CS5002NI Software Engineering

Global Tech Corporation

20% Group Coursework

Final project

AY 2024-2025

Credit: 30

Group Name			
SN	**Student Name**	**College ID**	**University ID**
1	Salina Thing(L2C1)	np01cp4a230017	23047540
2	Kriti Lama(L2C1)	np01cp4a230013	23047513
3	Aastha Aryal(L2C1)	np01cp4a230400	23049023
4	Rahul Jaiswal(L2C1)	np01cp4s230131	22085575
5	Krishal Acharya(L2C1)	np01cp4a230381	23049010

Assignment Due Date: 9th January 2025

Assignment Submission Date: 9th January 2025

Table of Contents

Tables of Figures

Tables of Tables

1.Introduction

This is the group coursework of module “CS5002NI-Software Engineering” which is carried out equally by the contribution of each member of the group within a given timeline. Global Tech Corporation implements the need for a reliable and uncomplicated Inventory Management System (IMS) to manage the administrative shortcomings and inefficiencies within its storage facilities in Nepal. This project arises in order to tackle the setback of a prior project, which was sustained from improper project management, an uncertain scope, and a lack of proper documentation.

To address the administrative issues and problems that have come up in the storage facilities in Nepal, Global Tech Corporation has begun a project to develop an Inventory Management System (IMS). The company implemented an organized and structured strategy to guarantee the success of this new initiative after a previous project failed due to poor project management, unclear scope, and a lack of documentation.

The IMS, which will work as an automated system, will improve important procedures including tracking inventory, purchase order management, sales management, and report generating. Through emphasis on user-oriented design, the system will offer admins and consumers with personalized access, promoting efficient administration and decision-making.

Safe user access, efficient product and sales management, report generation and a safe payment system in addition to inventory activities are some of the IMS's primary features. The system was designed to increase customer satisfaction, decrease financial losses, and boost production.

The business case, system requirements, analysis, and design specifications for the IMS project are documented in this document, which ensures alignment with company goals. The IMS will serve as a flexible and reliable solution that will assist Global Tech Corporation's operational success goals.

The system was designed to increase customer satisfaction, decrease financial losses, and boost production. The business case, system requirements, analysis, and design specifications for the IMS project are outlined in this document, which ensures alignment with company goals and provides verifiable value. The IMS will serve as a flexible and dependable solution that will assist Global Tech Corporation's operational success goals.

The aim of this project is to develop a flexible and reliable Inventory Management System (IMS) for Global Tech Corporation.

The objectives of this project are as follows:

- To improve efficiency within the company
- To make reporting generation of both sales and purchases easier
- To enhance customer satisfaction
- To promote productivity within the company
- To promote effective decision making among admins and customers.

2. Business Case

Business case serves as a tool for evaluating progress and ensures that the project continues track to meet its goals within the time frame specified (Korostashovets, 2024). A business case and a charter can serve the same function, depending on the management structure and environment of the organization.

Business case	
Project Name	Implementation of a Next-Generation Inventory Management System (IMS)
Project Description	The Global Tech Corporation is planning to implement a new automated Inventory management system (IMS) to overcome the inefficiencies of warehouse operations. Last year, due to lack of clear objectives, poor resource allocation, inadequate communication and no risk mitigation plan leads to the previous system resulting in operational delays, financial losses, and customer dissatisfaction. The new IMS Includes structured project management, streamline operations, improve inventory accuracy as well financial state and enhance customer service. As this system focuses on structured project management systems which include proper planning, effective communication, tracking and reporting, enough resources and so on. An effective management system and proper operations (proper analysis and design) can lead to smoother execution and alignment with organizational goals.

Problem Statement	Global Tech Corporation lacks an effective system to manage large-scale distribution, leading to operational inefficiencies, delays, and customer dissatisfaction. Key challenges include: • **Operational inefficiencies:** Faulty data disrupts order processing and manufacturing. • **Challenges with flexibility:** The system cannot adapt to emerging technologies or handle larger scales. • **Downtime and risks:** Poor risk evaluation causes system interruptions and late payments.
Assumption	• Hardware and software availability within six months. • Access to skilled staff. • No major disruptions during implementation.
Benefit	1. **Improved accuracy:** Better inventory control boosts profitability. 2. **Customer satisfaction:** Maintains stock availability and loyalty. 3. **Increased efficiency:** Reduces time and labor in inventory handling. 4. **Cost savings:** Cuts storage and operational expenses. 5. **Enhanced visibility:** Tracks inventory across locations for informed decisions.
Risks and Mitigation	1.**Downtime:** Regular maintenance and backup plans 2. **Data loss:** Comprehensive data backups. 3. **Resource issues:** Clear planning and progress monitoring. 4. **Resistance to change:** Training and engaging stakeholders. 5. **Poor UX:** Testing and refining design based on employee feedback.
Measures of success	1. 30% reduction in order processing time. 2. 20% lower inventory costs. 3. 25% higher customer satisfaction.

	4. 90% employee adoption rate.
Timeline	The project will take one year, costing Rs. 10,00,000, covering staff, equipment, and software licenses. 1. Planning (0-2 months) -Rs. 2,50,000 2. Development and Testing (3-8 months)- Rs.2,50,000 3. Implementation and Training (9-10 months)-Rs. 2,50,000 4. Go-Live and Review (11-12 months)- Rs. 2,50,000
Team members	see below

S.N.	Name	Position
1	Salina Thing	Project Manager
2	Aastha Aryal	Software Developer
3	Kriti Lama	UI/UX Designer
4	Rahul Jaiswal	Quality Assurance/Tester
5	Krishal Acharya	System Analyst

Table 1: Table of Business Case

3. SRS

3.1. Functional Requirements

The following functions are placed in a system.

a) Access users to the system:

Before using any system's feature, the system should provide a registration function for a user(customer/admin). The following information should be provided for the registration:

- Personal information includes full name, email address, phone number, address, role selection (admin or customer) and additional requirements (if any).
- The system verifies the user information for accuracy.
- After that, the system successfully generates a unique account for users.
- The password must meet security criteria.
- The system will maintain every individual user's information in a recorded database so that users can update their information time and again and able to login using credentials.
- The system must provide two-factor authentication for recovery if users forget their password.
- After successful registration, the system must give access to various features within the system to the user. The system shall send a conformation notification to the user via email.
- If they already have an account of the system, then they can simply login by using credentials.
- Admin sees inventory and system management options.
- Customers see product browsing and purchase-related features.

b) Purchase order

Purchase orders allow users to create and manage orders with suppliers, ensuring products are ordered in the correct quantities and at the right price.

- The system allows the users to keep a record of details of new purchases of products like product details, quantities, price and payment status made by customers.
- The system lets users review past purchase details like product names, quantities, purchase dates and so on.
- The system automatically updates the added purchased quantities to the inventory.
- The system may display things such as the total amount spent with a supplier, number of products purchased and outstanding orders or payments.

c) Generate Report

- The system will have a feature specifically for the admin to access all information about users, employees, and the institution.
- The admin must retrieve data from the respective databases for users, employees, and the institution.
- The system must provide detailed financial information, including income, expenditure, and other expenses of the institution.
- The admin, after collecting all necessary data, must be able to generate detailed reports.
- The system must track new inventory acquisitions, detailing quantities, costs, and suppliers.
- The system must provide information on total sales, quantities sold, and profit or loss across different time periods.
- The system must evaluate the overall performance of employees, including productivity and task efficiency.
- The report generated by the admin must meet audit requirements.

d) Sales management

- The system must allow the admin to view all sales orders, including order ID, product details, quantities, and order status.
- The system must enable selection of delivery methods for each sales order, such as standard shipping or in-store pickup.
- The system must display the delivery address provided by the customer, including information such as address, city, state, and postal code.
- The system must provide contact information about the customer, including name, phone number, and email address, for order confirmation and updates about the product.
- The system must include a module for managing dispatched details, such as:
    - Assigned courier or delivery service.
    - Expected delivery date and time.
- The system must allow sales personnel to update the status of sales orders, such as "Processing," "Dispatched," "Delivered," or "Canceled."
- The system must generate a summary of sales data, including total orders, revenue generated, and product-wise sales performance.
- The system must maintain a secure record of all sales transactions in the Sales database for future references and audits.
- The system must allow authorized personnel to edit or update order details in case of errors or customer requests.

e) Product management

To manage products, firstly the system should overview the features like “Add product” and “View product” of an inventory system.

- The system is used to add new items with product name, category, cost price, selling price, stock quantity, supplier information and so on.
- The system allows quick review of critical details about current stock levels.
- The system displays sales history like product names, supplier information, quantities and purchase dates to analyze the performance.
- The system helps user tracking spending, supplier performance and inventory trends.
- The system provides information on supplier data.

f) Payment

- The system must provide a secure payment process for buyers to make payments for the purchases they make.
- The system must allow customers to compare prices between similar products before proceeding with a purchase.
- The system must offer multiple secure payment options, some of them are:
  Bank Transfer, Credit Card, Online Banking, E-Sewa and Khalti
- The system must implement a secure payment gateway after product selection to complete transactions with no problems.
- The system must automatically calculate the total amount and taxes for the selected products or services.
- The system must send a payment confirmation message to the user after a successful transaction.
- The system must store all payment records in the account database to ensure data security and accessibility to only the authorized personnel.
- The system must include an interface to view payment history, showing past transactions, dates, and the total amount spent by the user.

3.2. Non-Functional Requirements

1. Security

The system would use strong security features to secure information from warehouses such as inventory records, login information, and log files. Restricted access will be limited, and all private information will be protected with encryption during storage and transmission.

2. Usability

The platform will include an easy-to-use user interface that allows warehouse staff to control supplies, maintain resources, and provide data. It will offer clear explanations and tool descriptions for newcomers while supporting different levels of technical skill.

3. Maintainability

The system will be independently established, which allows changes and updates without any delay. Users of the system will receive full documentation for fixing problems, maintenance, and future growth.

4. Performance

The system will respond in less than two seconds for basic tasks like stock requests, updates, and creating reports. It will also allow multiple users to use it at the same time without affecting its performance.

5. Scalability

As warehouse activities grow, the platform will be developed to deal with large numbers of clients, and multiple steps. It will stay secure and run smoothly even with more users and larger amounts of data.

3.2.1. Design and implementation Constraints

- To access the system, User must give the correct username and password to log in into the system.
- To make payment, users are free to pay through esewa, mobile banking, Khalti etc.
- The system will operate 24/7 for continuous warehouse operations.
- No individual is allowed to access other user data.

3.2.2. External interface

a) User Interfaces

- Users can log in to the system by providing personal information and can use the features.
- Users can on posts, upvote valuable contributions, and flag inappropriate content.
- The application has a properly structured registration form so that a new user can register easily.
- The system has a structured registration form to allow new users to register easily.
- Customers can compare product prices, check stock availability, and review their order history.

b) Hardware Interfaces

- The software is designed for all kinds of devices such as smartphones, tablets, and laptops.
- The software is compatible with built-in or external cameras and microphones for virtually interaction.

- The system is also compatible with printers and scanners for generating physical documents such as annual reports.
- The system can also interface with external devices like scales for weighing inventory and packaging purposes.

c) Software Interfaces

- The software will be secure with payment gateways, such as PayPal, Stripe, and local payment systems like E-sewa and Khalti, for seamless financial transactions.
- The software will communicate with a database management system like MySQL or PostgreSQl.
- The system will integrate with authentication systems like OAuth or LDAP to ensure secure user login for users.
- For location-based services (e.g., delivery tracking), the system may use Google Maps to display delivery locations and routes.
- The software will integrate with third-party ERP systems or supplier APIs to synchronize product and stock updates automatically.

d) Communication Interfaces

- Client-server communication is using secure and efficient transmission protocols, like HTTPS, to protect the privacy of information during transit.
- The system offers well-documented Application Programming Interfaces (APIs) to ensure smooth interaction with other services.
- To provide consistent and dependable user experience across multiple platforms, the communication interfaces are compatible with a wide range of web browsers.

3.3. Other Non-Functional requirements

a) Sustainability

- Codebase modular for updates and bug fix.
- Complete documentation for Developers and Users.

b) Compliance

- Follows a standard framework like ISO 27001 for the security of the information.
- Observance of local e-commerce and financial regulations.

c) Localization

- Supporting language for local users.
- Displaying of currency.

d) Disaster Recovery

- Recovery of system in 2 hours in case there is a failure in the system.
- Backup in the cloud with a Sustained period of 15-30 days.

4. Group tasks

4.1. Environmental model specification

4.1.1. Data Flow Diagram (DFD)

DFD stands for Data Flow Diagram, refers to the graphical representation of the flow of data across a system, showing processes, data sources, destinations, and data storage. It is an effective technique for system analysis and design that makes it possible to express the elements, data, and interaction of the system in an easy to-understand way. It helps to understand the flow of data in a structured system (Scott Robinson, 2024).

The symbols used during the development of DFD are listed below:

- **Process:** Represented by circles, ovals, or rectangles, processes are used to convert incoming data flow into outgoing data flow.
- **Data Flow:** Represented by arrows, it indicates the path and direction of the data as it passes through the system.
- **Data Store:** Represented by two horizontal lines, it indicates a data repository or storage where data is stored.
- **External Entity:** Represented by rectangles or squares, external entities interact with the system.

Actually, we have to make three levels of DFD in our coursework and we expand the level 0 DFD to level 1 and furthermore expand to level 2, we use grammatical phrase that narrative that describes the context-level process (bubble). This is how we isolate all nouns and verbs. The Tech corporation's system allows user registration. The verbs are written inside bubbles which are processes and noun are written on the arrow line which are data flow of the system.

4.1.1.1. Context Level Data Flow Diagram (Level 0 DFD)

Level 0 is the early steps of Data Flow Diagram (DFD), which provides building blocks of the entire system. It shows the major processes, data flows, and data stores in the system, without providing any details about the internal workings of these processes. It represents the entire system as a single bubble with input and output data indicated by incoming/outgoing arrows (geeksforgeeks, 2024). The context level diagram is mentioned below:

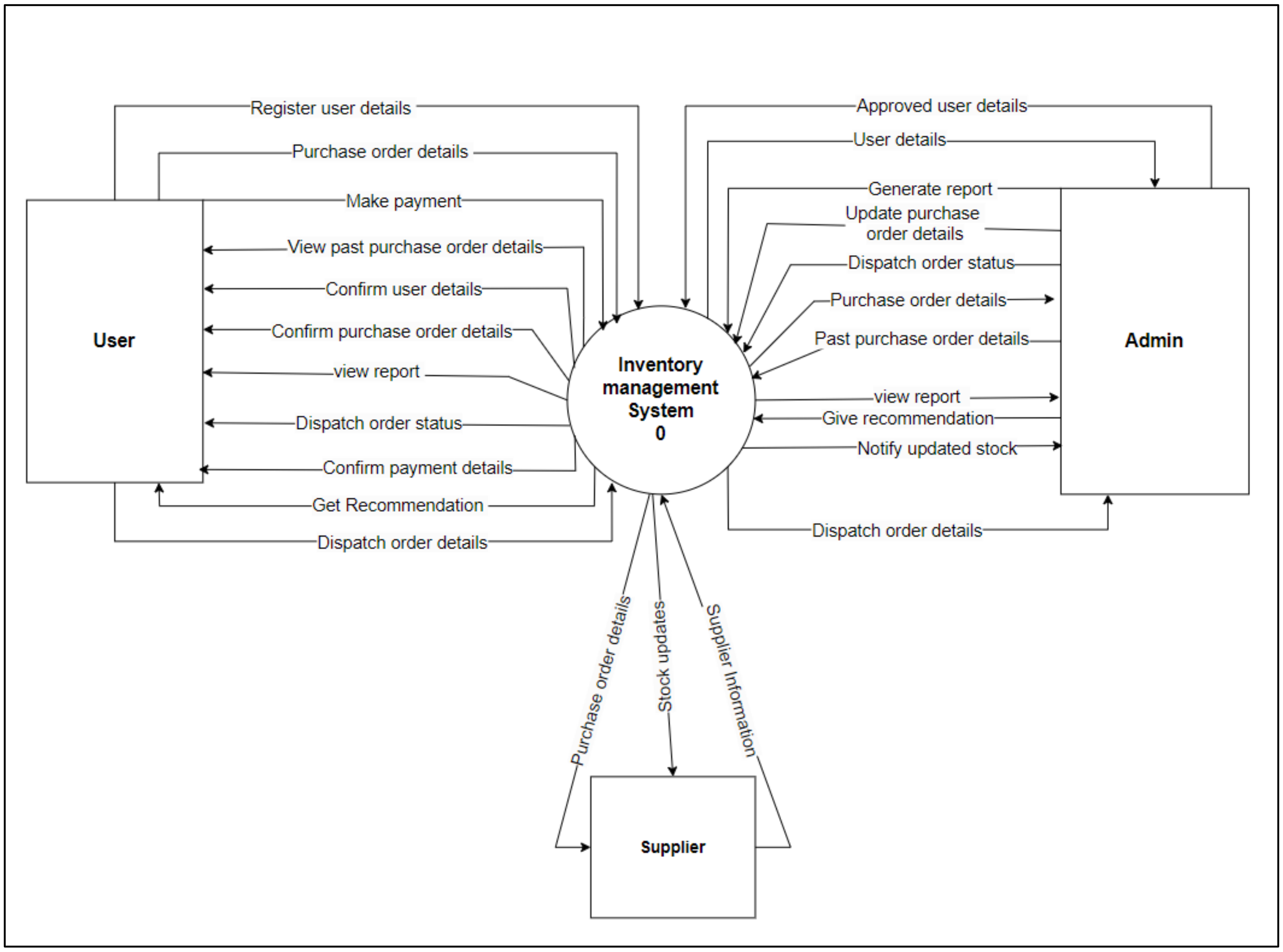

Figure 1: Level 0/ Context level DFD for whole system

4.1.1.2. Level 1 DFD (Data Flow Diagram)

In 1-level Data Flow Diagram (DFD), the context diagram is decomposed into multiple bubbles/processes. In this level, we break down the high-level process of 0-level Data Flow Diagram (DFD) into subprocesses. The data flows and data stores associated with each sub-process are also shown (geeksforgeeks, 2024). The level 1 diagram of the whole system is:

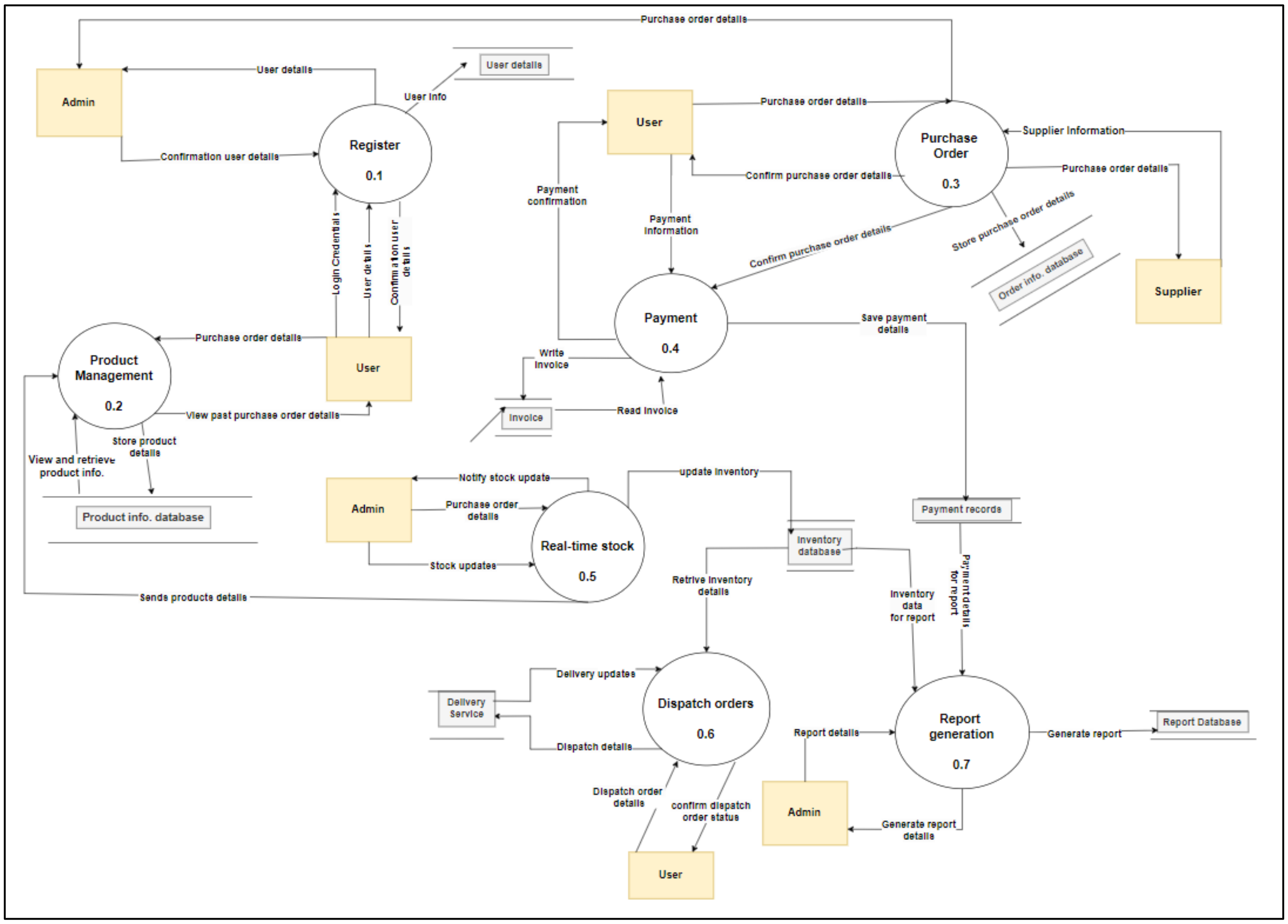

Figure 2: Level 1 DFD for whole system

4.1.1.3. Level 2 Data Flow Diagram

2-Level Data Flow Diagram (DFD) goes one step deeper into parts of 1-level DFD. It can be used to plan or record the specific/necessary detail about the system's functioning (geeksforgeeks, 2024).

The level 2 diagrams are mentioned below:

4.1.1.3.a. Level 2 Data Flow Diagram of Registration

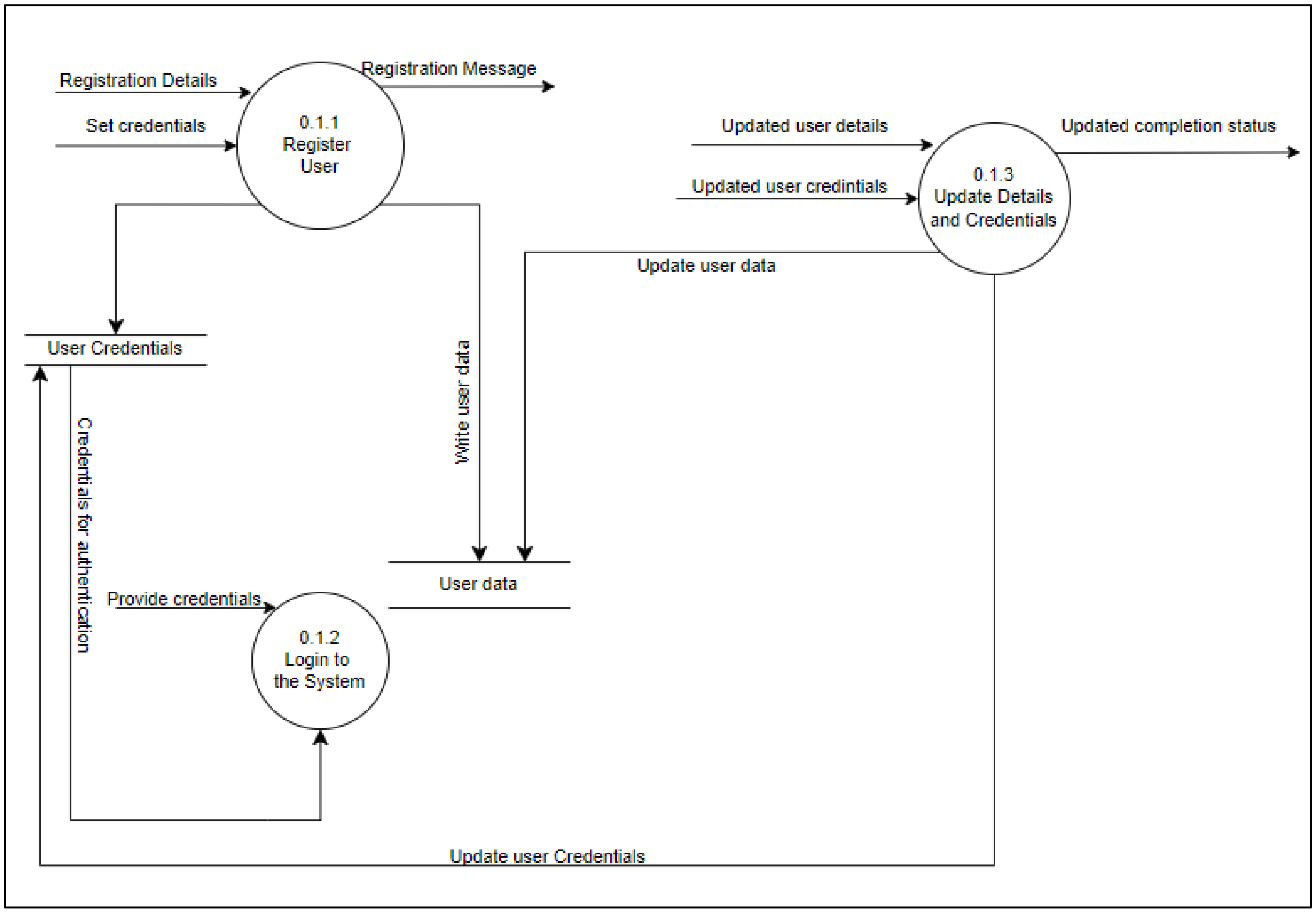

Figure 3: Level 2 DFD for Registration

4.1.1.3.b. Level 2 Data Flow Diagram for payment

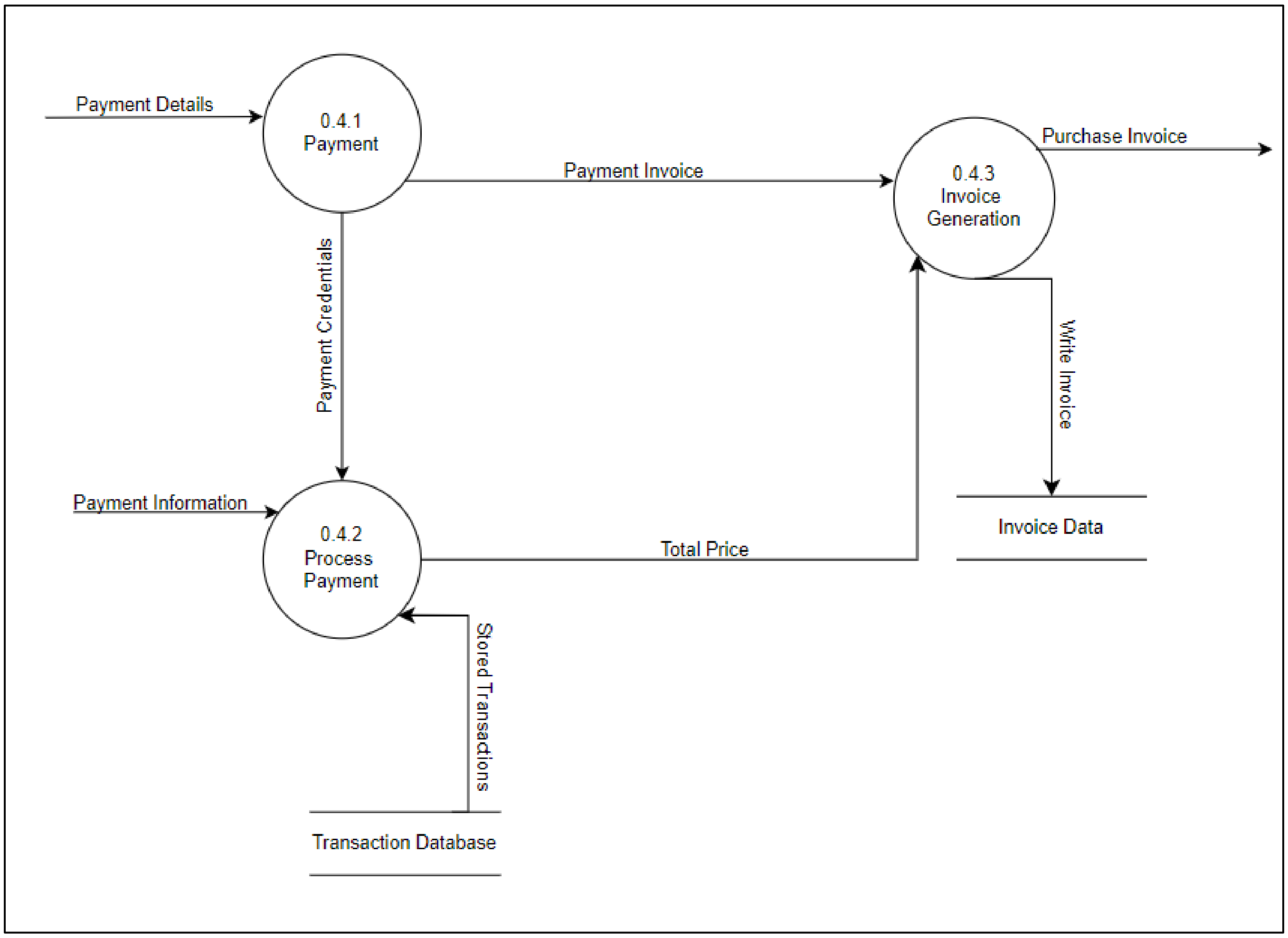

Figure 4: Level 2 DFD for Payment

4.1.1.3.c. Level 2 Data Flow Diagram of inventory management

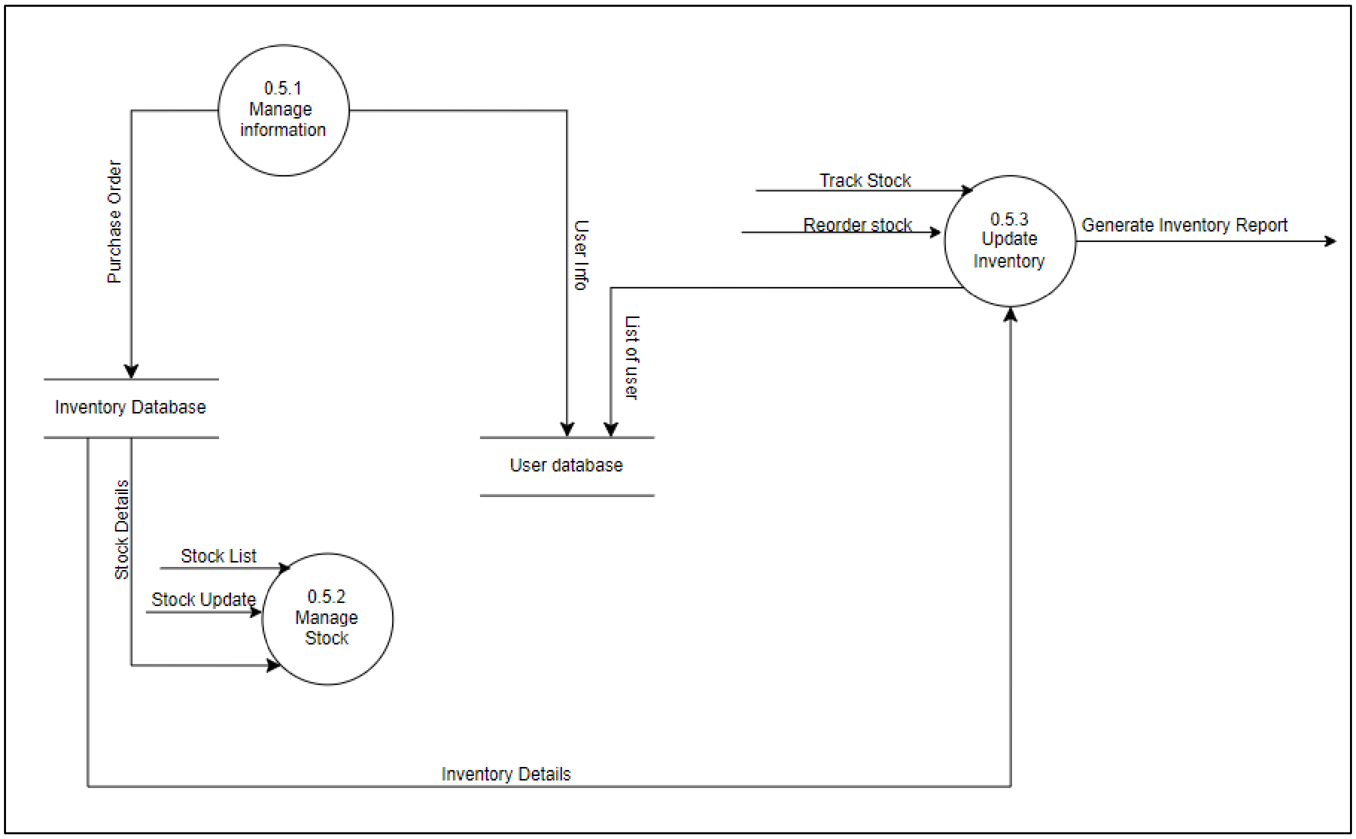

Figure 5: Level 2 DFD for Inventory Management

4.2.1. Internal model specification for the whole system

4.2.1.1. Entity Relationship Diagram (ERD)

An Entity Relationship Diagram (ERD) is a visual representation of the relationship among entities in a database. In DBMS, an ER Diagram is crucial for database design. It acts as a framework with specific symbols designed to specify how database entities are related to one another. The main three components are involved in creating an ER diagram including entities, attributes, and relationships. In ER diagram, different symbols are used, such as rectangles to represent entities, ovals to represent attributes, and diamonds shapes to represent relationships (smartdraw, 2024).

The components involved in creating ER diagram are described below:

a) Entity

An entity is an object that exists in the real world. Rectangle shapes are commonly used to represent entities in an ERD.

Example:

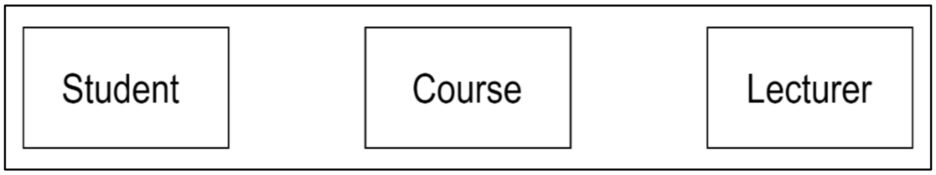

Figure 6: Examples of Entity

b) Attributes

Attributes are properties or characteristics of entities that describe the features of an entity.

Example:

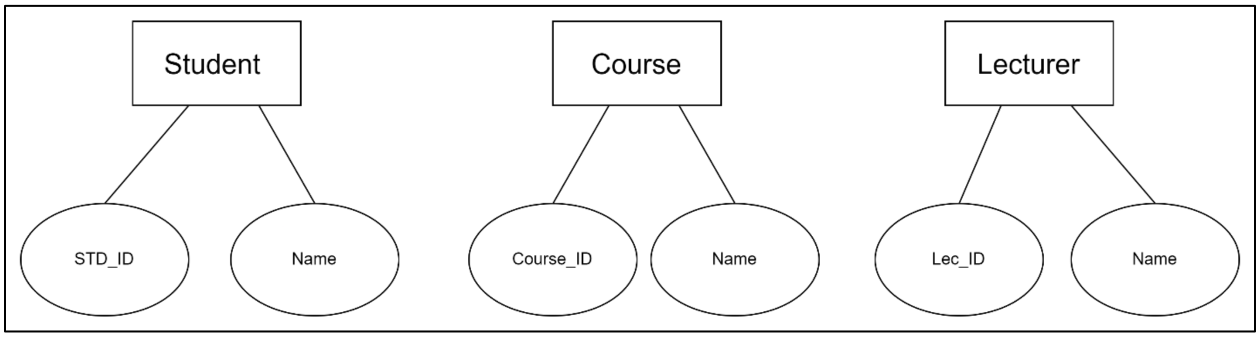

Figure 7: Examples of attributes

c)Relationship

Relationships are a logical connection between two or more entities.

Example:

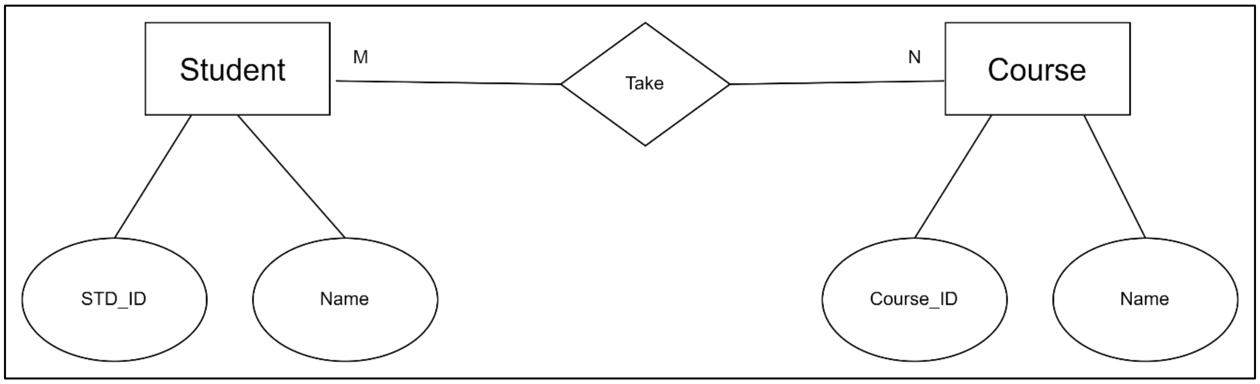

Figure 8: Examples of connecting entities

4.2.1.2. Entity Relationship Diagram of whole system

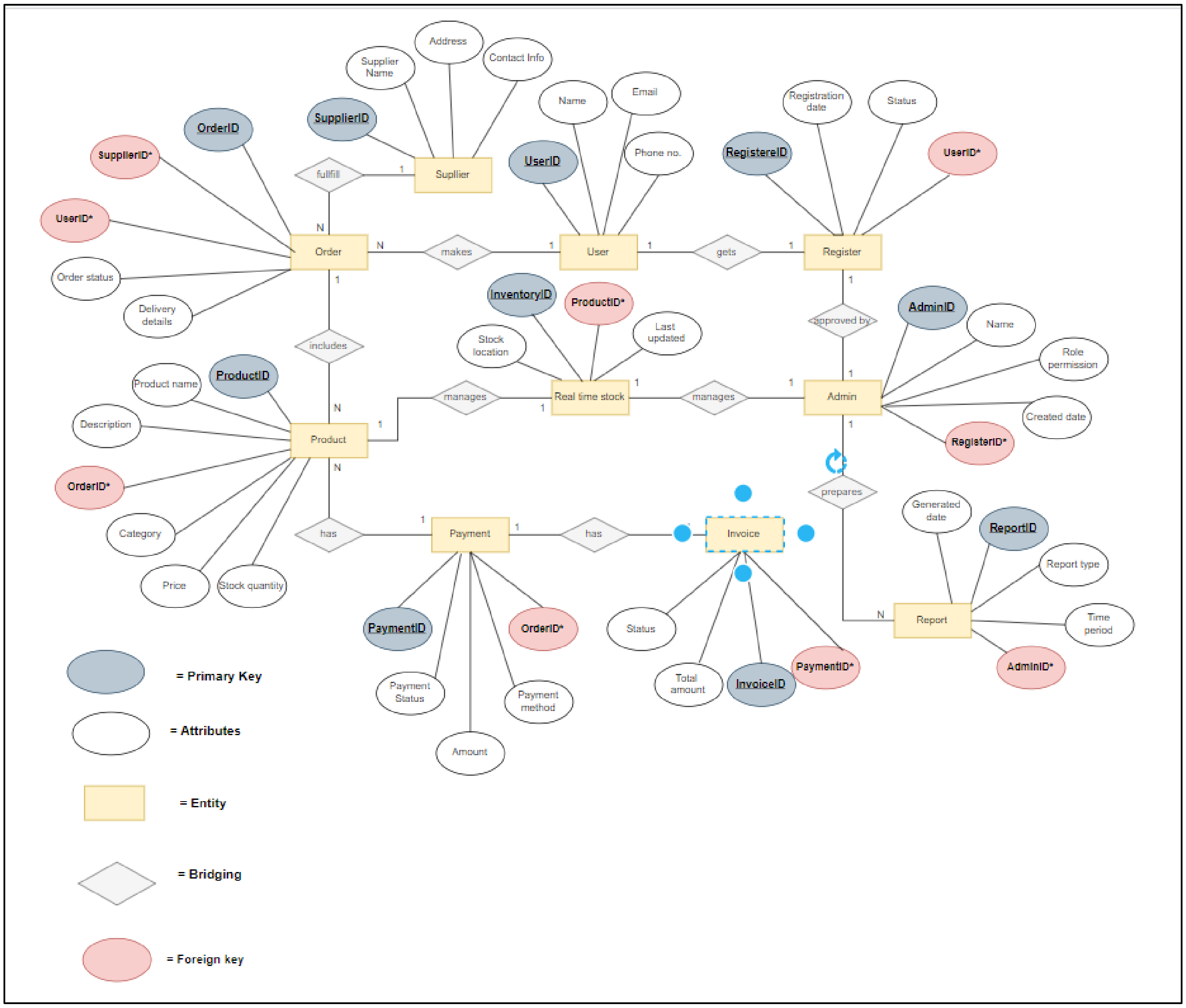

Figure 9: Entity Relationship Diagram of whole system

4.2.1.3. Data Dictionary

Data dictionary is a file or a set of files that includes a database's metadata. The data dictionary is an essential component of any relational database.

Components of the Data Dictionary for IMS:

The text below outlines the data dictionary for the whole system:

User = Customer

Register = register details + register records * DATA STORE*

Register records = { register details} *

Register details = **<u>RegisterID</u>** + RegistrationDate + Status + UserID*

<u>RegisterID</u> = int

RegistrationDate= date

Status= string

UserID* (FK) =int

Register User =command

User = user details + user records * DATA STORE*

User records = { user details} *

User detail = **<u>user_ID</u>** + username + address + email + phone number

<u>user_ID (PK)</u> = int

username= string

address = string

email = string

phone number= int

Admin = Admin details + Admin records * DATA STORE*

Admin records = { Admin details} *

Admin details= **<u>AdminId</u>** + registeredID* + Adminname + createddate+role_selection

<u>AdminId (PK)=</u> int

registeredID* (FK)= int

Adminname= string

Createddate = date

Role_selection= string

Supplier = supplier details + supplier records * DATA STORE*

Supplier records = { supplier details} *

Supplier details = supplierId + suppliername + address+ contactinfo

<u>supplierId</u> (PK)= int

suppliername = string

address= string

contactinfo= int

Purchase order = command

Order = order details + order records * DATA STORE*

Order records = { order details} *

Order details =**<u>OrderID</u>** + OrderDate + OrderQuantity + ProductID* + AdminID* + supplierID*

<u>OrderID (PK)</u> = int

OrderStatus = string

DeliveryStatus = string

ProductID* (FK) =int

AdminID* (FK) =int

supplierID* (FK) = int

Product = product details + product records * DATA STORE*

Product records = { product details} *

Product details = **<u>ProductID</u>** + Product Name + Product Description + Price + InventoryID* + OrderID* + category+ stock quantity

<u>ProductID (PK)</u> = int

Product Name = string

Product Description = string

Price = int

InventoryID* (FK) = int

OrderID* (FK)= int

Category = string

stock quantity = int

Make payment= command

Payment = payment details + payment records * DATA STORE*

Payment records = { payment details} *

Payment detail = payment method+ **<u>payment ID</u>** + amount+ Status+ orderID*

<u>payment ID</u> (PK) = int

payment method = string

amount = int

Status= string

orderID* (FK) = int

Create invoice= command

Invoice = Invoice details + Invoice records * DATA STORE*

Invoice records = { Invoice details} *

Invoice details= <u>InvoiceID</u> + paymentID* + total amount+ status

<u>InvoiceID (PK)</u> = int

paymentID* (FK) = int

total amount = int

status = string

View real time stock= command

Real time stock = real time stock details + real time stock records * DATA STORE*

Real time stock records = { Real time stock details} *

Real time stock details =**<u>InventoryID</u>** + StockLocation + ProductID* + LastUpdated

<u>InventoryID (PK)</u> = int

StockLocation = string

ProductID* (FK) = int

LastUpdated = string

Generate report = command

Report = Report details + report records * DATA STORE*

Report records = { Report details} *

Report details = **<u>ReportID</u>** + GeneratedDate + ReportType + TimePeriod + AdminID*

<u>ReportID (PK)</u> = int

GeneratedDate = date

ReportType = string

TimePeriod = string

AdminID* (FK) = int

Generate bill= command

Confirm supplier = command

Confirm dispatch order = command

4.2.1.4. Process specifications (Pspecs)

4.2.1.4.a. Process 1:

- **Process name: Register**
- **Description:** This process allows new users to register and old to login the system.
- **Input data flow:** User details like UserId, name, email, etc.
  Login credentials like username and password.
- **Output data flow:** Confirmation of successful registration.
- **Detailed logic:**
  - ➔ A customer enters their details. If new get new unique ID which is stored in the database. If not just simply use user credentials.
  - ➔ The system checks if the details given are valid.
  - ➔ The system updates with valid details.

4.2.1.4.b. Process 2:

- **Process name:** Purchase order
- **Description:** This process allows the admin to check product availability and order the product based on the provided purchase order details.
- **Input data flow:** Admin registration details for checking and ordering the available purchase order details.
- **Output data flow:** confirmation of successful payment, updated product and purchase order records.
- **Detailed logic:**
  - ➔ After receiving registration details, the admin checks the product availability based on the purchase order details in the datastore named “Product Info”.
  - ➔ During checking availability of product, the system allows to check through datastore name “Product info” if the product is available, it moves forward to another process if not it is dismissed on the spot.

- ➔ After checking the product availability, we came to know about the available product by sending the preferrable choice of purchase order details.
- ➔ During purchasing order, at first, we select product then, place order, confirm order, generate bill and track order for successful purchasing.
- ➔ Also, all the details are sent to the supplier and store in the database for updating and retrieving data.

4.2.1.4.c. Process 3:

- **Process name: Payment System**
- **Description:** User are allowed to pay the product they order and want to purchase.
- **Input data flow:** User information (Customer or Admin), payment information
- **Output data flow:** Confirm payment details
- **Detailed logic:**
    - ➔ The process is carried out once a user confirms the order details.
    - ➔ The process of payment is carried out by various service types like virtual (mobile banking through various apps) and physical present (cheque, atm card and debit or credit cards).
    - ➔ The final payment details are stored in the payment database.

4.2.1.4.d. Process 4:

- **Process name: Real time stock**
- **Description:** The process allows an admin to update, retrieve and store the purchase order details and informs the supplier too.
- **Input data flow:** store new purchase order details
- **Output data flow:** update the stock of the system
- **Detailed logic:**
    - ➔ After purchasing any new product, the system automatically updates and stores in the inventory database for future flexibility and scalability.

➔ It allows purchase orders to see the product availability and to generate reports after confirming order and payment.

➔ Also, allows the user to see the past dispatch details for user satisfaction.

4.2.1.4.e. Process 5:

- **Process name: Dispatch order**
- **Description:** The process allows you to view and update the dispatch order details and services.
- **Input data flow:** dispatch details, delivery updates
- **Output data flow:** Confirm dispatch order details
- **Detailed logic:**

  ➔ After purchasing any product, the system provides the dispatch order details to the new user (Admin or Customer) to know the good services of corporation.

4.2.1.4.f. Process 6:

- **Process name: Report generation**
- **Description:** The process generates the report after purchasing products and confirming payment.
- **Input data flow:** Purchase order details, generate bill details
- **Output data flow:** confirm details of payment and purchase order
- **Detailed logic:**

  ➔ After purchasing any product, we have to confirm the bill so that we can update it in the inventory database and proof of purchasing product.

  ➔ So, we generate reports to know the no. of quantity, total profit/ loss, dispatch order details, customer preferences product and so on.

  ➔ Lastly, we update all in the inventory database to update the stocks for future flexibility.

4.3. Design specification

4.3.1. Structure Chart

Structure chart is the hierarchical structure of modules which breaks down the entire system into simplest form with individual description of each function and sub-functions (geeksforgeeks, 2024).

Symbol in structure chart:

a) **Module:**
It represents the process of the system which are of three types: Control, sub-module and library module.
Control module: It consists of more than one sub-module branches.
Sub-module: It is the child module or module of another module.
Library module: It is reusable for any module. (geeksforgeeks, 2024)

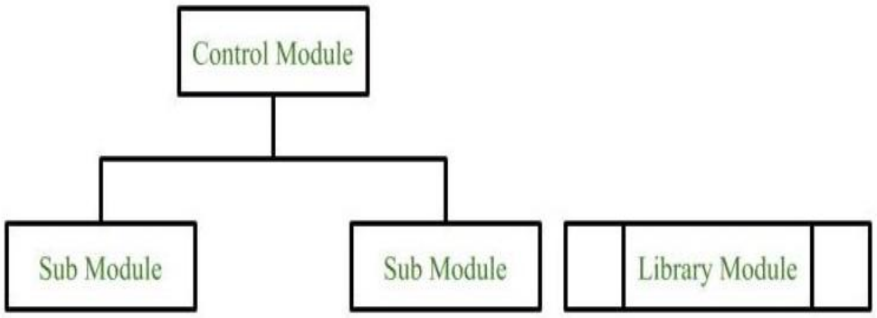

Figure 10: Example of module in structure chart

b) **Conditional call**
It represents control module which can select either one of them with some condition basis (geeksforgeeks, 2024).

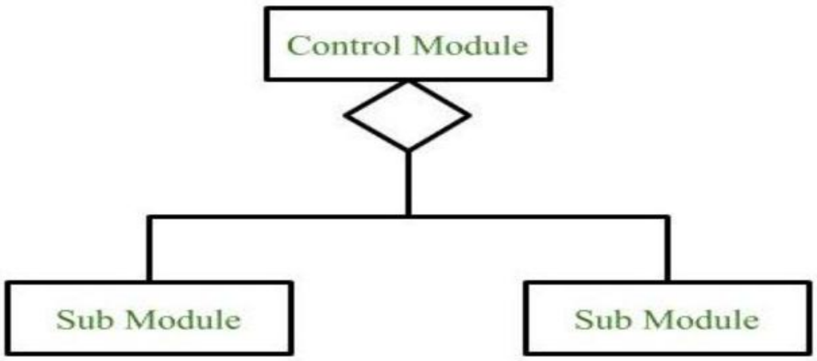

Figure 11: Example of conditional call of structure chart

c) **Loop (Repetitive call module)**
It represents loop in module by the sub-module with curved arrow notation (geeksforgeeks, 2024).

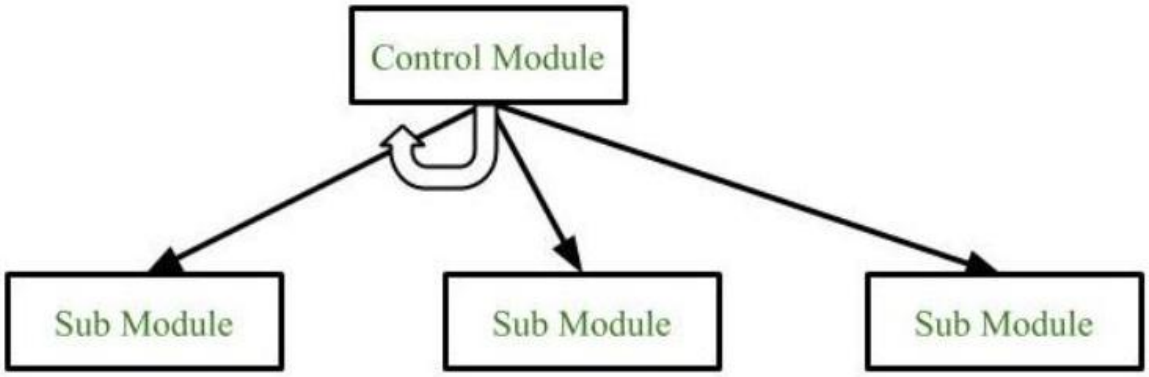

Figure 12: Example of loop in structure chart

d) **Data flow**
It represents the flow of data between the modules which is represented by a directed arrow with empty circle at the end (geeksforgeeks, 2024).

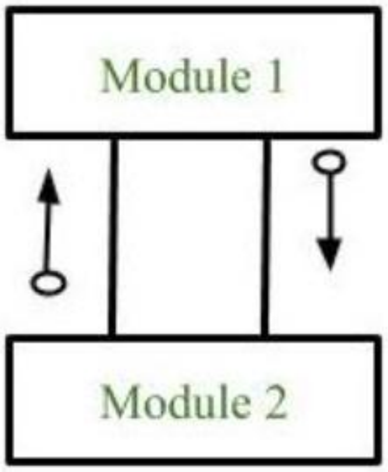

Figure 13: Example of data flow in structure chart

e) Control flow

It represents the flow between the modules which is represented by directed arrow with filled circle at the end. It indicates that a criteria has been met, providing confirmation for the system to proceed (geeksforgeeks, 2024).

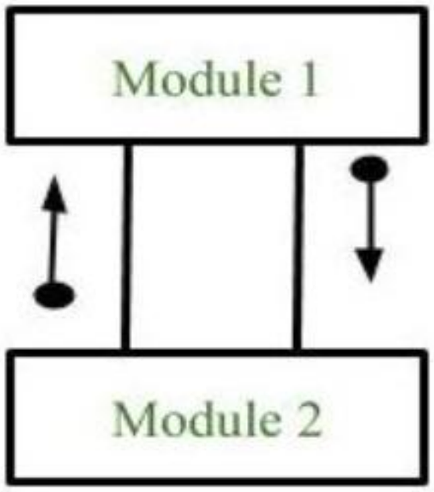

Figure 14: Example of control flow in structure chart

Structure chart

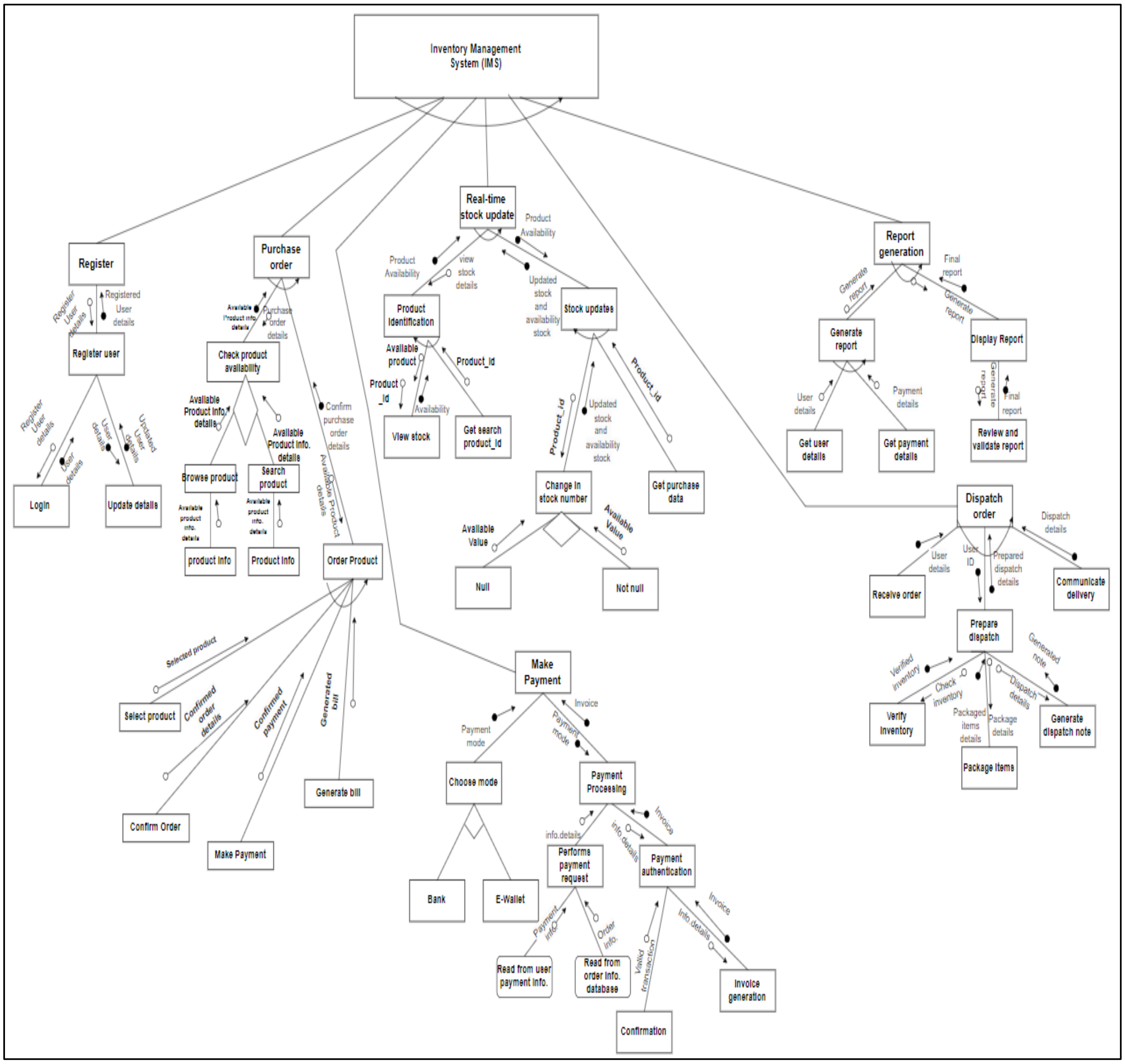

Figure 15: Structure chart of whole dfd

4.4. Progress Logs/ Assignment diary

4.4.1. Group members details and responsibility

During the completion of the project, we did face a lot of problems and difficulties and are assigned with various tasks and responsibilities to each member of the team to complete tasks. To remove the obstacles, we had different milestone submissions where we had to submit some steps before completing the whole project. All the team members agreed to discuss the problems which may encounter in future by holding meeting virtually or physically mode. This meeting and milestone were extremely helpful for the completion of the project. The division of the tasks is outlined below:

S.N	Name	Member Responsibilities
1.	Salina Thing	• Tasks on Individual work “Purchase order”. • Works on “Benefits, risks and mitigating and conclusion” of business case. • Works on functional requirements: User access, purchase order and product management. • Had a role in designing of DFD level 0, 1 and 2 of the whole system. • Had a major role in Entity Relationship Diagram. • Took lead and assigned each group member the responsibilities that had to be fulfilled. • Had a major role in Entity Relationship Diagram. • Handled the formatting and compilation of the documentation. • Works on progress log:group responsibilities. • Works on process specification of group tasks.
2.	Kriti Lama	• Works on Individual work “Report generation”. • Works on SRS functional requirements. • Works on functional requirements: generate report, sales management and payment. • Had a role in design of DFD level 0, 1 and 2 of the whole system. • Works on “Goals and timeline” of business case. • Had a role in Entity Relationship Diagram. • Works on structure chart of the whole system.

		• Works on Assumptions diary "Assumptions". • Works on data dictionary "report part".
3.	Rahul Jaiswal	• Works on Individual work "Dispatch Order". • Works on "Project description and problem statement" of business case. • Had a role on "Entity Relation Diagram definition". • Works on "External interface" of non-functional requirement. • Had a major role on structure chart of the whole system. • Works on level 0, 1 and 2 DFD of the whole system.
4.	Aastha Aryal	• Works on Individual work "Real-time-stock-update". • Works on "Scope" of business case. • Works on non-functional: "design and implementation" and non-functional requirements. • Had a role in designing of DFD level 0, 1 and 2 of the whole system. • Works on structure chart of the whole system.
5.	Krishal Acharya	• Works on individual task for "Make Payment". • Works on "Aims, objectives and member tables" of business case. • Works on "other non-functional requirements". • Had a role in design of DFD level 0, 1 and 2 of the whole system. • Works on data dictionary of whole system. • Works on structure chart of the whole system. • Works on meeting logs of progress log.

Table 2: Table of group responsibilities

4.4.2. Meeting details

Meeting Entry 1

Meeting Entry 1 **Date: 03/12/2024** **Location: Canteen** **Start Time: 01:00 PM** **End Time: 02:00 PM**
Discussion: - We discussed about the coursework topic and gathered the team members. Task concluded for meeting: - We gathered all the interested students and formed a group of 5 students

Team Members	**Students Signature**
Aastha Aryal	Aastha
Krishal Acharya	Krishal
Kriti Lama	Kriti
Rahul Jaiswal	Rahul
Salina Thing	Salina

Table 3: Meeting log 1

Meeting Log 2

Meeting Entry 2 **Date: 06/12/2024** **Location: Alumini Block** **Start Time: 11:00 AM** **End Time: 02:00 PM**
Discussion: - We gathered and group discussed and properly anaylzed the coursework question Task concluded for meeting: - Created a plan to go ahead with the coursework

Team Members	Students Signature
Aastha Aryal	Aastha
Krishal Acharya	Krishal
Kriti Lama	Kriti
Rahul Jaiswal	Rahul
Salina Thing	Salina

Table 4: Meeting log 2

Meeting Log 3

Meeting Entry 3 **Date: 10/12/2024** **Location: Canteen** **Start Time: 11:00 AM** **End Time: 12:30 PM**
Discussion: - We decided to gather and correct if any mistake of the team member Task concluded for meeting: - We submitted the Milestone 1

Team Members	**Students Signature**
Aastha Aryal	Aastha
Krishal Acharya	Krishal
Kriti Lama	Kriti
Rahul Jaiswal	Rahul
Salina Thing	Salina

Table 5: Meeting log 3

Meeting Log 4

Meeting Entry 4 **Date: 16/12/2024** **Location: London Block** **Start Time: 11:00 AM** **End Time: 12:30 PM**
Discussion: - Multiple errors we found in some individual tasks so we stay after class to correct those mistakes and provide some insights Task concluded for meeting: - After students we made to correct their portions and we arranged the document for 2nd Milestone and submitted it

Team Members	**Students Signature**
Aastha Aryal	Aastha
Krishal Acharya	Krishal
Kriti Lama	Kriti
Rahul Jaiswal	Rahul
Salina Thing	Salina

Table 6: Meeting log 4

Meeting Log 5

Meeting Entry 5

Date: 23/12/2024
Location: Nepal Block

Start Time: 02:00 PM

End Time: 04:00 PM

Discussion:

- We developed a plan for 3rd Milestone. We gave everyone their responsibility to develop structured charts and module specification

Task concluded for meeting:

- Everyone was required to complete their task and send it to leader through Outlook and submit it by deadline

Team Members	**Students Signature**
Aastha Aryal	Aastha
Krishal Acharya	Krishal
Kriti Lama	Kriti
Rahul Jaiswal	Rahul
Salina Thing	Salina

Table 7: Meeting log 5

Meeting Log 6

Meeting Entry 6 Date: 02/01/2025 Location: Canteen Start Time: 11:00 AM End Time: 03:00 PM
Discussion: - For final submission, we gathered and checked all of our mistakes with module leader and we corrected all the mistakes as a team together Task concluded for meeting: - The module leader uploaded the file in MST in accordance to all the team members approval for Final Submission

Team Members	Students Signature
Aastha Aryal	Aastha
Krishal Acharya	Krishal
Kriti Lama	Kriti
Rahul Jaiswal	Rahul
Salina Thing	Salina

Table 8: Meeting log 6

4.4.3. Assumptions

- The system must support three users: Admin, Customers and suppliers with monitored access.
- Each user must be allocated with a unique ID and phone number.
- Purchase records must contain details such as supplier, product name, quantity, price, and purchase date.
- Past purchase records can be made accessible to admins and the specific customer
- The sales report may help the admin track the total number of sales and profit and loss occurred during that specific time.
- Sales orders contain delivery methods, addresses, and contact details.
- The system updates the inventory in real-time.
- Historical data for each product, including sales and purchases, are stored to study trends for research purposes in the future.
- Customers can compare product prices among similar items before purchase.
- Multiple payment options are available, such as credit/debit cards, e-wallets, and bank transfers.
- The system uses role-based access control to protect sensitive information like payment data, addresses, phone numbers, supplier information etc.
- The interface is designed to be user-friendly, with separate dashboards for admins, customers and suppliers.
- The system provides error messages for invalid actions like duplicate entries or low stock alerts.

4.4.4. Omission

The some of the omission comes from our assumptions are mentioned below:

- Clarifying if purchase records include additional fields like tax amounts, discounts, or order statuses (e.g., pending, completed).
- Expand on individual product performance.
- Mentioning if user can update or reset the system.
- Indicating if admin can deactivate accounts or modify access permissions to users.
- Informing about the error in processing to the users through messages.
- Ensuring if the system has provided backup and recovery in case of failure or corruption.

5. Individual tasks

5.1. Purchase Order

5.1.1. Environmental model specification

Name: Salina Thing (Leader)

Section: L2C1

London met ID: 23047540

College ID: np01cp4a230017

The Global Tech Corporation has requested to improve purchase order management feature in the system for effective implementation. The system allows admins to create, view and update purchase orders necessary before validated processing. Also, the system generated reports to ensure accuracy, stored validate data and give credentials services.

5.1.1.a. Context level diagram

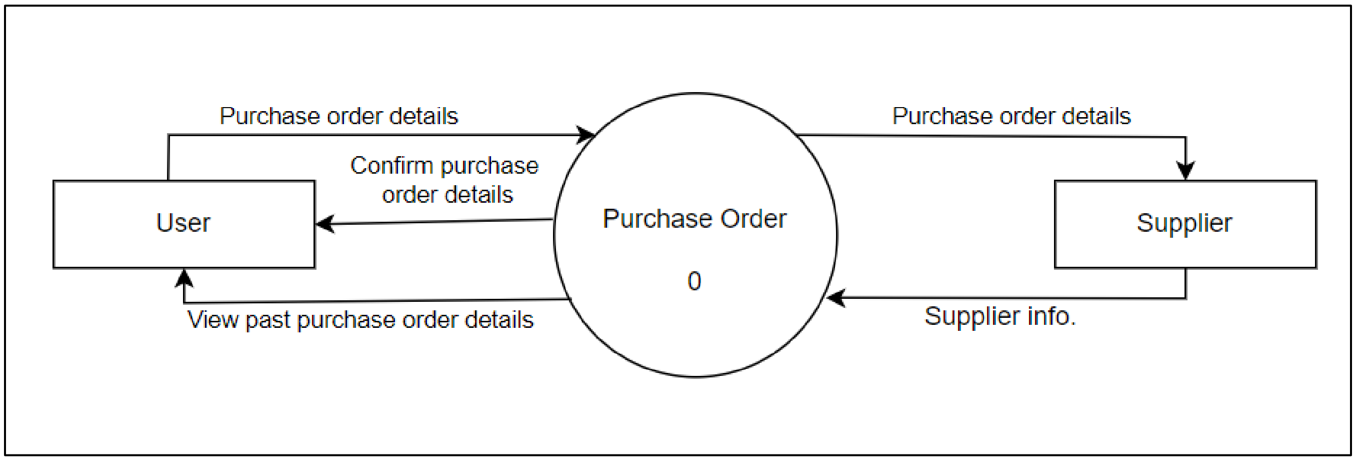

Figure 16: Level 0 of purchase order

Description:

User = customer (According to my assumption).

In the context diagram, there are two external entities or terminators like user and supplier interact with the system (purchase order system). Here, User is the one who initiates the process by entering purchase order details have accessed to review past purchase order details and confirm purchase order. As the supplier is responsible for receiving order details and providing supplier information to complete the process.

5.1.1.b. Level 1 of Data Flow Diagram

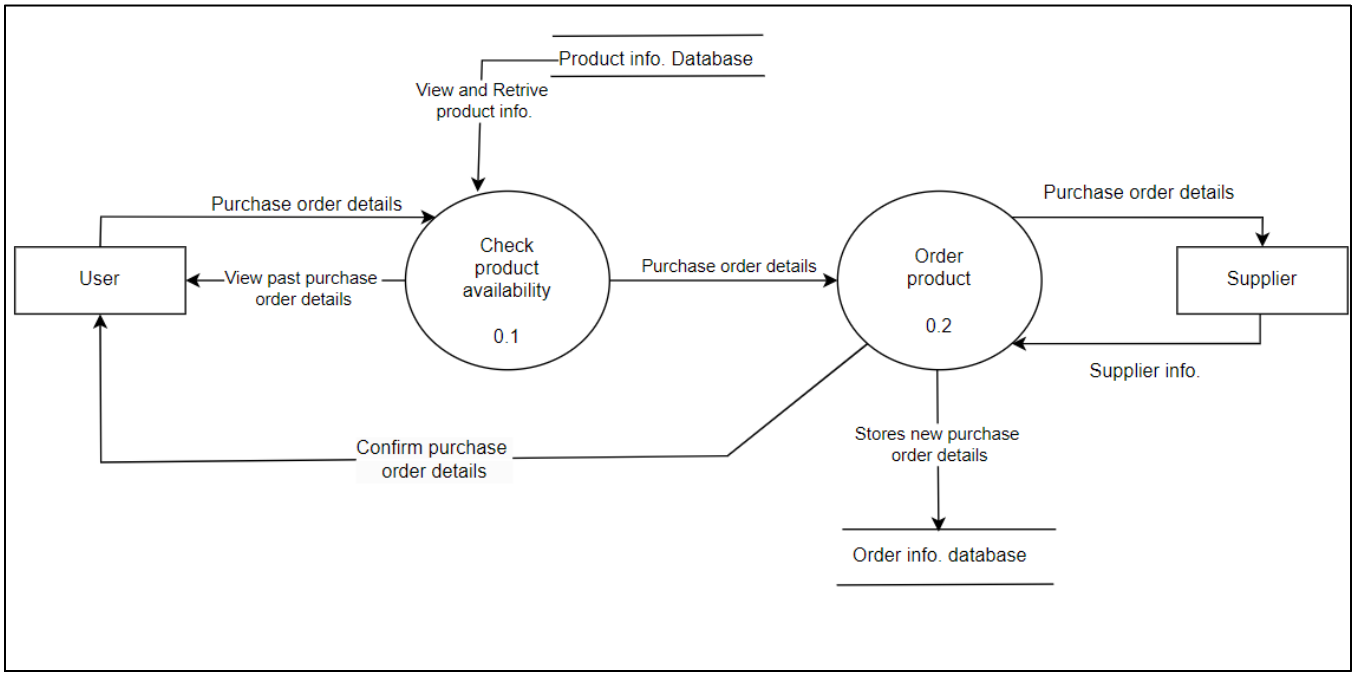
Product info. Database
View and Retrive product info.
Purchase order details
User
View past purchase order details
Check product availability 0.1
Purchase order details
Order product 0.2
Purchase order details
Supplier
Supplier info.
Confirm purchase order details
Stores new purchase order details
Order info. database

Figure 17: Level 1 of Purchase order

Description:

In level 1 diagram, we have two entities (User and supplier) and three processes. The process

of purchasing order has been divided into 2 parts which are discussed below: -

- ➔ **Process 1. (Check product availability)**
  In this process, we view product information from database " Product info. database" to interact by retrieving and verifying product information based on user input. If the product is available, it allows further steps; otherwise, the process ends. Also, this process allows user to view past purchase details and transfer further process if the product is available.

- ➔ **Process 2. (Order product)**
  Once product availability is confirmed, the user confirms the order, and the system stores the order details in the datastore (Order info) and sends the order details to the supplier. Supplier details are sent by supplier which is linked to the order process. The system handles the placement of orders with supplier and updates the 'Order info. database' with confirmed order details. The database is stored for future references.

5.1.1.c. Level 2 of Data Flow Diagram

Description:

In level 2 diagram, order product has been elaborate into 4 sub-processes which are Select product, place order, confirm order and track order which are described in detail in below:

- ➔ **Select product (0.2.1)**
  The **user views and selects the products** based on required **purchased order details** which are stored in the **datastore (Product Info. database).** The **output for** further process is **"Selected product details".**

➔ **Confirm order (0.2.2)**

After selecting a product, the **user** confirms an order then stores new purchase order details in the database “**Order info. database”.** Also, it sends **purchase order details** to supplier “External entity” so in return it provides **“supplier info”.** Then, the process moves forward with database “ **Order info. Database”.**

➔ **Make payment (0.2.3)**

The process allows the user to makes payment after a complete order. The input is **“Confirmed purchase Order details**” and output is **confirmed payment.** Then, we move forward process for generating bill.

➔ **Generate bill (0.2.4)**

Once the payment is done, a bill is generated **and stores confirmed payment** of confirmed purchase order in the **datastore “invoice or bill**” then sends to the **user to keep records.**

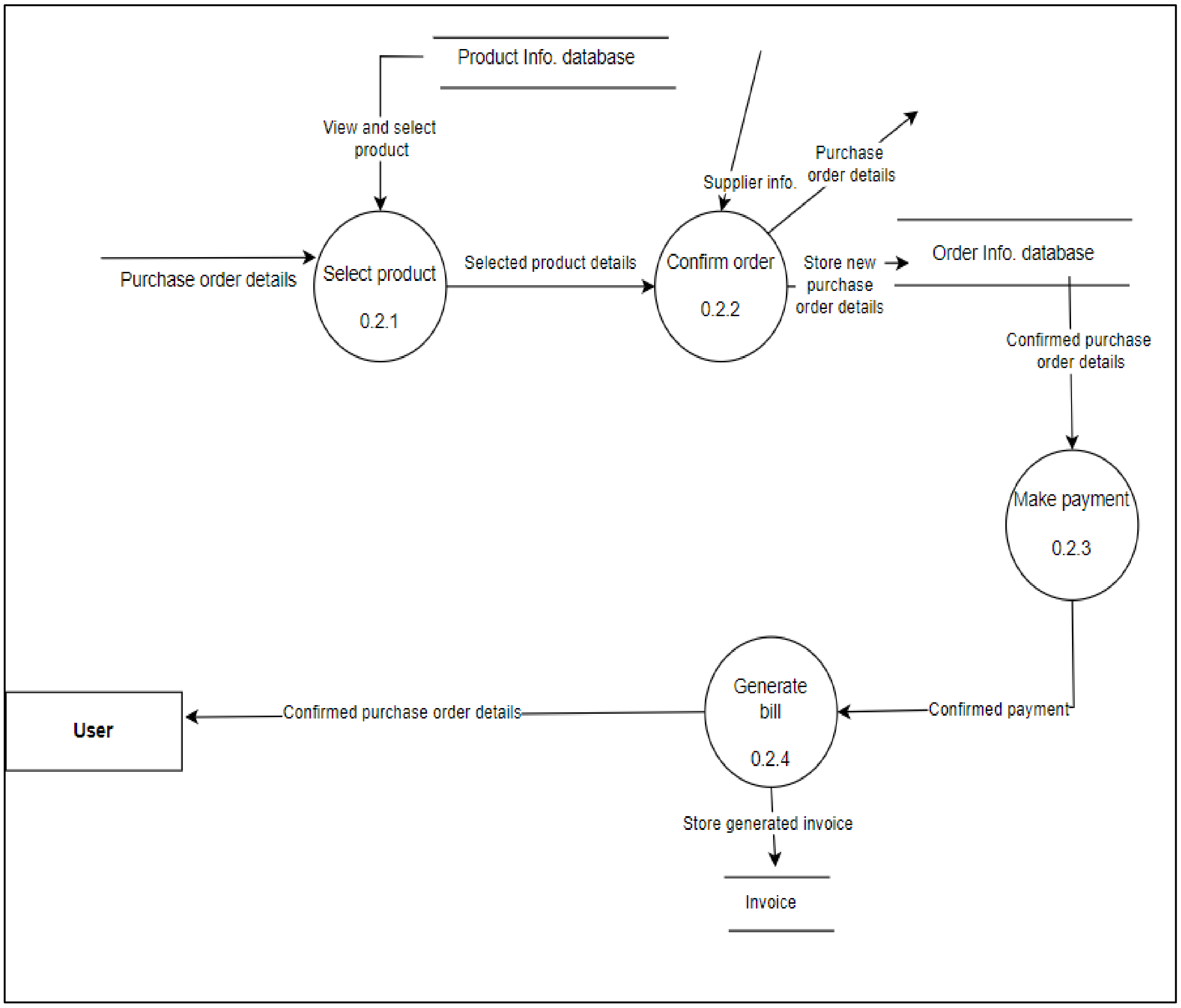

Figure 18: Level 2 of purchase order

5.1.2. Design specification

5.1.2.a. Structure chart

The figure below represents the general structure of purchase order. As seen in the figures, the main module is responsible for central entry point for purchasing order process. **Purchase order** direct the flow to check availability of product and order product. **Check product availability** verifies whether the product is in stock or not whereas **ordering product** orders the product after selecting product, confirm order, make payment and generate bill. Here, in checking product availability, we used **the decision** to represent SELECTION and split the charts sequence into multiple paths. And in the order product, we use **repetition** to allow multiple times. Also, we use **call line, parameter and control parameter** in the structure chart.

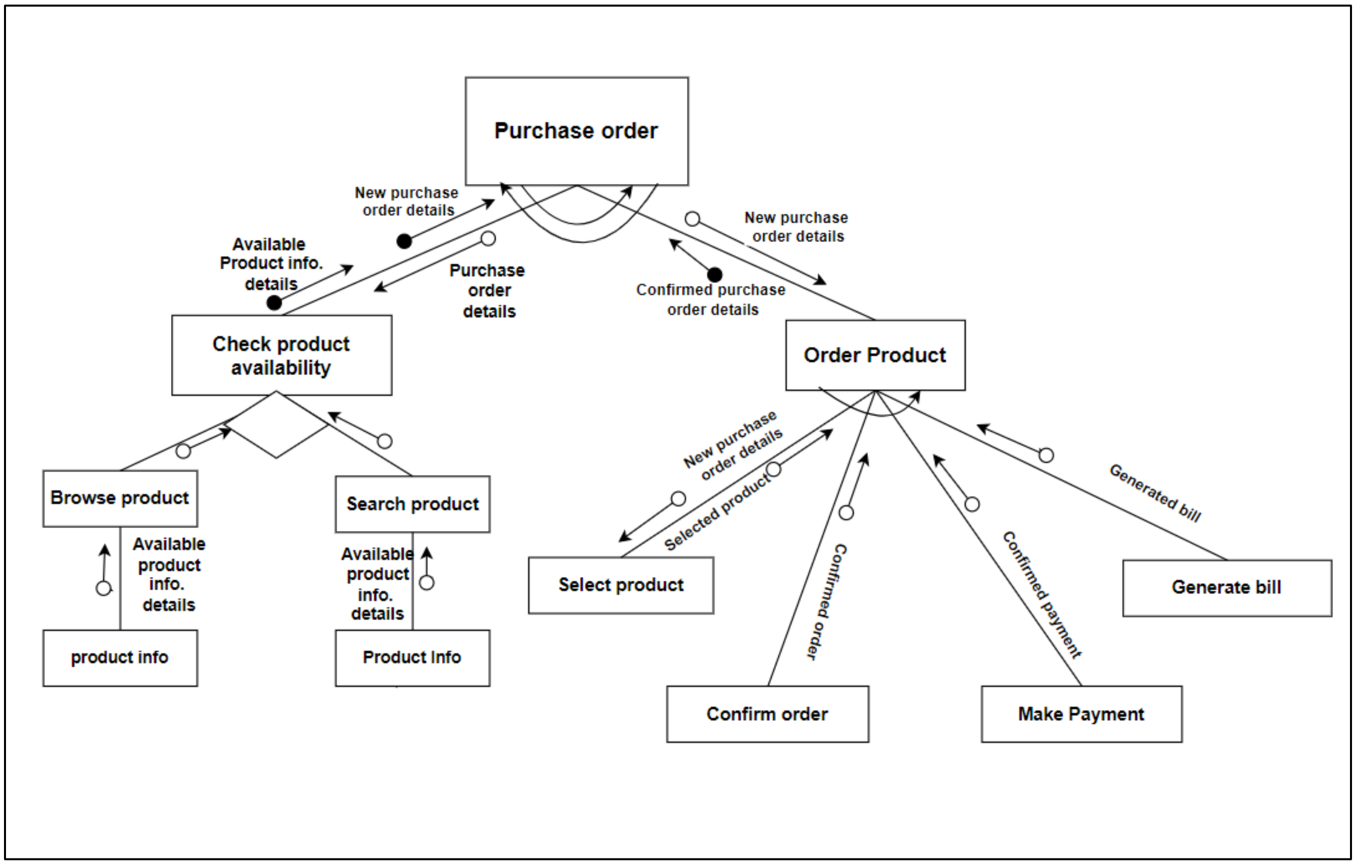
Purchase order
New purchase order details
Purchase order details
Available Product info. details
New purchase order details
Confirmed purchase order details
Check product availability
Order Product
Browse product
Search product
Available product info. details
Available product info. details
product info
Product Info
New purchase order details
Selected product
Confirmed order
Confirmed payment
Generated bill
Select product
Generate bill
Confirm order
Make Payment

Figure 19: Structure chart of purchase order

5.1.2.b. Module specification

NAME	**Purchase Order**
Purpose	To order product

Pseudocode	
	```
	BEGIN purchaseOrder
	  CALL Check product availability ()
	  CALL Order product ()
	END purchaseOrder

	PROCEDURE CheckProductAvailability
	  INPUT purchaseOrderDetails
	  RETRIVE productInfo FROM ProductInfoDatabase
	  IF ProductInfo IS available THEN
	      RETURN "Product Available"
	  ELSE
	      RETURN "Product Not Available"
	  END IF
	END PROCEDURE

	PROCEDURE SelectProduct
	  INPUT **“**PurchaseOrderDetails"
	  GET ProductInfo FROM ProductDatabase
	  OUTPUT "View and select product"
	  SET SelectedProductDetails = "Product selected
	based on user input"
	  RETURN SelectedProductDetails
	END PROCEDURE

	PROCEDURE ConfirmOrder
	    INPUT SelectedProductDetails, Supplier info.
	    CONFIRM Order WITH User
	    STORE Purchaseorder details IN
	OrderInfoDatabase
	    RETURN "Purchas Order Confirmed"
	END PROCEDURE

	PROCEDURE MakePayment
	    INPUT **“**ConfirmedPurchaseOrderDetails”
	    OUTPUT **“**Processing payment”
	    SET payment status = “ Payment confirmed”
	    RETURN  PaymentStatus
	END PROCEDURE
	```

	PROCEDURE GenerateBill
	   INPUT” ConfirmedPaymentDetails”
	   SET Invoice = "Generate invoice based on confirmed payment"
	   STORE Invoice IN InvoiceDatabase
	   OUTPUT: "Invoice generated"
	   RETURN Invoice
	END PROCEDURE

Input parameters	PurchaseOrderDetails, SelectdProductDetails, supplierInfo, confirmedPurchaseOrderDetails and ConfirmedPaymentDetails.
Output parameters	ProductInfo, selectedproductDetails, purchaseorderconfirmation, paymentstatus and Invoice.
Global variable	OrderInfo, ProductInfo and invoicedatabase
Local variable	productInfo, SelectedProductDetails, paymentstaus and invoice
Call	Check product availability, selectProduct, confirmorder, makepayment and generatebill.
Called by	Purchase order

Table 9: Table of module specification of purchase order

5.2. Report Preparation

5.2.1. Environmental model specification

Name: Kriti Lama

Section: L2C1

London met ID: 23047513

College ID: np01cp4a230013

The "Global Tech Corporation" has requested a report generating feature in the system that we are designing for them. This feature must provide admins the access and the necessary tools to generate a detailed report regarding any aspect of the business. To generate the sales report and purchase report, all the necessary information must be provided after the data provided has been validated, they will have access to the report. To prepare the reports, the system must be able to compile data about sales, purchases, profit/loss, quantity in inventory.

5.2.1.a. Context level diagram

The diagram below is the context level diagram for the process of report generation. It shows the processes that occur when a report is generated.

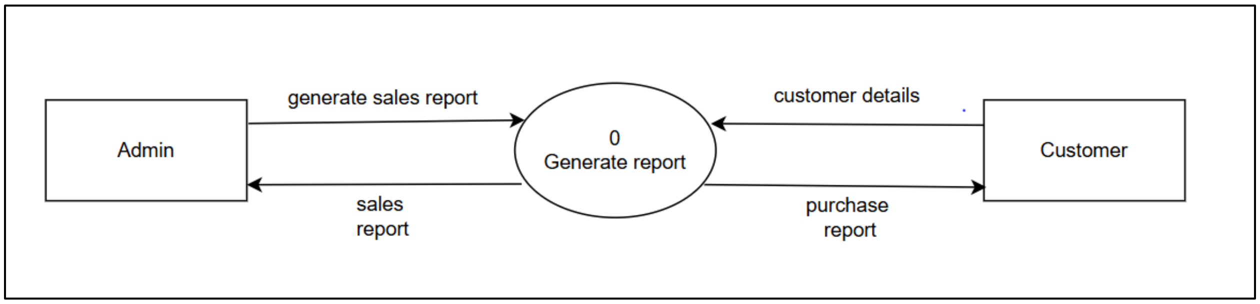

Figure 20: Context level diagram for report generation

5.2.1.b. Level 1 of Data Flow Diagram

The given diagram is the level 1 DFD diagram that describes the process of generating a report. Here, the admin enters the sales detail and it is stored in a database called sales info. The customer enters customer details which is stored in the database customer info. The information from the databases are forwarded to generate report process when suitable command is entered “generate sales report” or “generate purchase report”.

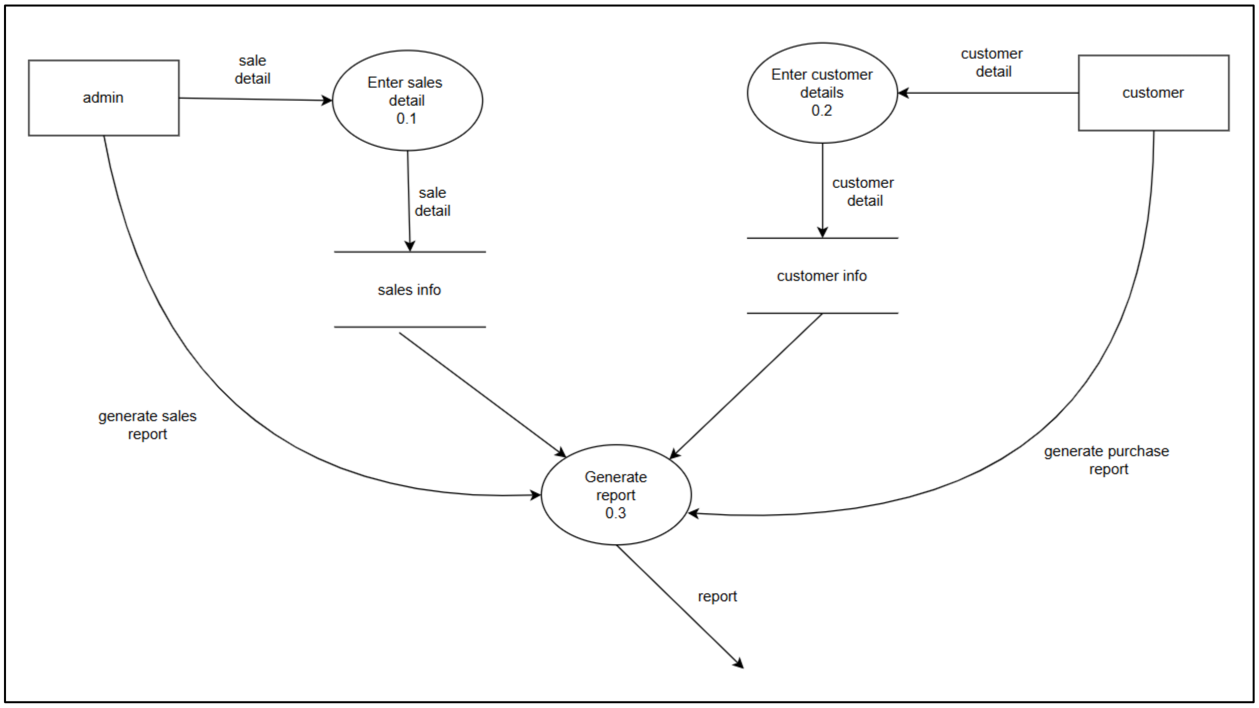

Figure 21: Level 1 diagram for report generation

5.2.1.c. Level 2 of Data Flow Diagram

Here in the level 2 diagram, level 1 has been expanded to show more detail. In order to generate purchase reports, first user ID is entered then the ID is verified. The supplier for the

customer is confirmed and date and time is specified. The format in which the user wants the report to be printed is selected and the report is generated.

Here in the level 2 diagram, level 1 has been expanded to show more detail. In order to generate sales report, the sales detail is extracted from sales info database. The total number of sales is confirmed and forwarded to report database. Then, valid date and time is entered from the saved sales database and profit or loss from that specific time frame is calculated and then forwarded to report database. The report details are forwarded and the type of report is selected by user then when the appropriate command is entered report is generated.

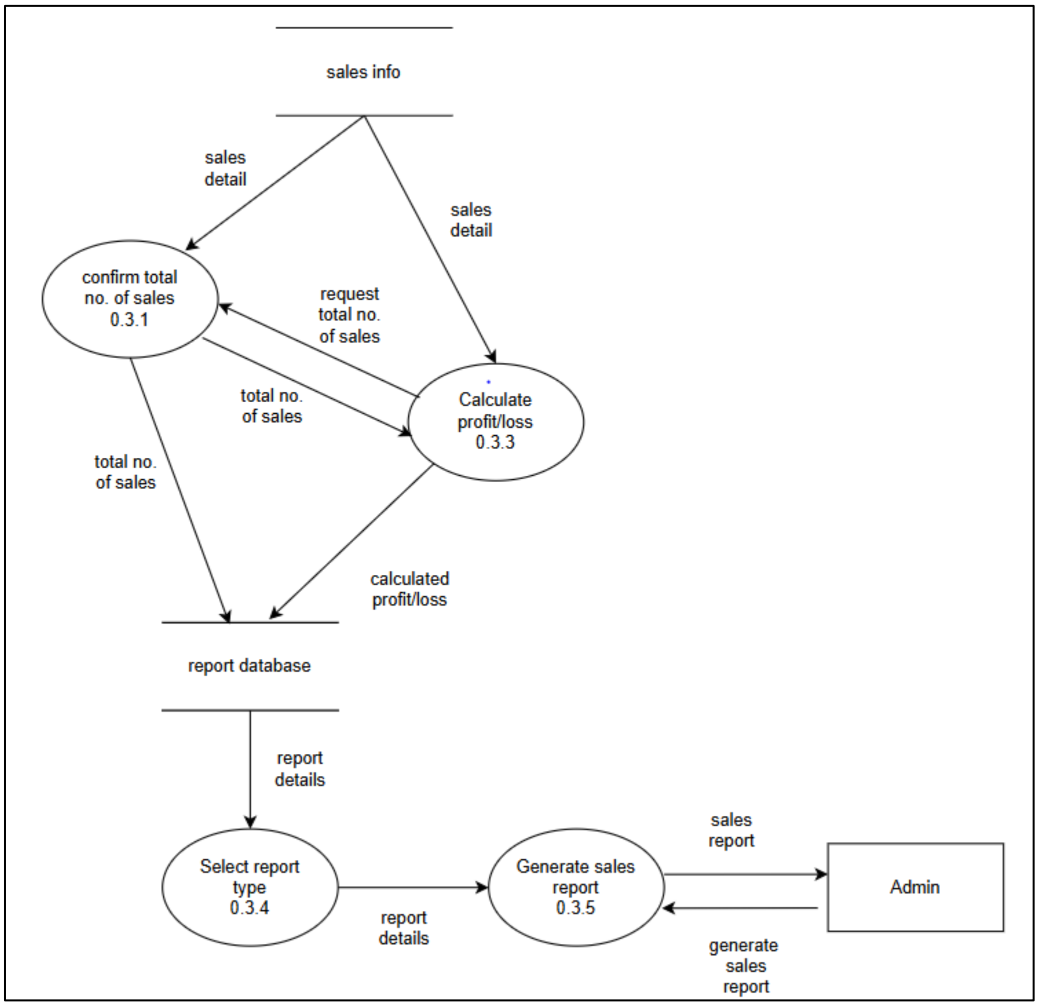

Figure 22 : Level 2 diagram for report generation

5.2.2. Design specification

5.2.2.a. Structure chart

The following structure chart indicated the fundamental modules that are triggered during report generation:

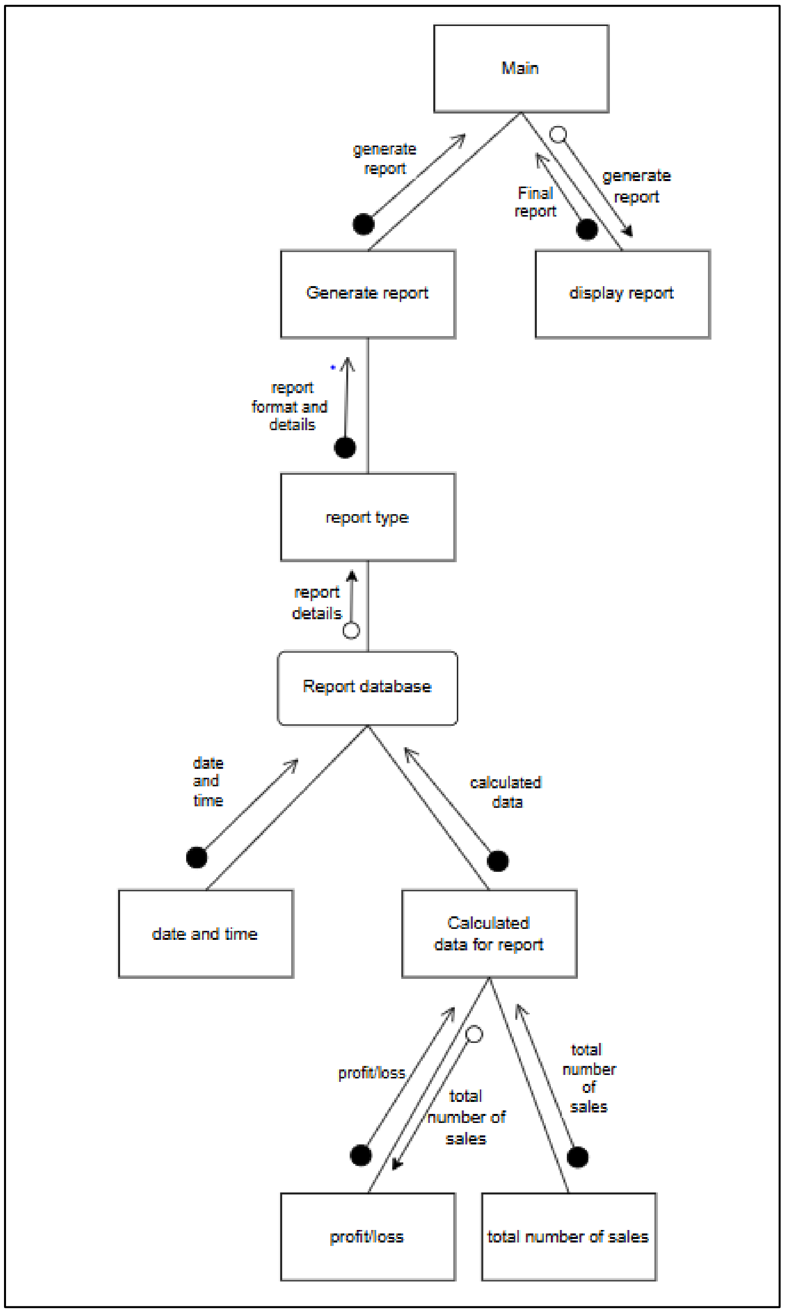

Figure 23 Structure chart of report generation

5.2.2.b. Module specification

Module name	Report generation
Purpose	To generate a report.
Pseudocode	**BEGIN** ReportGeneration **CALL** GeneratePurchaseReport() **CALL** GenerateSalesReport() **END** ReportGeneration **PROCEDURE** GeneratePurchaseReport **GET** PurchaseData **SET** reportData = [] **FOR EACH** purchase IN PurchaseData **GET** purchaseID, quantity, cost, supplier **ADD** {purchaseID, quantity, cost, supplier} **TO** reportData **END FOR** **RETURN** reportData **END PROCEDURE** **PROCEDURE** GenerateSalesReport **GET** SalesData **SET** reportData = [] **SET** totalSales = 0 **SET** totalProfit = 0 **FOR EACH** sale IN SalesData **GET** saleID, quantity, revenue, cost **SET** profit = revenue - cost

	ADD {saleID, quantity, revenue, profit} **TO** reportData
	SET totalSales = totalSales + revenue
	SET totalProfit = totalProfit + profit
	END FOR
	RETURN reportData, totalSales, totalProfit
	END PROCEDURE

Input parameters	PurchaseData, salesData
Output parameters	ReportData, totalSales, totalProfit
Global variable	NONE
Local variable	ReportData, purchaseID, quantity, cost, supplier, reportData, salesID, revenue, profit,total sales, totalProfit
Call	GeneratePurchaseReport(),GenerateSalesReport()
Called by	ReportGeneration

Table 10: Table of module specification of report generation

5.3. Real-Time Stock Updates

5.3.1 Environmental model specification

Name: Aastha Aryal

Section: L2C1

London met ID:23049023

College ID: np01cp4a230400

Stock management is a function of the inventory management system that checks the availability of stocks for purchase and updates them to a new number when they are developed.

5.3.1.a. Level 0 of Data Flow Diagram

In the context level diagram of real-time stock management, the system obtains purchase details from a purchase order procedure that is directly linked to the user. After entering the purchase details, the system displays the product's availability, and when the product is purchased, stocks are updated and the user is notified of the update.

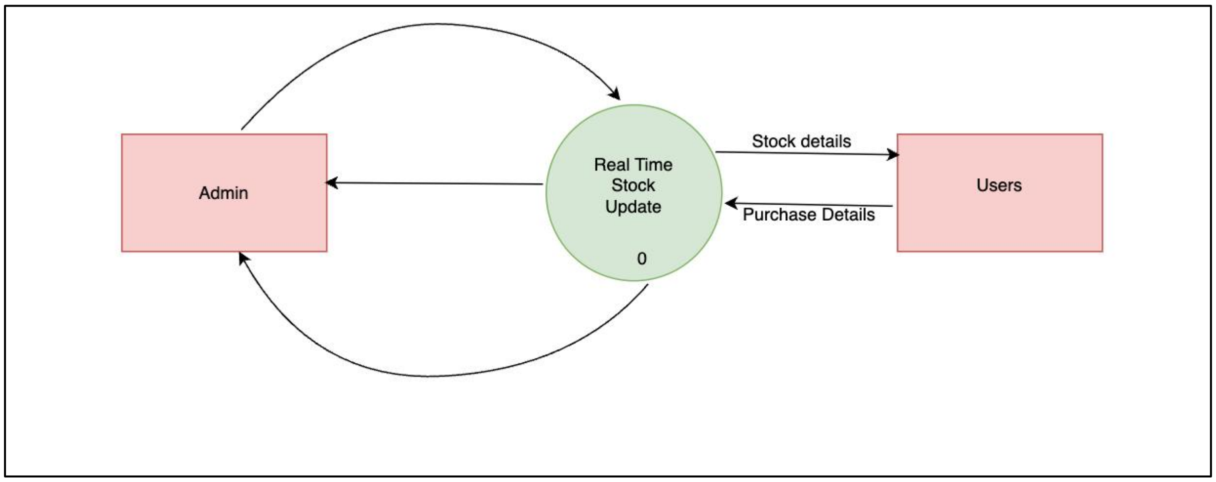

Figure 24 : Level 0 of Real-time stock updated

5.3.1.b. Level 1 of Data Flow Diagram

Level 1 DFD of the context level diagram in which the detailed procedure of real-time stock management is shown.

The product identification takes the purchase details and passes them on to the customer demands, which take the product name and give it to the product data store, which gives the report back to pass on and passes the product number to stock management, which gives the user the stock data, and when the product is purchased, the same process alert the admin.

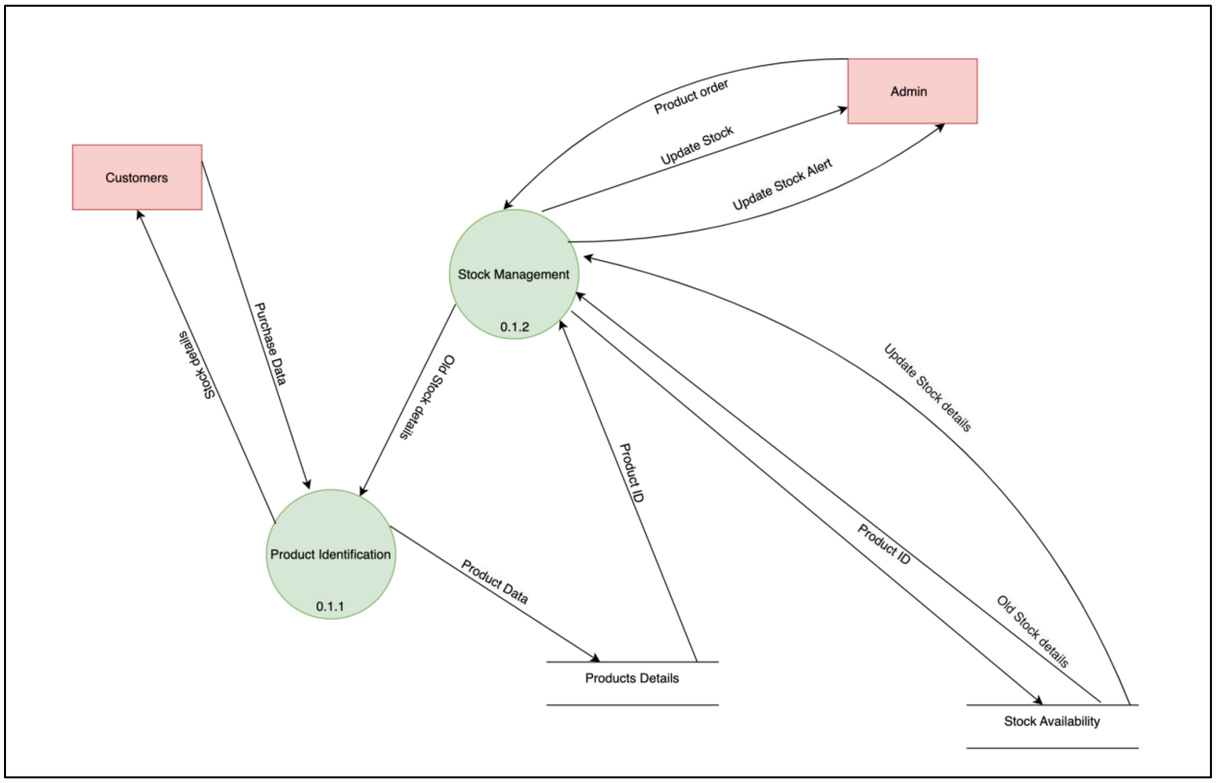

Figure 25: Level 1 of Real-time stock updates

5.3.1.c. Level 2 of Data Flow Diagram

The level 2 DFD breaks down the stock management process from the level 1 DFD. This process collects the product number for stock updates and forwards it to another process, which sends the product number and stock to the stock availability data store, which then returns the update confirmation to the process.

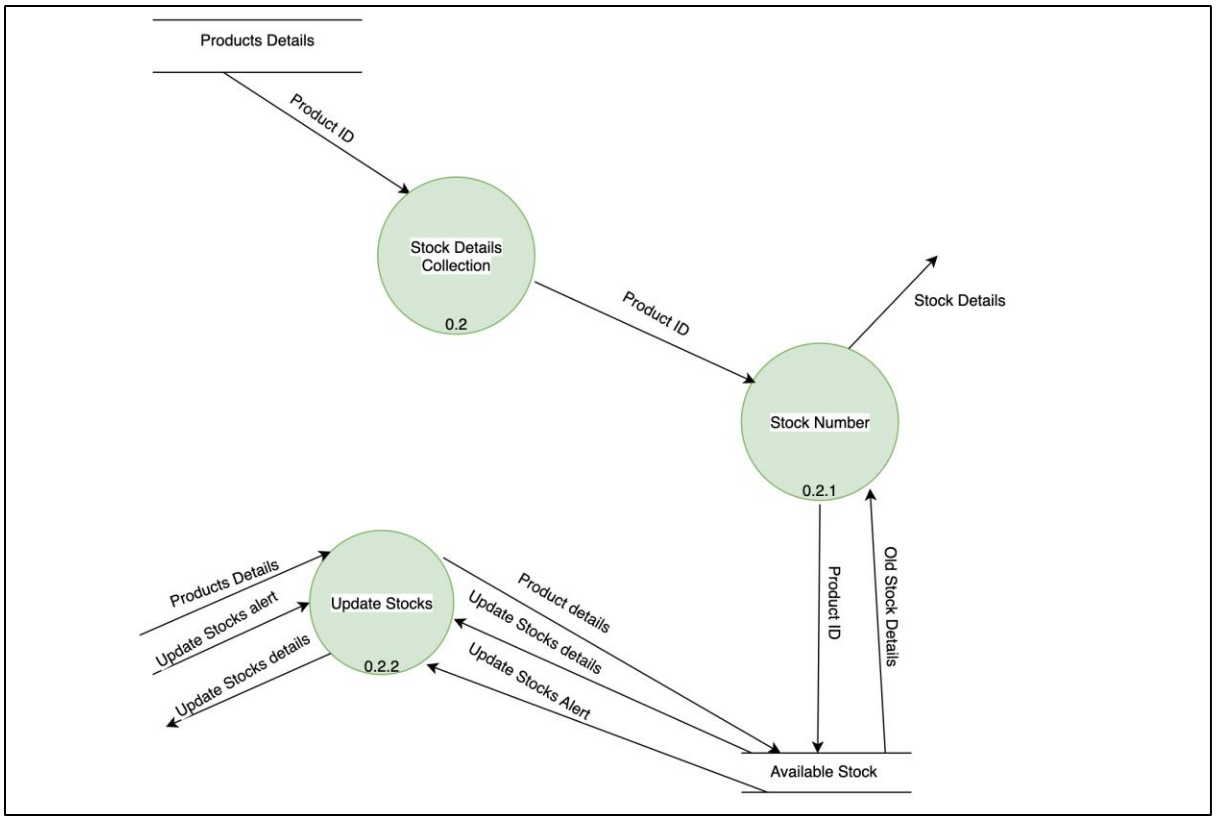

Figure 26: Level 2 of Real-time stock updates

5.3.2. Design specification

5.3.2.a. Structure chart

The structural chart for Real Time Stocks Update depicts the data flow of the process. The main function calls two functions: Product Identification and Stock Management. The first function (Product Identification) is called to show the user the availability and stock number of products, and the second function (Stocks Management) is called after the purchase to update the stock number and availability of the products.

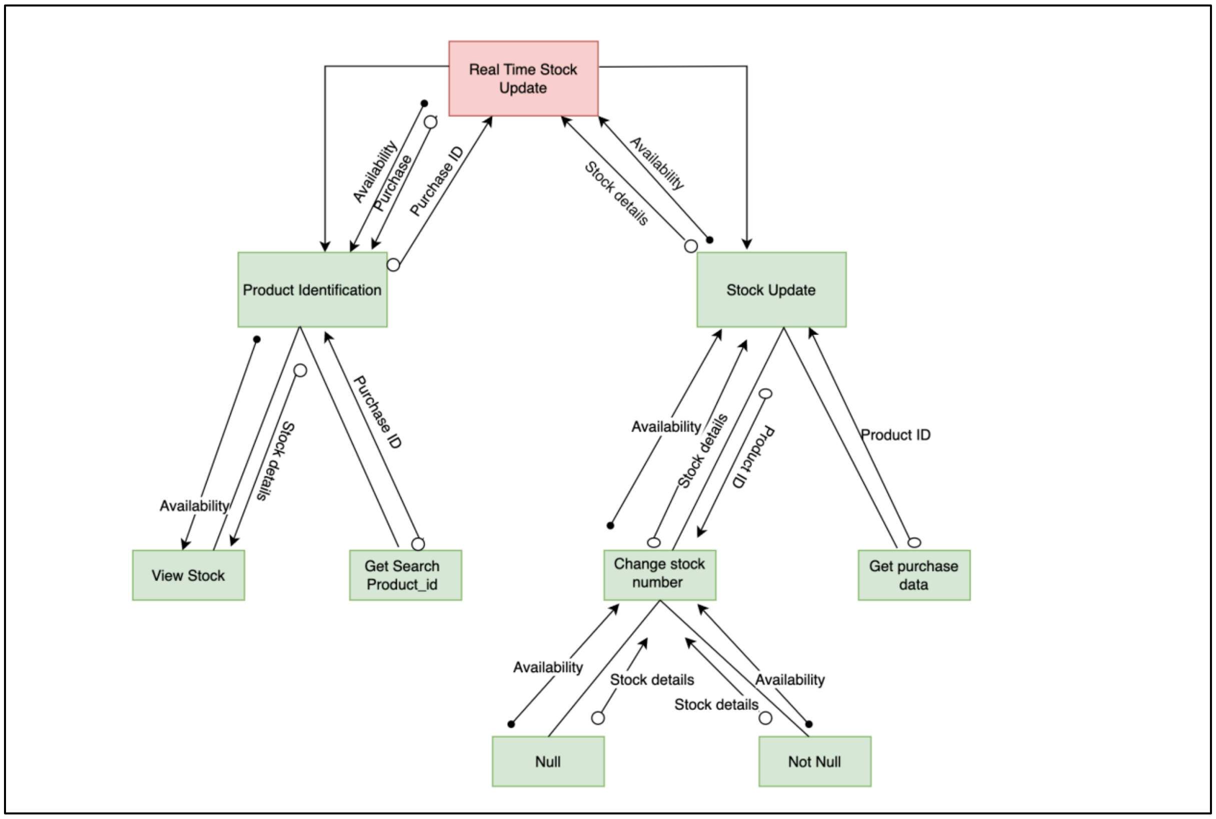

Figure 27: Structure chart of Real-time stock update

5.3.2.b. Module specification

Name	Real-Time Stock Updates
Purpose	To track and update inventory data in real time.

Pseudocode	BEGIN RealTimeStocksUpdate
	CALL ProductIdentification()
	CALL StocksManagement()
	END RealTimeStocksUpdate
	PROCEDURE ProductIdentification
	INPUT Product_ID
	SET productID = Product_ID
	GET Availability, Stocks_Data
	RETURN Availability, Stocks_Data
	END PROCEDURE
	PROCEDURE StocksManagement
	GET Purchased_Product_ID
	SET pP_ID = Purchased_Product_ID
	CALL ChangeStocksNumber()
	GET Availability, Stocks_Data
	RETURN Availability, Stocks_Data
	END PROCEDURE
	PROCEDURE ChangeStocksNumber
	SET Stocks_Data -= Purchased_Product_ID
	RETURN (IF Stocks_Data == 0 THEN "NA" ELSE "Available")

	END PROCEDURE
Input Parameters	Product Details Search , Purchased Product ID
Output Parameters	Availability, Stocks Numbers
Global Variable	Availability, Stocks Numbers
Local Variable	Purchased Product ID
Call	Purchased Record ,Stocks Management
Called By	Main(Real Time Stock Update)

Table 11: Module specification of Real-time stock update

5.4. Dispatch Order

5.4.1. Environmental model specification

Name: Rahul Jaiswal

Section: L2C1

London met ID: 22085575

College ID: np01cp4s230131

5.4.1.a. Level 0 of Data Flow Diagram

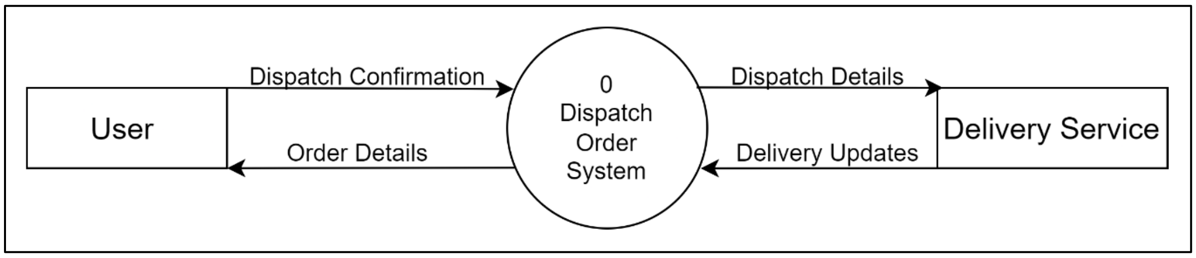

Figure 28: Context Diagram for Make Payment

A context diagram which is also known as a level 0 data-flow diagram, is used to describe and illustrate the boundaries of a software system. It identifies information flows between the system and external entities. The whole software system is depicted as a single process.

The context level diagram for the process of making dispatch order is illustrated in the above diagram. In this system User and Delivery Service are the external entity Here, User will enter a Dispatch Confirmation in the system and the system sends the users request to the Delivery Service and it will check whether the details are Updated or not and send dispatch details to the user. And the Order Details is sent to the user.

5.4.1.b. Level 1 of Data Flow Diagram

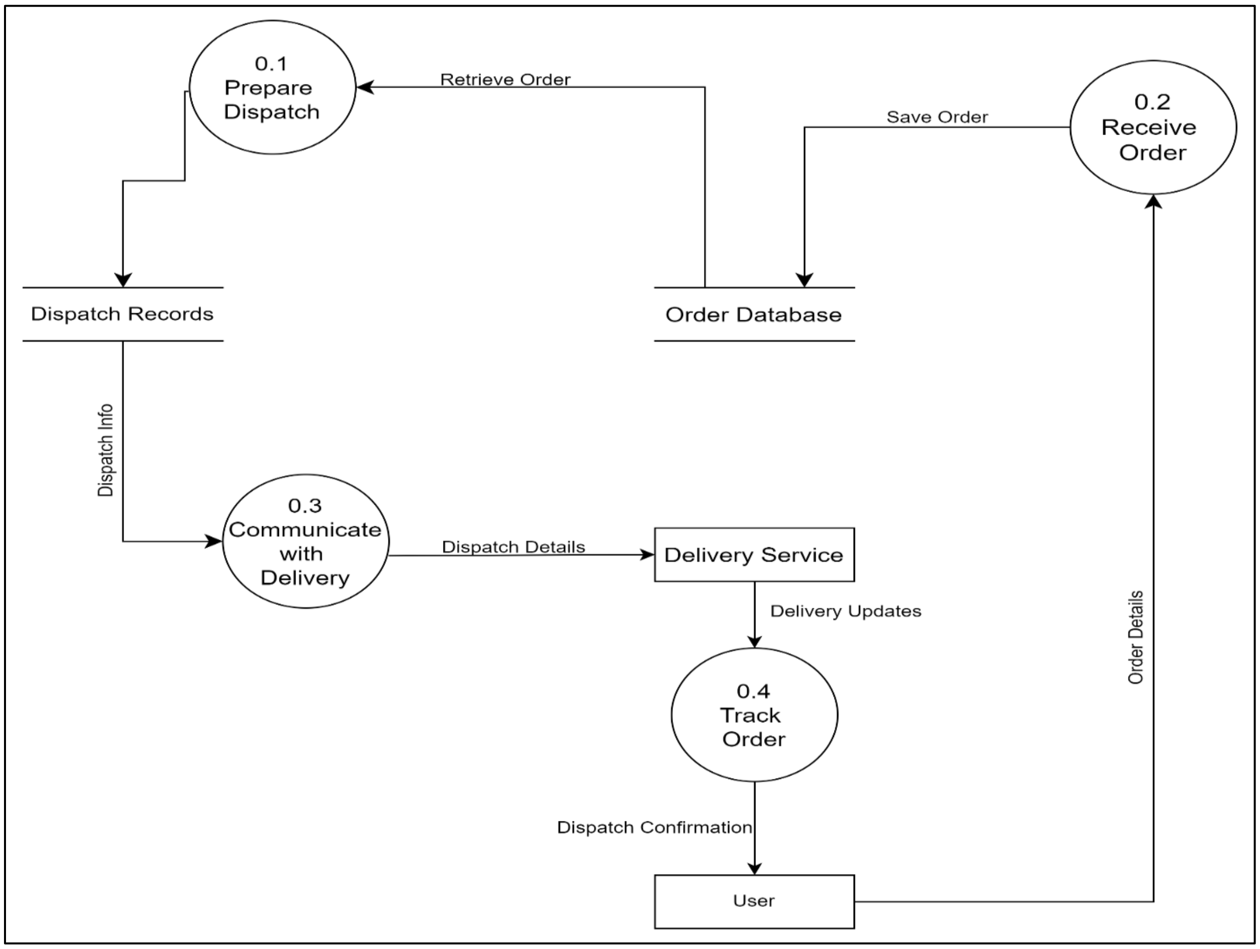

Figure 29: Level 2 DFD for Dispatch Order

In a 1-level DFD, the context diagram is divided into several bubbles/processes. At this level, we highlight the main functions of the system and break down the high-level process of the 0-level DFD into sub-processes. In a Level 1 DFD, one process node in the context diagram is divided into sub-processes.

In the diagram above, the "Dispatch Order" function is broken down into sub processes as Prepare Dispatch, Communicate with Delivery, Track Order and Receive Order. When customers place an order, their details are saved in the system. The system checks what's available in stock, prepares the items for shipping, and keeps a record of what's ready to go. The system uses the stored details to contact the delivery service and arrange shipping. As the delivery service provides updates, the system keeps track and informs the customer about the dispatch and delivery status.

5.4.1.c. Level 2 of Data Flow Diagram

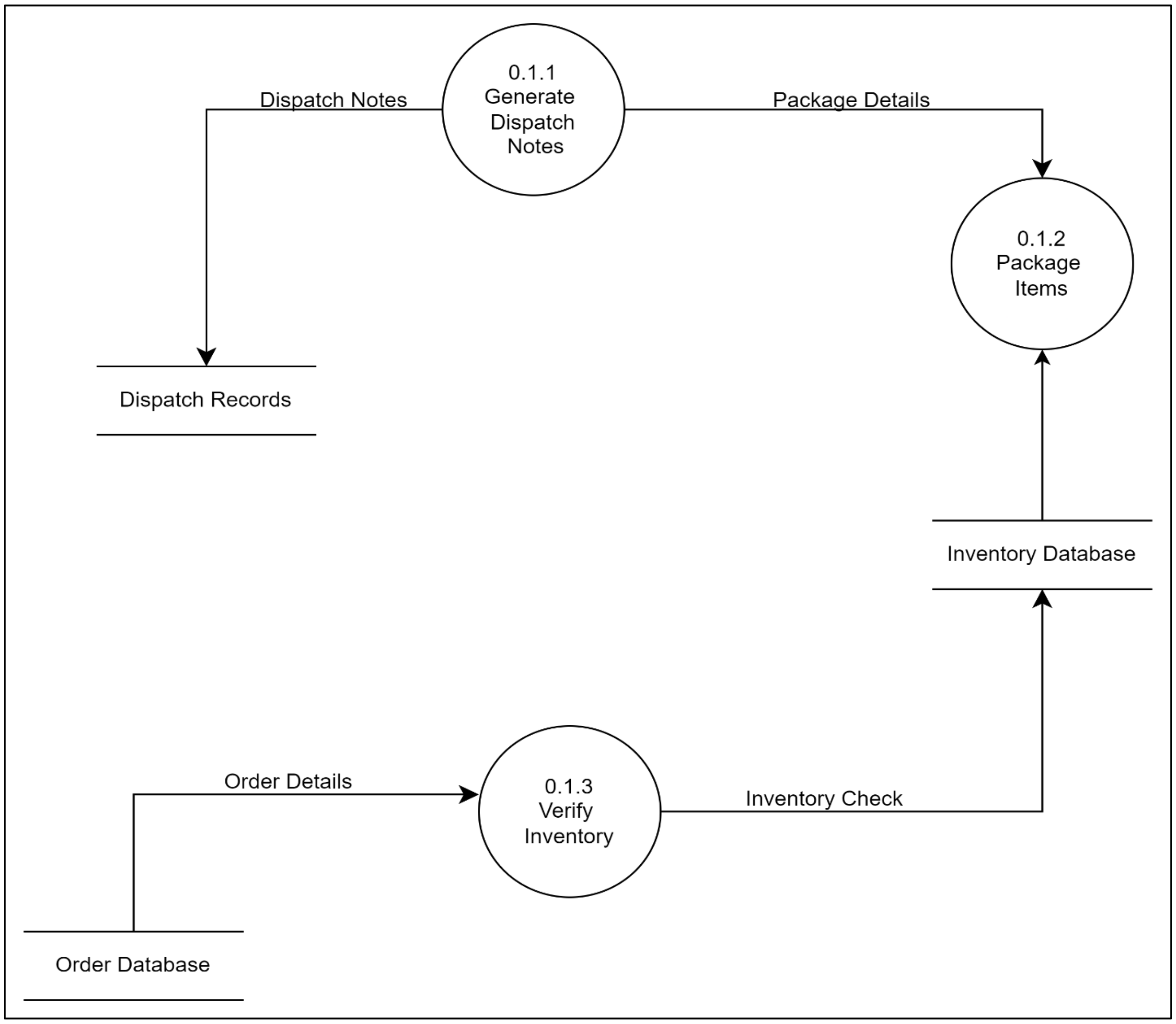

Figure 30: Level 2 DFD for Dispatch Order

Data is processed and modified as it goes through the various phases of a system, and a Level 2 Data Flow Diagram (DFD) illustrates this process graphically. The system's primary data flow and operations are identified using this high-level perspective of the system. A Level 2 DFD shows the data flow between the system's sub-processes and external entities. The system is divided into sub processes. The external entities stand in for data sources or destinations, while the sub-processes reflect how data is transformed or processed.

In the above diagram from the Receive Order of a level 1 is divided into the sub-processes. Here, there are three sub-processes which are Verify Inventory, Package Items, and Generate Dispatch Notes.

The system looks up the order details to make sure the items are in stock. Once the system confirms availability, it organizes and packs the items for shipping. After packaging, the system writes a document listing the order details, shipping instructions, and the customer's information, saving it for future reference.

5.4.2. Design Specification

5.4.2.a. Structure Chart

Structure Chart is a distinct aspect of logic programming that uses a set of relations to depict data structure and data flow. It shows the hierarchical relationship between the different parts of a system. It depicts the relationships among processes, functions, data store and the data flow and control flow are also represented.

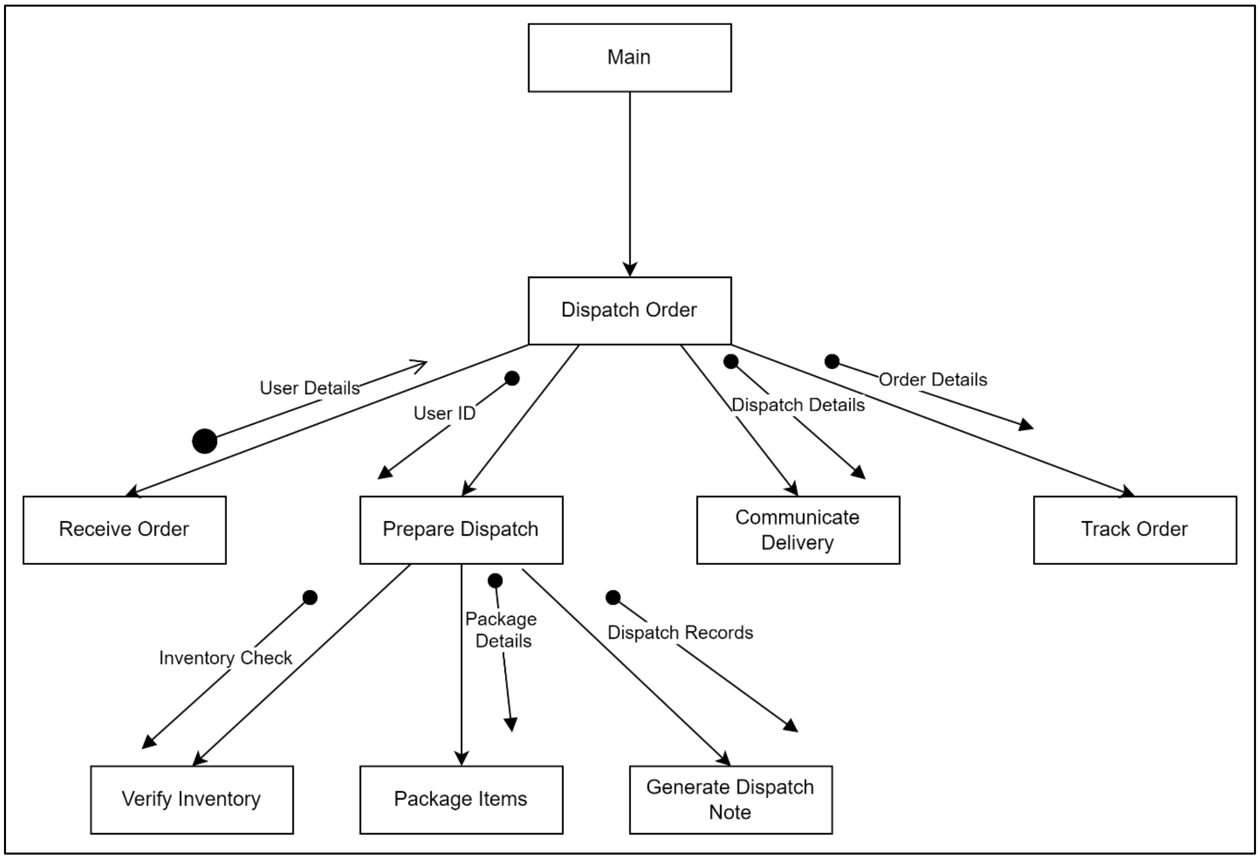

Figure 31: Structured Chart for Dispatch Order

The structure chart from the above provides a visual representation of the logical processes involved in dispatching an order. It illustrates the logical data flow and represents the system hierarchy, encompassing all the related processes. This diagram includes obtaining order details and dispatching orders.

5.4.2.b. Module Specification

Module Name	Dispatch Order
Purpose	This module obtains Dispatch details and add this data to prepare dispatch.
Pseudocode	**FUNCTION** DispatchOrder **CALL** ReceiveOrder **CALL** PrepareDispatch **CALL** CommunicateWithDelivery **CALL** TrackOrder **END FUNCTION** **FUNCTION** ReceiveOrder **INPUT** customerOrderDetails **STORE** customerOrderDetails **IN** OrderDatabase **RETURN** confirmation **TO** Customer **END FUNCTION** **FUNCTION** PrepareDispatch **CALL** VerifyInventory **CALL** PackageItems **CALL** GenerateDispatchNote **END FUNCTION** **FUNCTION** VerifyInventory **INPUT** orderDetails **FROM** OrderDatabase **RETRIEVE** stockDetails **FROM** InventoryDatabase **IF** stockDetails **IS** available

	THEN RETURN "Inventory Verified" **ELSE** **RETURN** "Out of Stock" **END IF** **END FUNCTION** **FUNCTION** PackageItems **INPUT** verifiedOrderDetails **PREPARE** packaging **FOR** verifiedOrderDetails **RETURN** "Packaging Completed" **END FUNCTION** **FUNCTION** GenerateDispatchNote **INPUT** packagedOrderDetails **CREATE** DispatchNote **WITH** orderDetails **AND** packagingInfo **STORE** DispatchNote **IN** DispatchRecords **RETURN** DispatchNote **END FUNCTION** **FUNCTION** CommunicateWithDelivery **INPUT** DispatchNote **SCHEDULE** delivery **WITH** DeliveryService **UPDATE** deliveryStatus **IN** DispatchRecords **RETURN** "Delivery Scheduled" **END FUNCTION**

	FUNCTION TrackOrder
	INPUT trackingRequest
	FROM Customer
	RETRIEVE deliveryStatus
	FROM DispatchRecords
	RETURN deliveryStatus
	TO Customer
	ELSE END
	END IF
	END DO

Input Parameters	orderid, dispatch Order
Output parameters	deliveryStatus
Global Variables	DB
Local Variables	Dispatch Order, Verify inventory, Package Details
Called By	Main

Table 12: Module Specification of Dispatch Order

5.5. Payment

5.5.1. Environmental model specification

Name: Krishal Acharya

Section: L2C1

London met ID: 23049010

College ID: np01cp4a230381

5.5.1.a. Level 0 of Data Flow Diagram (Context diagram)

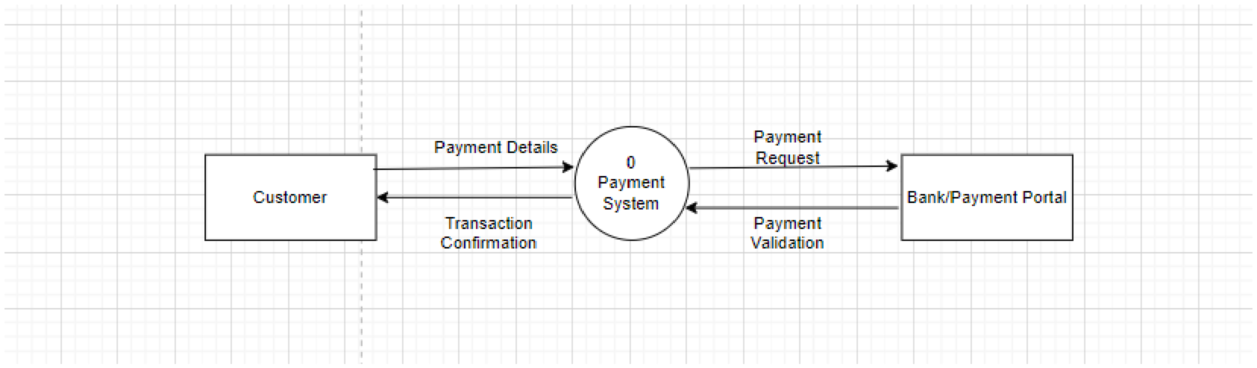

Figure 32 Level 0 DFD of DFD

A level 0 Data Flow Diagram (DFD) is also known as a Context Diagram. Level 0 DFD is used to illustrate the processes of software system. Here, a rough idea about the entities and their relationships are shown. The context diagram for payment process is given above.

In the Context Level diagram, Customer and Bank/Payment Portal are external entities. Here a customer sends a payment details, which then goes to payment system and if the customer payment matches with payment required it then sends payment request to Bank/Payment Portal. And if the customer has balance then, the Bank/Payment Portal sends a payment validation to payment system. Then the transaction is done and then payment confirmation is sent to customer to provide evidence of the successful transaction.

5.5.1.b. Level 1 of Data Flow Diagram

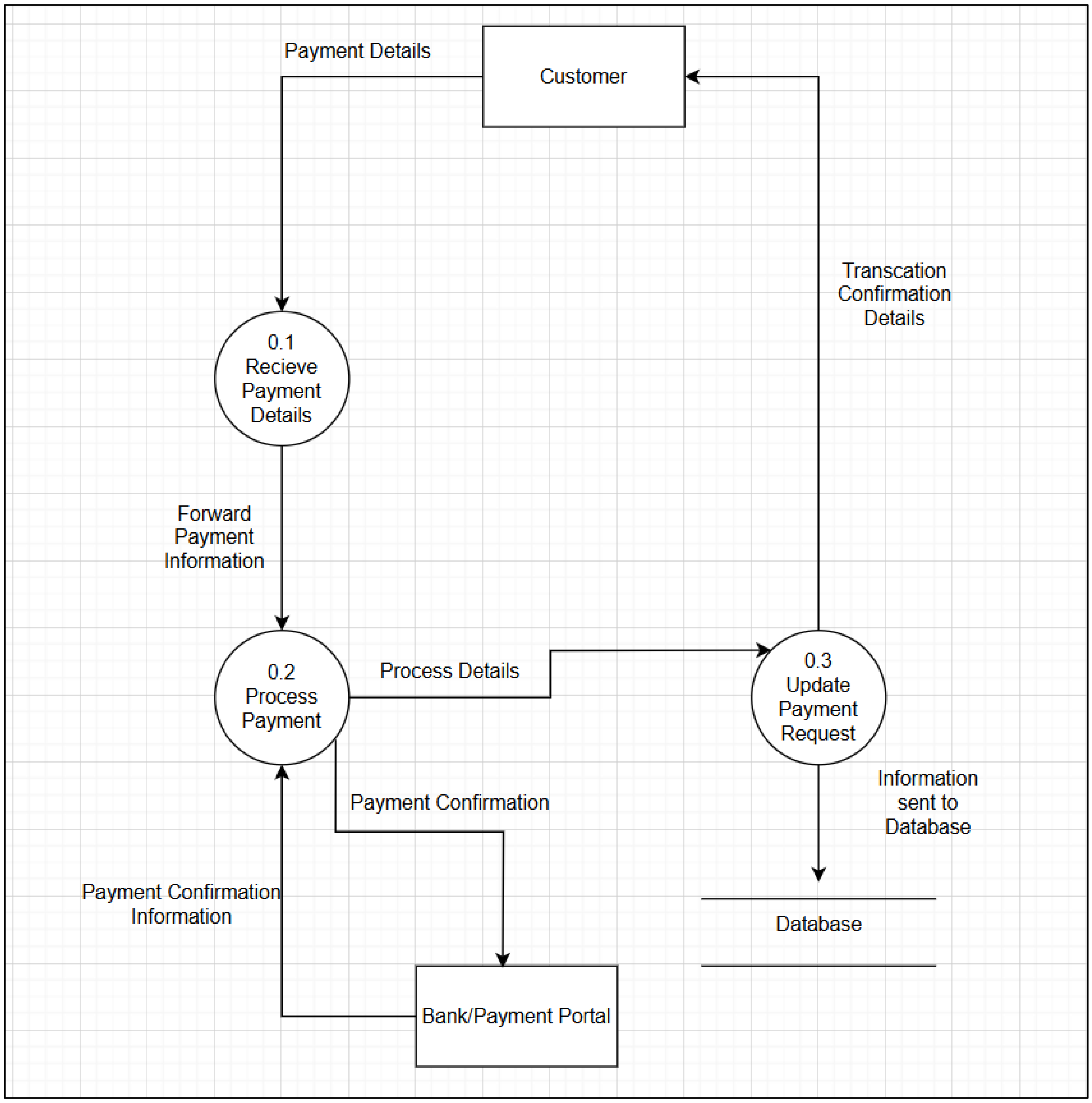
Payment Details
Customer
Transcation Confirmation Details
0.1 Recieve Payment Details
Forward Payment Information
0.2 Process Payment
Process Details
0.3 Update Payment Request
Payment Confirmation
Information sent to Database
Payment Confirmation Information
Database
Bank/Payment Portal

Figure 33 Level 1 Data Flow Diagram of payment

A Level 1 Data Flow Diagram is a context diagram where information is divided into many bubbles and processes. In this Level 1 diagram, we break down the Level 0 Data Flow Diagram and highlight it into many sub processes. Here one or many process are divided in to sub-processes.

In the mentioned diagram, the payment system is divided into many sub processes. Specifically, it is divided into three sub processes. Receive Payment Details, Process Payment and Update Payment Request. Then, the data is stored in database. After receiving all the payment details and sending request to bank and then bank/payment portal confirming the request and sending the data to update in payment request part. The transaction confirmation is sent and they can complete the transaction.

5.5.1.c. Level 2 of Data Flow Diagram

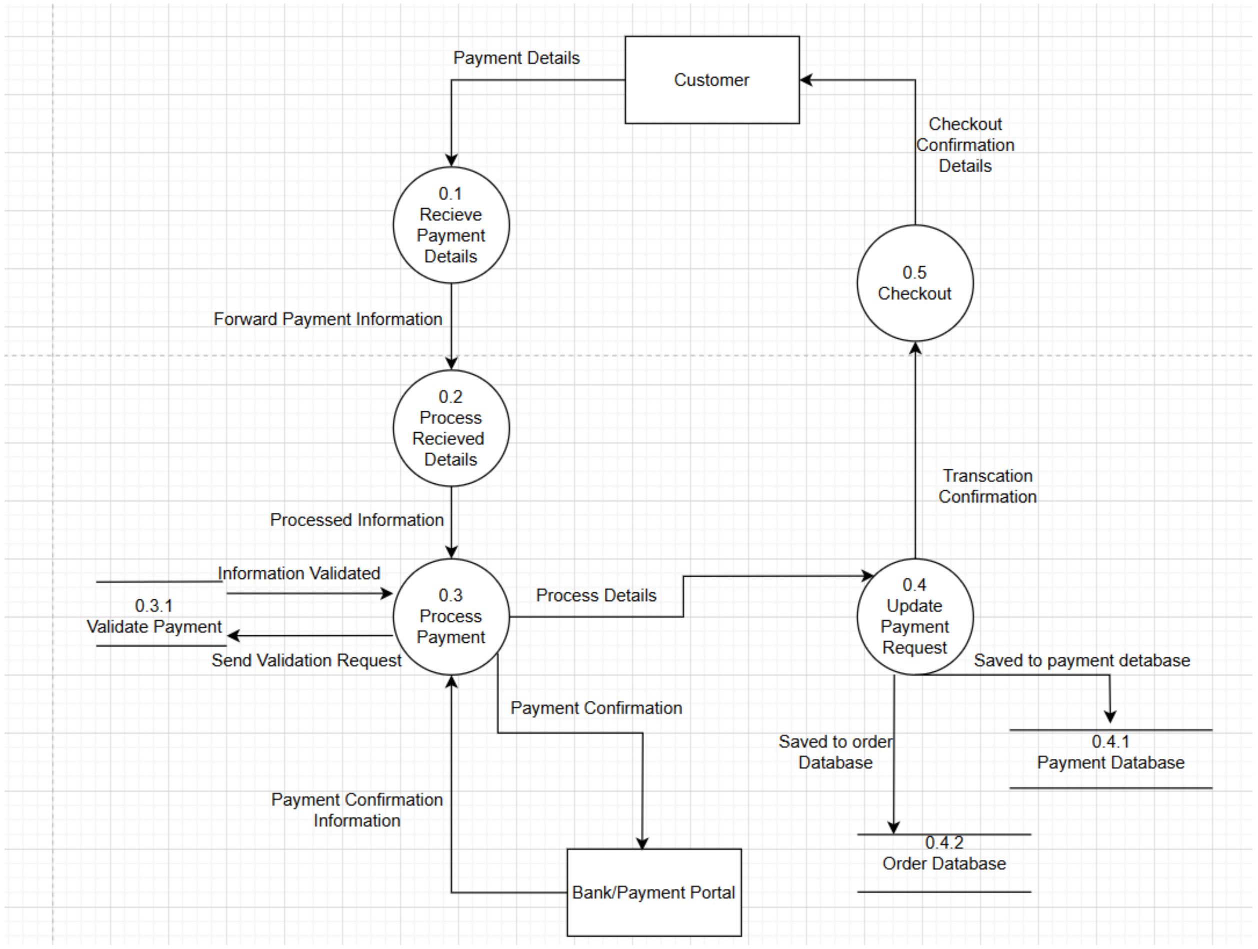

Figure 34: Level 2 of payment

The data is modified, checked processed as it is handled along the various steps of the system thus, in Level 2 Data Flow Diagram it shows data flow between the sub processes and external entities. Here, the sub process shows us how data are managed and handled and external entities stand for data destinations and sources.

In the above figure, the make payment system is broken down into many sub processes. I.e. Receive payment details, Process Receive Details, Process payment and Update payment records. Here, the customer sends payment details and payment system receives the payment detail. Then, the received payment details is forwarded to process received details. Once the information is then checked it is next send to process payment. Then, a payment confirmation information is sent to Bank/Payment portal. After that payment request is sent to Bank/Payment Portal it then sends payment confirmation to process payment. Here the process detail is then sent to update payment records. Which then sends date at three parts. Firstly, it sends payment information to Payment database. Secondly it sends a request to update order status to order database. Lastly, it sends a request for transaction confirmation called checkout to then Customer.

5.5.2. Design Specification

5.5.2.a. Structure chart

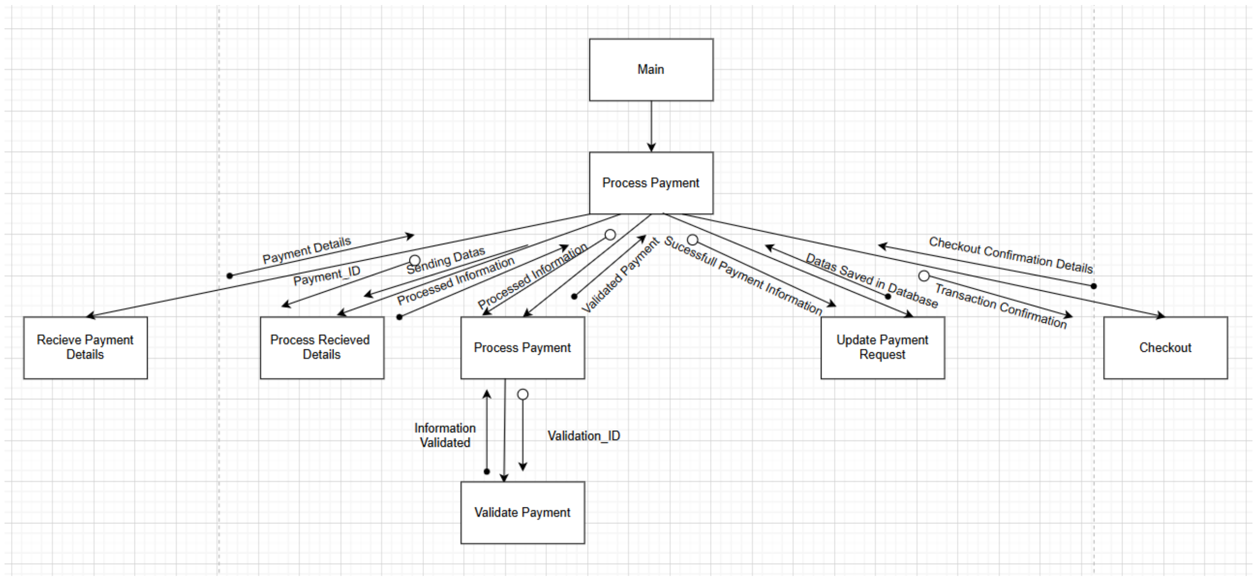

Figure 35: Structure chart of payment

Structure Chart is a top-down modular design which shows the different modules in a system. Structured chart includes different squares with different values and all of them have different lines values. Structure Chart depicts relationship among processes, functions, data store and the data flow and control flow are also shown.

In the above structured chart, we can see that there are different functions and properties. Payment process has 5 sub processes. The above structure chart gives visual representation of logical sub processes required in making a payment in real life scenario. Here, first of all, payment_id are received and sent back with payment details. And then, raw data are sent to be processed in process received details. Then the processed information is sent back to process payment preparing it for the next step. After that, payment is done with bank or payment portal as shown in level 2 data flow diagram. Firstly, processed information is sent to process payment then, validation_id is sent to be validated. After validation, information of validation is sent to bank/payment portal and then is forwarded back to process payment as validated payment. Then successful payment information is next sent to update payment request and saved information prompt is sent back to process payment and last the transaction confirmation request is send to checkout where the user can confirm to proceed the transaction or not then the checkout confirmation details is sent to the user/customer depending on the scenario.

Imagine you're ordering pizza online. You start by choosing your toppings and size as Customer. Then, you enter your payment details as Payment Details. Then, the restaurant checks if your card is valid or not as Validate Payment. If it is, they process the payment as Process Payment and update their system as Update Payment. Finally, they deliver your pizza as Checkout and provide checkout details along with it.

5.5.2.b. Module specification

Module Name	Payment
Purpose	This module obtains payment details from the customer and then processes this data and completes the transaction

Pseudocode	DO
	userId = database.getUserId()
	paymentDetails = database.getPaymentDetails()
	paymentMethod = database.getSelectedPaymentMethod()
	validationResponse = ValidatePaymentDetails(userId, paymentDetails, paymentMethod)
	IF
	validationResponse.status == "failure":
	StoreFailureDetails(validationResponse.reason)
	ELSE
	checkoutDetails = GenerateCheckout(validationResponse.validPaymentDetails)
	StoreTransactionDetails(userId, checkoutDetails)
	UpdateDatabase(userId, checkoutDetails)
	DisplayCheckout(checkoutDetails)
	END IF
	END DO

Input Parameters	customer_id, paymentDetails
Output Parameters	DisplayCheckout
Global Variables	Database
Local Variables	validationResponse, checkoutDetails, StoreTransactionDetails, UpdateDatabase

Calls	database.getUserId()
	
	database.getPaymentDetails ()
	
	ValidatePaymentDetails(userId, paymentDetails, paymentMethod)
	
	StoreTransactionDetails(userId, checkoutDetails)
	
	DisplayCheckout(checkoutDetails)

Called by	Main

Table 13: Module specification of payment

6. Summary (Conclusion)

The development of Global Tech Corporation's Inventory Management System (IMS) project is to address problems from previous system failure, including delays, financial damage, and complaints from clients. The purpose is to develop a more effective management system for managing their warehouse operations in Nepal.

Key features like restricting user access (Admins and Buyers), storing record of purchases, updating inventory in real-time, managing purchases and products, providing secure payment methods, and generating reports to help in decision-making have all been incorporated into this new system. The inventory is going to become highly efficient and well-organized with all of the new features.

We were required to focus on one component of the system, such as purchase orders, report generation, payments, and document thorough documentation, DFD charts, ERDs and diagrams that demonstrate the way it functions. To arrange everything carefully and precisely, we used systematic methods like procedure descriptions and data flow diagrams (DFDs).

Through cooperation and keeping track of our progress, this project taught us problem-solving, teamwork, and how to design an efficient system. The IMS is a useful tool that we can use in real-world scenarios since it will help the business manage time efficiently, reduce losses, calculates total sales, profit and loss, improve customer satisfaction and provides users with a secure virtual environment.

References

geeksforgeeks, 2024. *geeksforgeeks.* [Online]
Available at: https://www.geeksforgeeks.org/levels-in-data-flow-diagrams-dfd/
[Accessed 14 December 2024].

geeksforgeeks, 2024. *geeksforgeeks.* [Online]
Available at: https://www.geeksforgeeks.org/software-engineering-structure-charts/
[Accessed 22 December 2024].

Korostashovets, A., 2024. *Worksection.* [Online]
Available at: https://worksection.com/en/blog/project-charter.html
[Accessed 12 December 2024].

Scott Robinson, T. N., 2024. *TechTarget.* [Online]
Available at: https://www.techtarget.com/searchdatamanagement/definition/data-flow-diagram-DFD#:~:text=A%20data%20flow%20diagram%20(DFD)%20is%20a%20graphical%20or%20visual,and%20Design%20Method%20(SSADM).
[Accessed 13 December 2024].

smartdraw, 2024. *smartdraw.* [Online]
Available at: https://www.smartdraw.com/entity-relationship-diagram/
[Accessed 13 December 2024].