInvoice()

Called By: Main

10. Conclusion

The development of the Inventory Management System (IMS) has significantly increased the problem-solving skill for the operational challenges faced by Global Tech Corporation. Focusing on warehouse management, the project aims to minimize the financial losses and increase customer satisfaction. Through different processes with proper planning, clear documentation and teamwork, the system was successfully designed and implemented.

The project focuses on the critical elements of warehouse management such as resource allocation, risk management and clear objective. The use of tools like data flow diagram, structured chart and data dictionary helps in designing the system functionally and non-functionally. Definition of detailed module specification provides the flexibility toward the process for development of the project.

Throughout the whole coursework process, practical knowledge was gained in system analysis, design and documentation. The importance of communication and progress check was mainly highlighted throughout the process. The challenges faced during testing and implementation were solved with proper communication and done within a timeline.

In conclusion, this project and coursework highlights the lifecycle of system development. Solving the problems faced in warehouse management with essential skills in project management provides positive gratitude towards the practical skills, which can be important in the future.

11. References

Barney, N., 2023. *What is SRS ?*. [Online]
Available at: <https://www.techtarget.com/searchsoftwarequality/definition/software-requirements-specification>

[Accessed 25 12 2024].

Chia, A., 2024. *What is data dictionary?*. [Online]
Available at: https://www.splunk.com/en_us/blog/learn/data-dictionary.html

[Accessed 18 12 2024].

Geeks for Geeks, 2022. *Module Specifications in Software Engineering*. [Online]
Available at: <https://www.geeksforgeeks.org/module-specifications-in-software-engineering/>

[Accessed 22 12 2024].

GeeksForGeeks, 2024. *Structure Charts – Software Engineering*. [Online]
Available at: <https://www.geeksforgeeks.org/software-engineering-structure-charts/>

[Accessed 26 12 2024].

Guerrero, K., 2021. *What is Process Documentation: 7 Key Benefits*. [Online]
Available at: https://www.bizagi.com/blogs/bpm/what-is-process-documentation-7?utm_source=chatgpt.com

[Accessed 25 12 2024].

Jacqueline Biscobing, K. T. H., 2024. *ERD*. [Online]
Available at: <https://www.techtarget.com/searchdatamanagement/definition/entity-relationship-diagram-ERD>

[Accessed 09 08 2025].

Lindemulder, G., 2024. *What is DFD?*. [Online]
Available at: <https://www.ibm.com/think/topics/data-flow-diagram>

[Accessed 05 01 2025].

