SRS PROPOSAL

Content

INTRODUCTION : -

The mother company of the First Steps Babywear Lanka is situated in India.The Indian company decided to establish First Steps in Sri Lanka because of the GSP and from this many job opportunities have been created. All the transactions in this company are done in dollars,therefore it is a main advantage for the economy of Sri Lanka.During the coronavirus, many industries in Sri Lanka fell due to the coronavirus. But it did not affect the First Steps Baby Wear company at all. Because the mother company of it is in India, they sponsor the first steps at any time and took all the necessary solutions for it, so the jobs of all employees who are working in the Panadura branch and the other branches(about 4000) were not stopped, the company always tried to protect the employees.

Now with the current situation in our country, all the industries have fallen into a very low condition because the tax has increased a lot, so we have to spend a lot of money mainly on electricity, water and everything else. However our company is BUI approved therefore we don't need to pay tax, India has given this opportunity to our country,this is an enormous advantage.

First Steps Baby Wear company which is in Panadura is the head office and there are 9 branches all around Sri Lanka.For their products fabric is the foremost thing,they import fabric from their own fabric mill owned by the Indian company,therefore it is also a dominant advantage.The process in this company is briefly explained below.In addition to the fabric there are some raw materials that company have to buy from Sri Lanka which are approved by the buyers who are in UK and US. At the head office cutting,printing and embroidering sections are covered and only the clothes are sewn in the other branches. After the clothes are sewn up in the 9 branches,they are double checked and then exported to the UK and US.

The goals of our company are,currently our buyers are only in the UK and Us,they have a target to increase buyers in the future and in the printing section in the company there are both manual and auto printing,however the manual printing has some weaknesses,so they hope to improve it too.

Another problem they have is that when they import fabric and take raw materials from Sri Lanka and transport them to other branches,they should have to transport raw materials before other companies are opened to continue the process. But there are weaknesses in the company's transport system and raw materials are not at the companies on time and the employees also have to wait.We have decided to make our software to control this transport system. In this, they have used GPS, but due to the reasons of the drivers or the reasons of the time schedule, this has changed a lot and the time period has increased a lot, so it has become a problem to keep the process flowing because the transport system is not properly controlled.

1.1 Background of the Project

The decision to implement a software solution for controlling the transport system at First Steps Babywear Lanka stems from a crucial need to address operational challenges and enhance efficiency within the company's supply chain. The existing transport system, relying on GPS technology, has encountered issues related to drivers and scheduling, resulting in delays and disruptions in the production process.

1.2 Transport System Structure

The transport system at First Steps Babywear Lanka is a critical component of the company's supply chain, responsible for the timely and efficient movement of raw materials and finished products between the fabric mill, the head office in Panadura, and the nine branches across Sri Lanka. The structure of the transport system is designed to support the entire production process, ensuring a seamless flow from material procurement to product delivery. The following outlines the key elements and components of the transport system:

1. **Source and Procurement:**

    - **Fabric Mill in India:** The primary source of fabric for First Steps Babywear is the company's fabric mill in India, owned by the Indian mother company. Fabric is imported from this mill to meet the production requirements of the Panadura head office and the other branches.

03.Diagrams

A diagram, in the context of software engineering, is a graphical representation of a system, process, or concept. Diagrams are used to visualize and communicate complex ideas, structures, and relationships in a way that is easier to understand than written text alone. They can help software engineers, developers, and stakeholders to understand, design, and communicate about software systems more effectively.

A diagram is usually a two-dimensional display which communicates using visual relationships. It is a simplified and structured visual representation of concepts, ideas, constructions, relations, statistical data, anatomy etc. It may be used for all aspects of human activities to explain or illustrate a topic.

1.Visualisation: Diagrams provide a visual representation of a system, process, or concept, making it easier for stakeholders to grasp complex ideas. By using shapes, symbols, and connections, diagrams can represent intricate relationships and structures in a clear and concise manner.

2.Communication: Diagrams serve as a common language that can be easily understood by all stakeholders, including software engineers, developers, project managers, and clients. They facilitate effective communication by providing a shared visual reference that helps to align everyone's understanding of the system.

3.Analysis: Diagrams can be used to analyze various aspects of a software system, such as its architecture, design, and behavior. For example, a class diagram can help to identify classes and their relationships, while a sequence diagram can help to visualize the flow of messages between objects.

4.Design: Diagrams play a crucial role in the design phase of software development. They help to conceptualize and plan the structure of the system, including its components, interfaces, and interactions. By visualizing the design, engineers can identify potential issues and make informed decisions about the system's architecture.

5.Documentation: Diagrams serve as a form of documentation that captures the key aspects of a software system. They provide a visual record of the system's structure, behavior, and requirements, which can be referenced by developers, testers, and other stakeholders throughout the software development lifecycle.

6.Problem-Solving: Diagrams can be used as a problem-solving tool to analyze and solve complex issues in a software system. For example, a sequence diagram can help to identify bottlenecks or inefficiencies in the system's message flow, while a state diagram can help to identify potential states and transitions in the system's behavior.

There are many types of diagrams .

- ER Diagram
- Use Case Diagram
- Activity Diagram
- DFD Diagram

ER Diagram:

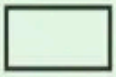
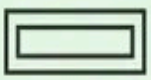
An Entity-Relationship (ER) diagram is a type of diagram used in software engineering to visually represent the data model of a system. It is a graphical representation of entities (objects or concepts) and the

relationships between them. ER diagrams are commonly used in database design to model the structure of a database and the relationships between its entities.

The Entity Relational Model is a model for identifying entities to be represented in the database and representation of how those entities are related. The ER data model specifies an enterprise schema that represents the overall logical structure of a database graphically.

The Entity Relationship Diagram explains the relationship among the entities present in the database. ER models are used to model real-world objects like a person, a car, or a company and the relation between these real-world objects. In short, the ER Diagram is the structural format of the database.

ER Model is used to model the logical view of the system from a data perspective which consists of these symbols:

Figures	Symbols	Represents
Rectangle		Entities in ER Model
Ellipse		Attributes in ER Model
Diamond		Relationships among Entities
Line		Attributes to Entities and Entity Sets with Other Relationship Types
Double Ellipse		Multi-Valued Attributes
Double Rectangle		Weak Entity

1. Entity: An entity represents a real-world object or concept, such as a person, place, thing, or event. In an ER diagram, entities are depicted as rectangles. Each entity has a name that describes the type of object it represents.

2.Attribute: An attribute is a property or characteristic of an entity. Each attribute has a name and a data type (e.g., string, integer, date). In an ER diagram, attributes are depicted as ovals connected to their respective entities. For example, if an entity represents a "Person," it might have attributes like "Name," "Age," and "Date of Birth."

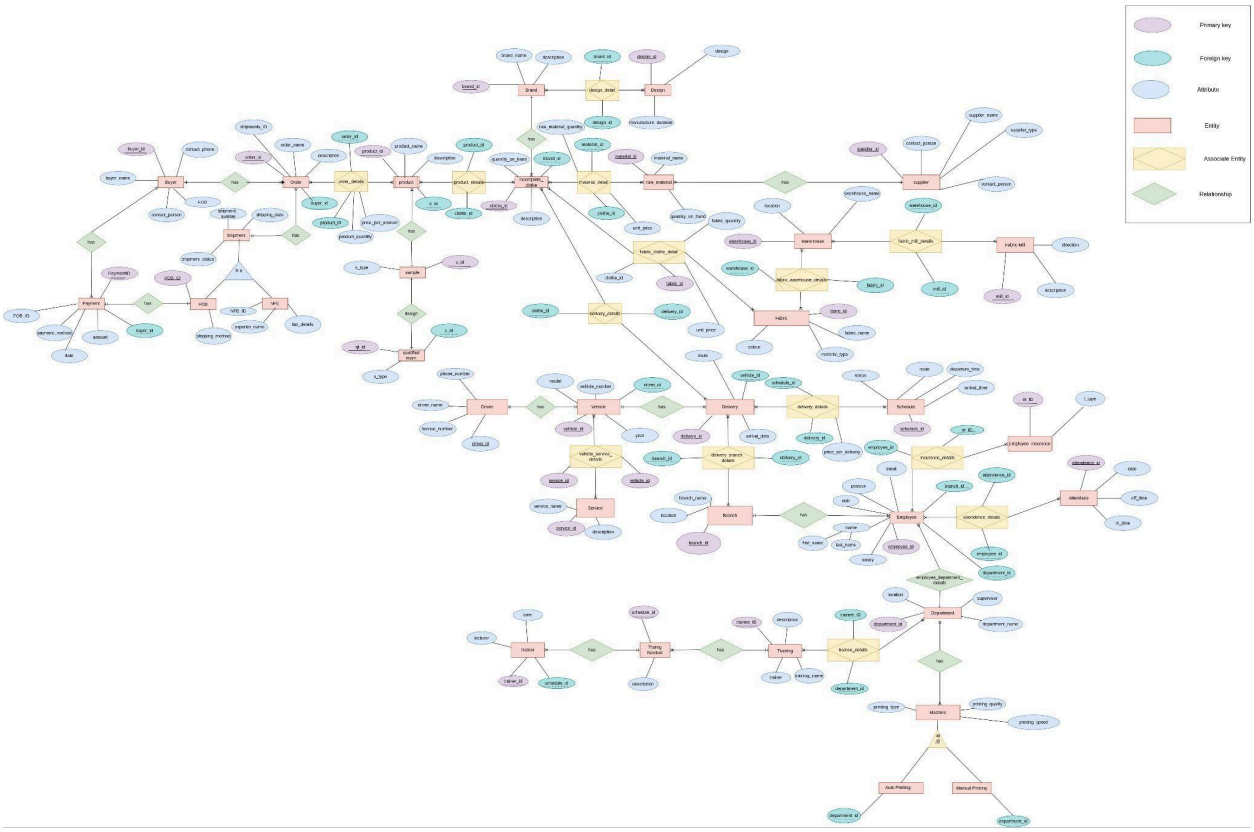
3.Relationship: A relationship represents the association between two or more entities. It describes how entities are related to each other. In an ER diagram, relationships are depicted as lines connecting the related entities. Each relationship has a name that describes the nature of the association. For example, if there is a relationship between a "Person" and a "Company," it might be called "Works for."

4.Cardinality: Cardinality specifies the number of instances of one entity that can be associated with the number of instances of another entity. It is represented using symbols such as "1" (one), "M" (many), and "0" (zero). For example, if a "Person" can work for "Many" companies, the cardinality of the "Works for" relationship would be "M."

5.Key: A key is an attribute or combination of attributes that uniquely identifies an instance of an entity. In an ER diagram, keys are depicted using an underline beneath the attribute(s) that form the key. For example, if the "Person" entity has an attribute called "SSN" (Social Security Number), it might be underlined to indicate that it is a key attribute.

ER diagrams are used to visualize the structure of a database and the relationships between its entities. They help software engineers, database designers, and stakeholders to understand the data model of a system and communication tool to discuss and document the data model of a system, facilitate design and implement databases more effectively. ER diagrams are also used as a g collaboration among
team members and stakeholders.

This is our ER diagram:



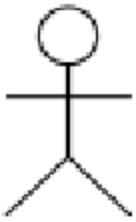
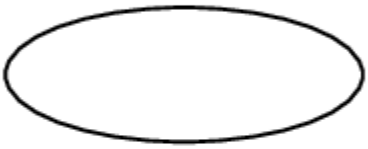
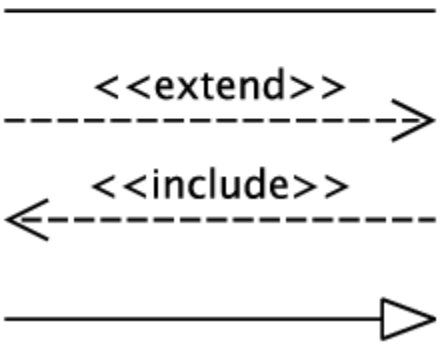
Use Case Diagram

A Use Case Diagram is a type of behavioral diagram defined in the Unified Modeling Language (UML) that is used to represent the interactions between users (actors) and a system. It is a powerful tool in software engineering for modeling the functional requirements of a system and the interactions between the system and its users.

Use Case Diagrams are used to model the functional requirements of a system and the interactions between the system and its users. They help software engineers, developers, and stakeholders to understand the behavior of a system and to design and implement software systems more effectively. Use Case Diagrams are also used as a communication tool to discuss and document the functional requirements of a system, facilitating collaboration among team members and stakeholders.

Overall, Use Case Diagrams are a valuable tool in software engineering for modeling the functional requirements of a system and the interactions between the system and its users. They provide a visual representation of the behavior of a system and help to ensure that the system meets the needs of its users.

The components of a Use Case Diagram:

Symbol	Reference Name
	Actor
	Use case
	Relationship

Explanation of the components of a Use Case Diagram:

1.Actor: An actor is a user or external system that interacts with the system being modeled. In a Use Case Diagram, actors are represented as stick figures. Each actor has a name that describes the role they play in the system. For example, in a banking system, an actor might be a "Customer" or a "Bank Employee."

2.Use Case: A use case represents a specific functionality or behavior of the system being modeled. It describes a sequence of interactions between the system and its actors to achieve a specific goal. In a Use Case Diagram, use cases are represented as ovals with the name of the use case inside. For example, in a banking system, a use case might be "Withdraw Money" or

"Transfer Funds."

3.Association: An association represents the relationship between an actor and a use case. It indicates that the actor is involved in the use case. In a Use Case Diagram, associations are represented as lines connecting the actor to the use case. For example, an association between a "Customer" actor and the "Withdraw Money" use case indicates that the customer is involved in the process of withdrawing money.

4.System Boundary: The system boundary represents the boundary of the system being modeled. It separates the system from its actors and shows what is inside the system and what is outside. In a Use Case Diagram, the system boundary is represented as a box that encloses the use cases and actors.

5.Extend Relationship: The extended relationship represents a relationship between two use cases where one use case (the extended use case) provides additional functionality to another use

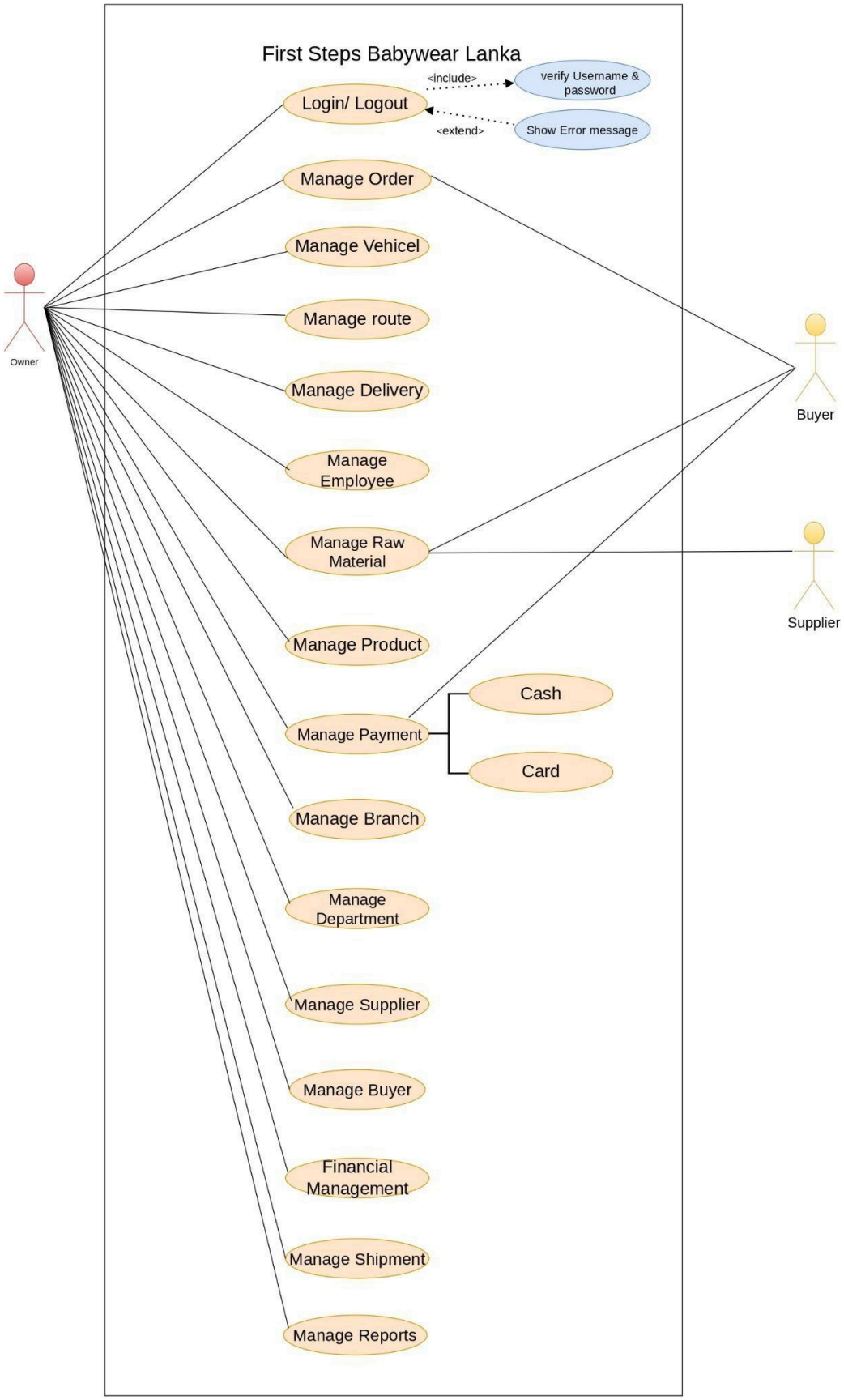
case (the base use case). It is used to model optional or alternative behavior. In a Use Case Diagram, the extend relationship is represented as a dashed line with an arrow pointing

from the extended use case to the base use case.

6.Include Relationship: The include relationship represents a relationship between two use cases where one use case (the included use case) includes the functionality of another use case (the included use case). It is used to

model common or shared behavior. In a Use Case Diagram, the include relationship is represented as a solid line with an arrow pointing from the included use case to the included use case.

This is our use case diagram.


First Steps Babywear Lanka
Login/ Logout
<include>
verify Username & password
<extend>
Show Error message
Manage Order
Manage Vehicel
Manage route
Manage Delivery
Manage Employee
Manage Raw Material
Manage Product
Manage Payment
Cash
Card
Manage Branch
Manage Department
Manage Supplier
Manage Buyer
Financial Management
Manage Shipment
Manage Reports
Owner
Buyer
Supplier

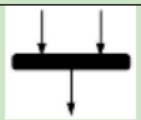
Activity Diagram

An activity diagram is a type of behavioral diagram defined in the Unified Modeling Language (UML) that is used to model the flow of control or data in a system. It is a powerful tool in software engineering for modeling the dynamic behavior of a system and the activities that take place within it.

Activity diagrams are used to model the dynamic behavior of a system and the activities that take place within it. They help software engineers, developers, and stakeholders to understand the flow of control or data in a system and to design and implement software systems more effectively. Activity diagrams are also used as a communication tool to discuss and document the dynamic behavior of a system, facilitating collaboration among team members and stakeholders.

Overall, activity diagrams are a valuable tool in software engineering for modeling the dynamic behavior of a system and the activities that take place within it. They provide a visual representation of the flow of control or data in a system and help to ensure that the system meets the needs of its users.

Components of an Activity Diagram:

Notations	Symbol	Meaning
Start		Shows the beginning of a process
Connector		Shows the directional flow, or control flow, of the activity
Joint symbol		Combines two concurrent activities and re-introduces them to a flow where one activity occurs at a time
Decision		Represents a decision
Note		Allows the diagram creators o communicate additional messages
Send signal		Show that a signal is being sent to a receiving activity
Receive signal		Demonstrates the acceptance of an event
Flow final symbol		Represents the end of a specific process flow
Option loop		Allows the creator to model a repetitive sequence within the option loop symbol
Shallow history pseudostate		Represents a transition that invokes the last active state.
End		Marks the end state of an activity and represents the completion of all flows of a process

Explanation of the components of an Activity Diagram:

1.Activity: An activity represents a specific action or task that takes place within the system being modeled. It describes a sequence of actions that are performed to achieve a specific goal. In an activity diagram, activities are represented as rounded rectangles with the name of the activity inside. For example, in a banking system, an activity might be "Withdraw Money" or

"Transfer Funds."

2.Control Flow: Control flow represents the flow of control or the sequence of activities that take place in the system. It is used to model the order in which activities are performed and the conditions that determine the flow of control. In an activity diagram, control flow is represented as arrows connecting the activities. For example, an arrow from the "Withdraw Money" activity to the

"Transfer Funds" activity indicates that the "Withdraw Money" activity must be completed before the "Transfer Funds" activity can begin.

3.Initial Node: The initial node represents the starting point of the activity diagram. It indicates where the control flow begins. In an activity diagram, the initial node is represented as a small solid circle with an arrow pointing to the first activity.

4.Final Node: The final node represents the ending point of the activity diagram. It indicates where the control flow ends. In an activity diagram, the final node is represented as a small solid circle with no arrows pointing to it.

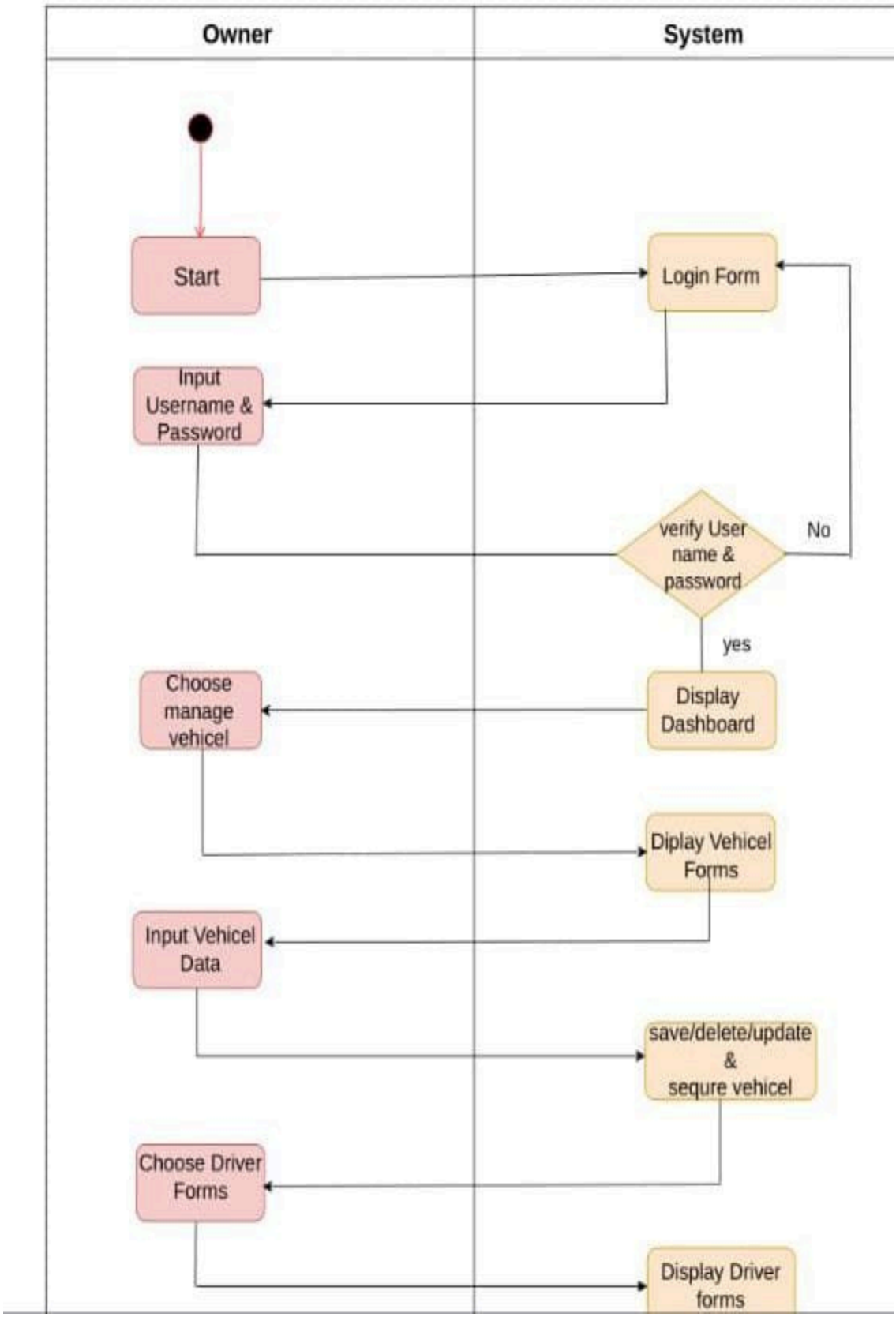
5.Decision Node: The decision node represents a decision point in the activity diagram where the flow of control can take different paths depending on the outcome of a decision. It is used to model conditional behavior. In an activity diagram, the decision node is represented as a diamond shape with arrows pointing to the different paths.

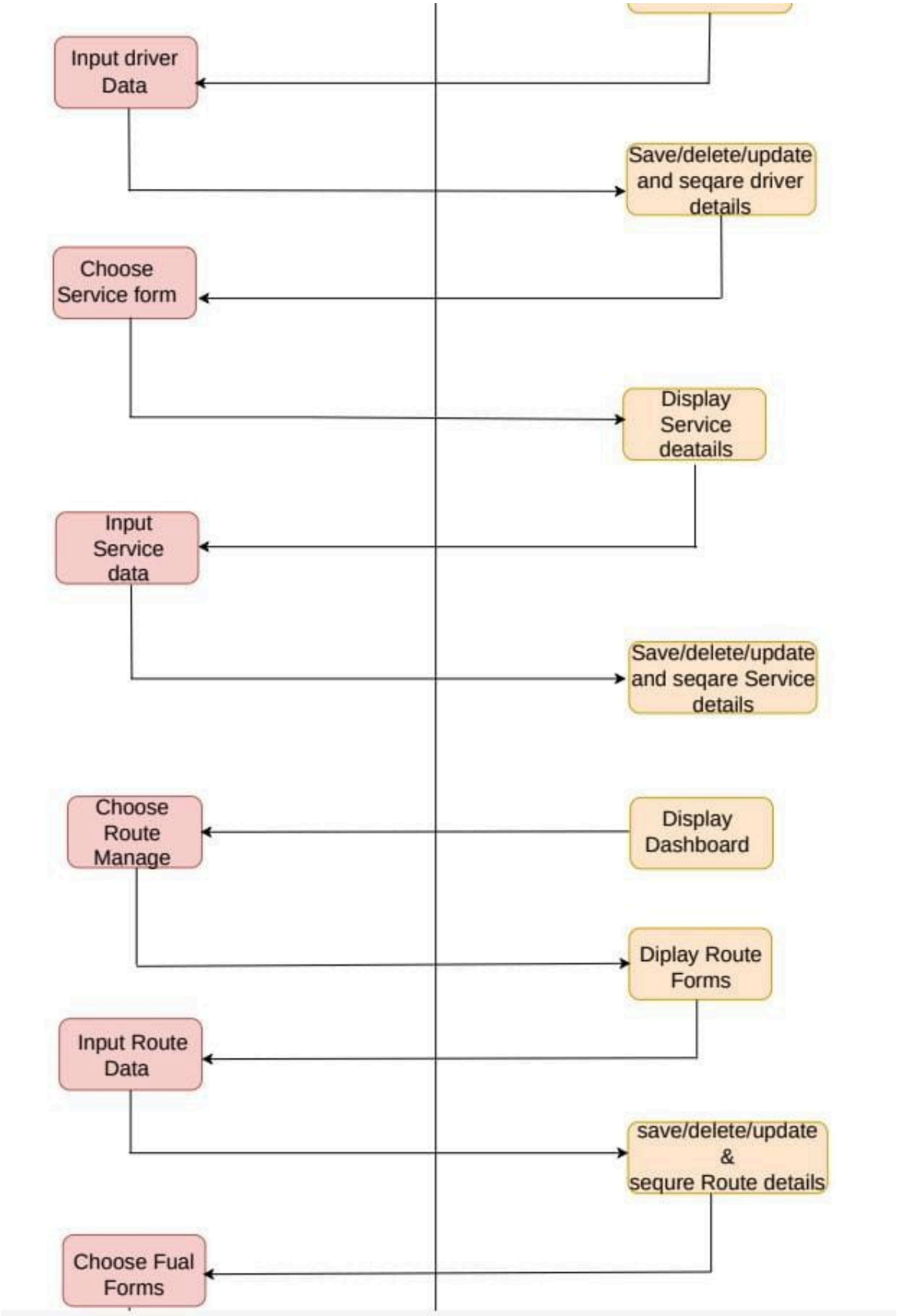
6.Merge Node: The merge node represents a point in the activity diagram where multiple control flows converge into a single flow. It is used to model the merging of parallel flows. In an activity diagram, the merge node is represented as a diamond shape with arrows pointing to it.

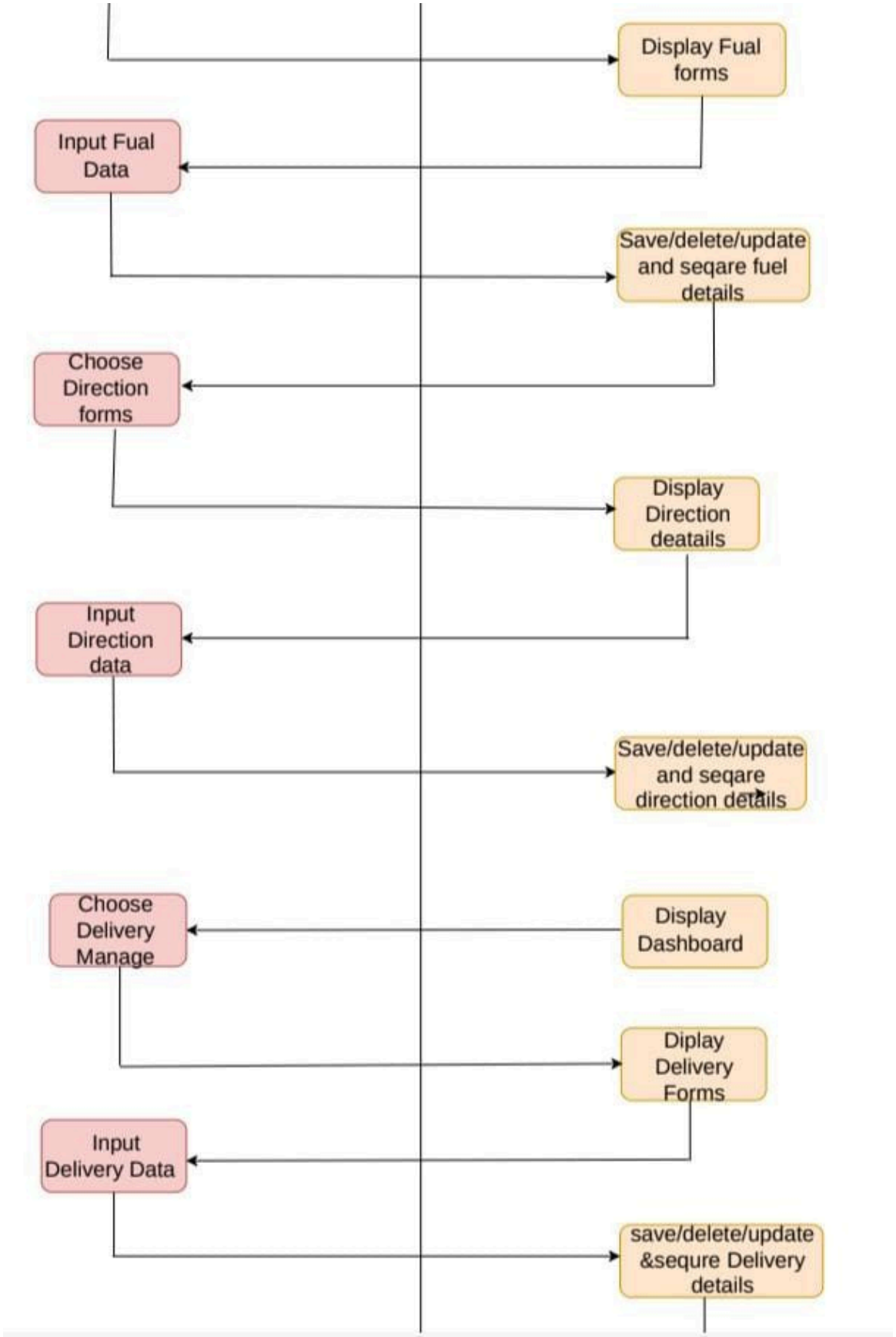
7.Fork Node: The fork node represents a point in the activity diagram where a single control flow splits into multiple parallel flows. It is used to model concurrent behavior. In an activity diagram, the fork node is represented as a solid vertical line with arrows pointing away from it.

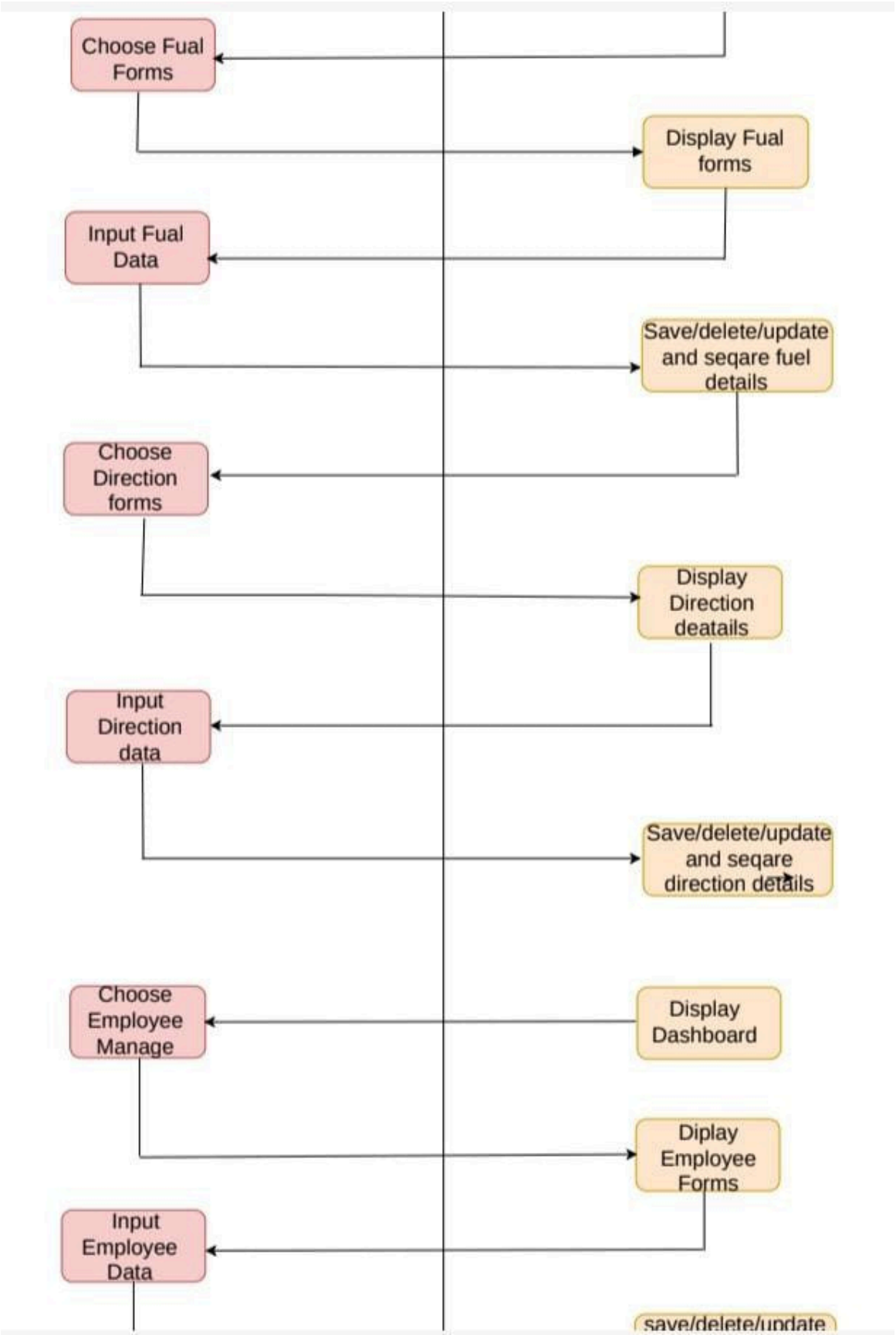
8.Join Node: The join node represents a point in the activity diagram where multiple parallel flows converge into a single flow. It is used to model the joining of parallel flows. In an activity diagram, the join node is represented as a solid vertical line with arrows pointing to it.

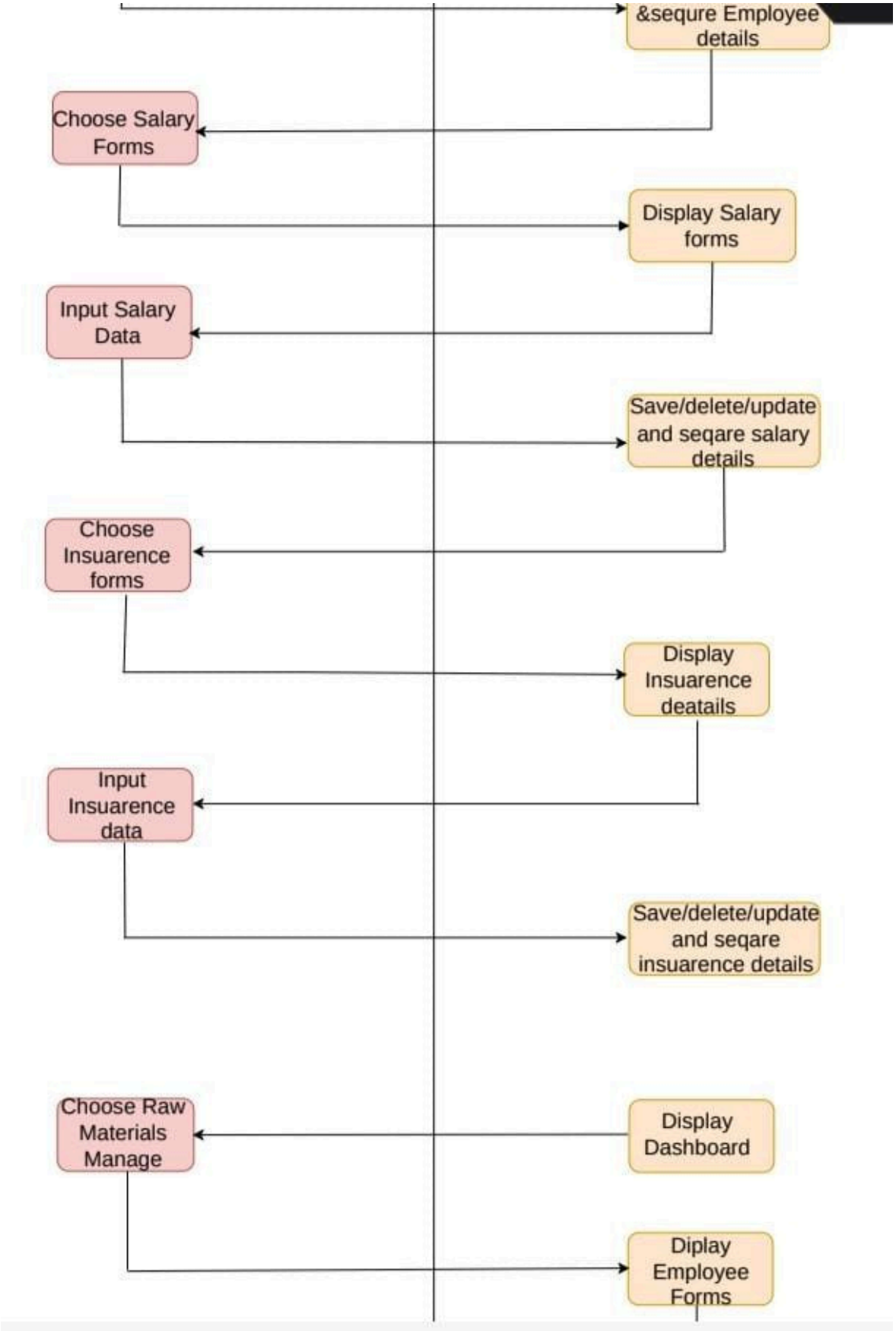
This is our Activity Diagram:

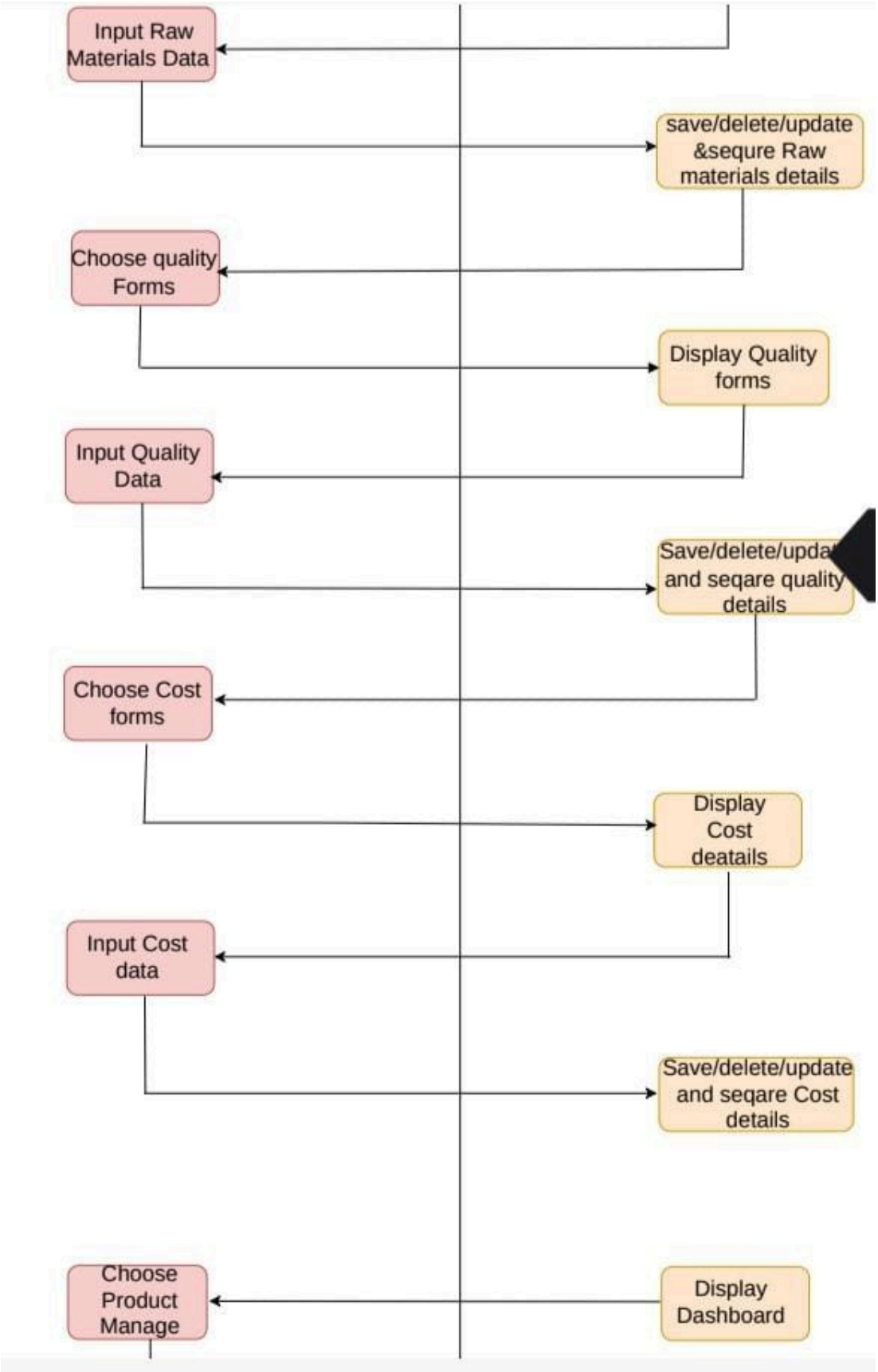

Owner
System
Start
Login Form
Input Username & Password
verify User name & password
No
yes
Choose manage vehicel
Display Dashboard
Diplay Vehicel Forms
Input Vehicel Data
save/delete/update & sequre vehicel
Choose Driver Forms
Display Driver forms

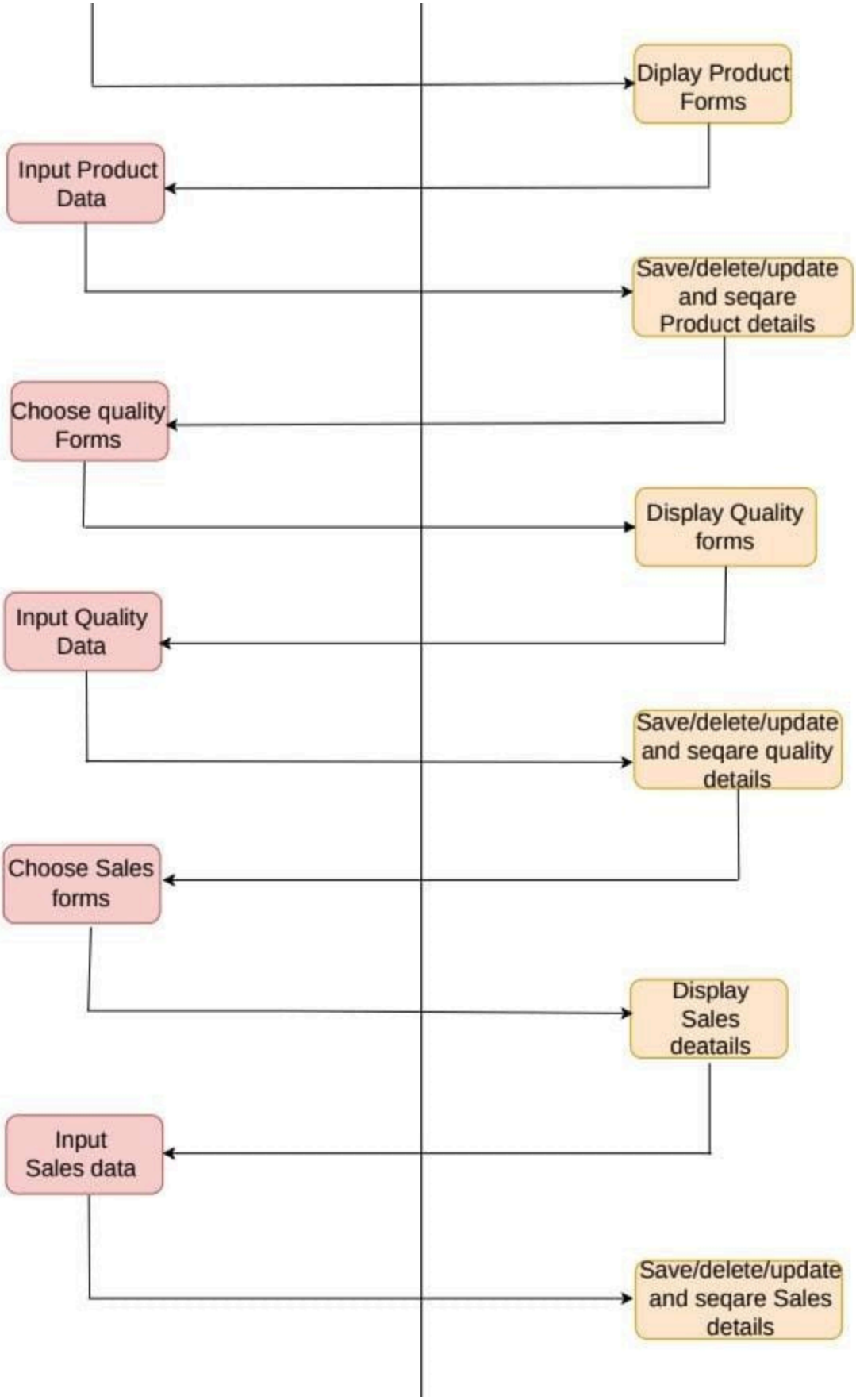

Input driver Data
Save/delete/update and seqare driver details
Choose Service form
Display Service deatails
Input Service data
Save/delete/update and seqare Service details
Choose Route Manage
Display Dashboard
Diplay Route Forms
Input Route Data
save/delete/update & sequre Route details
Choose Fual Forms

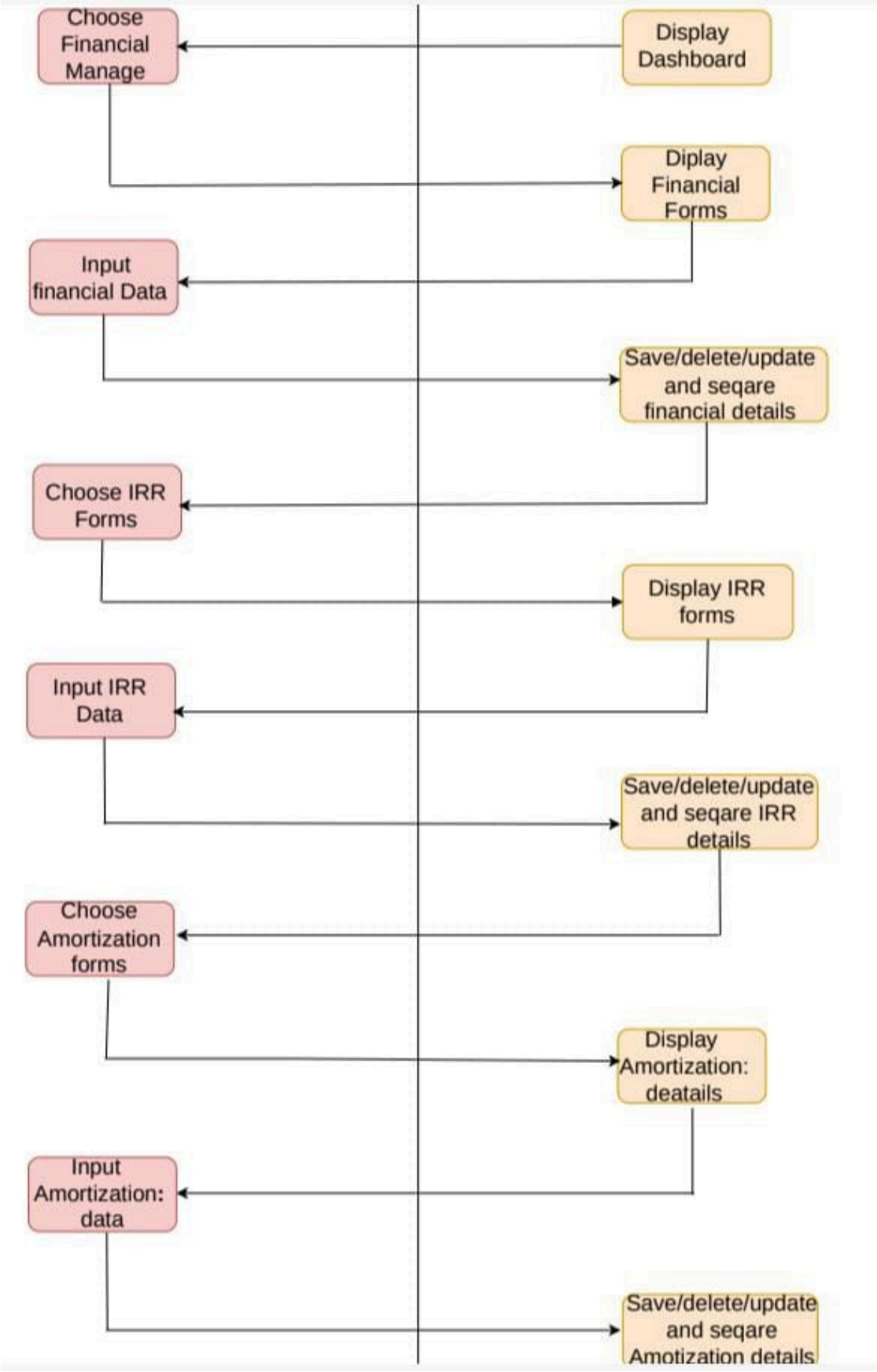

Display Fual forms
Input Fual Data
Save/delete/update and seqare fuel details
Choose Direction forms
Display Direction deatails
Input Direction data
Save/delete/update and seqare direction details
Choose Delivery Manage
Display Dashboard
Diplay Delivery Forms
Input Delivery Data
save/delete/update &sequre Delivery details

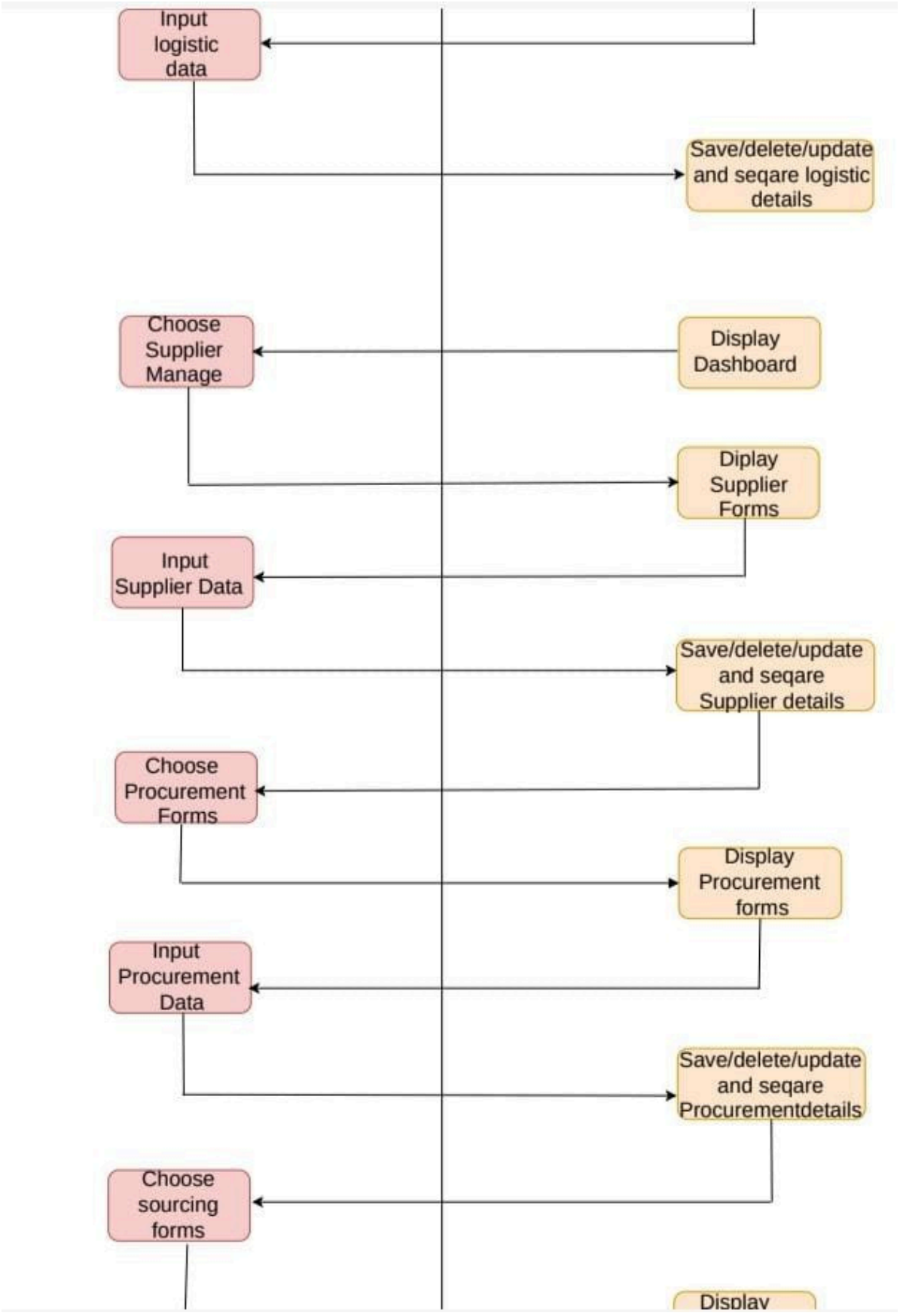

Choose Fual Forms
Display Fual forms
Input Fual Data
Save/delete/update and seqare fuel details
Choose Direction forms
Display Direction deatails
Input Direction data
Save/delete/update and seqare direction details
Choose Employee Manage
Display Dashboard
Diplay Employee Forms
Input Employee Data
save/delete/update

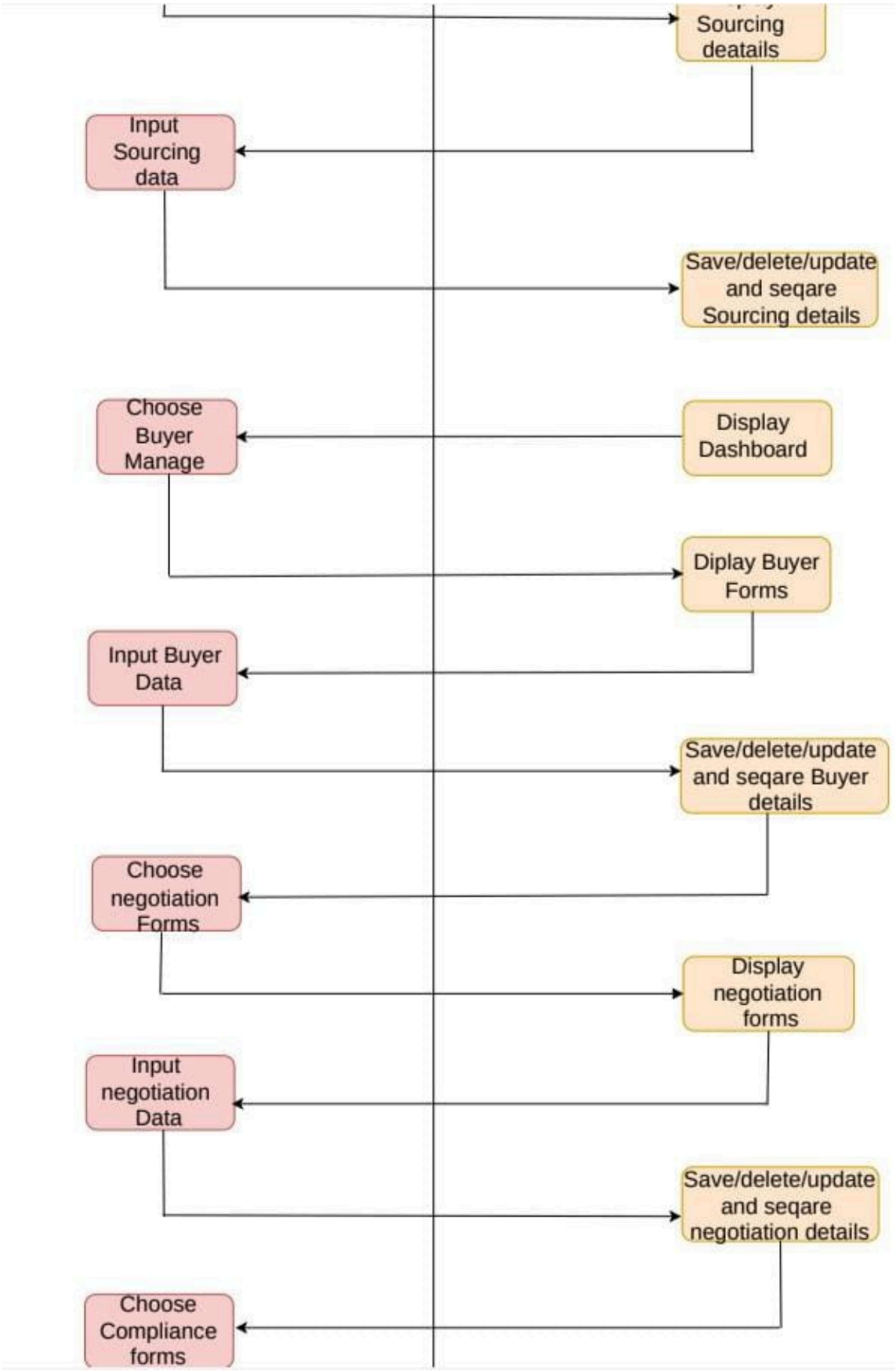

&sequre Employee details
Choose Salary Forms
Display Salary forms
Input Salary Data
Save/delete/update and seqare salary details
Choose Insuarence forms
Display Insuarence deatails
Input Insuarence data
Save/delete/update and seqare insuarence details
Choose Raw Materials Manage
Display Dashboard
Diplay Employee Forms

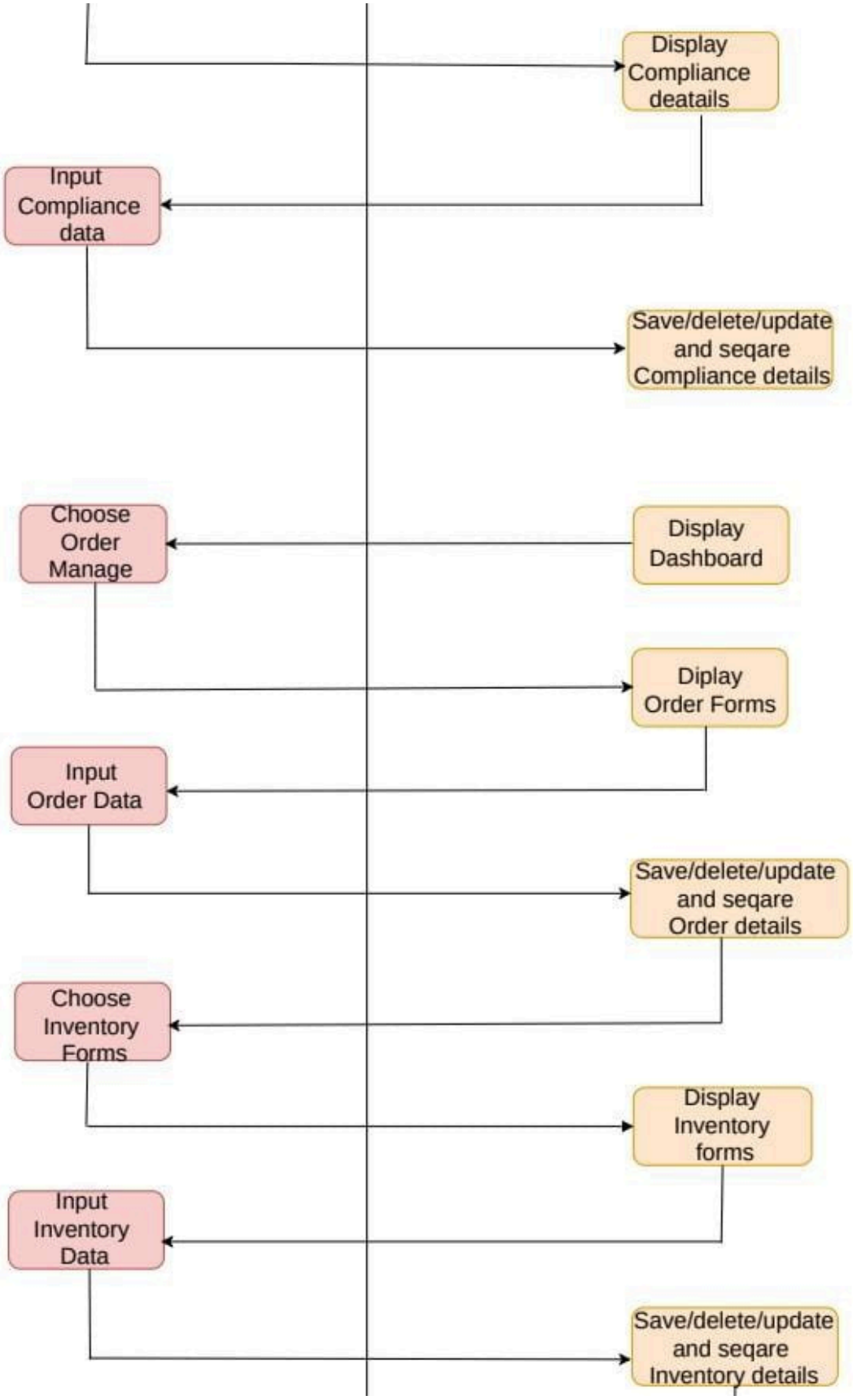

Input Raw Materials Data
save/delete/update &sequre Raw materials details
Choose quality Forms
Display Quality forms
Input Quality Data
Save/delete/upda and seqare quality details
Choose Cost forms
Display Cost deatails
Input Cost data
Save/delete/update and seqare Cost details
Choose Product Manage
Display Dashboard

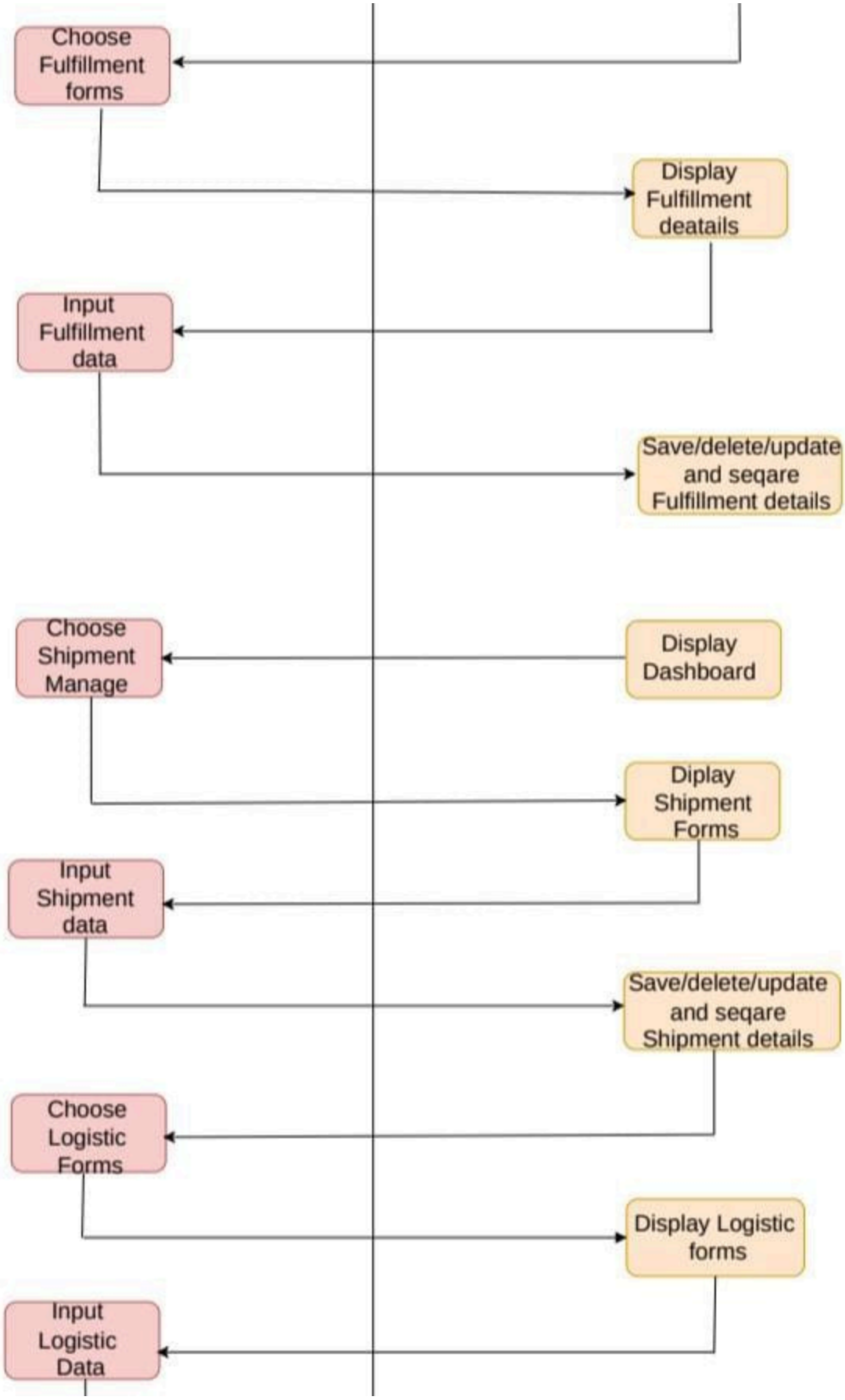

Diplay Product Forms
Input Product Data
Save/delete/update and seqare Product details
Choose quality Forms
Display Quality forms
Input Quality Data
Save/delete/update and seqare quality details
Choose Sales forms
Display Sales deatails
Input Sales data
Save/delete/update and seqare Sales details

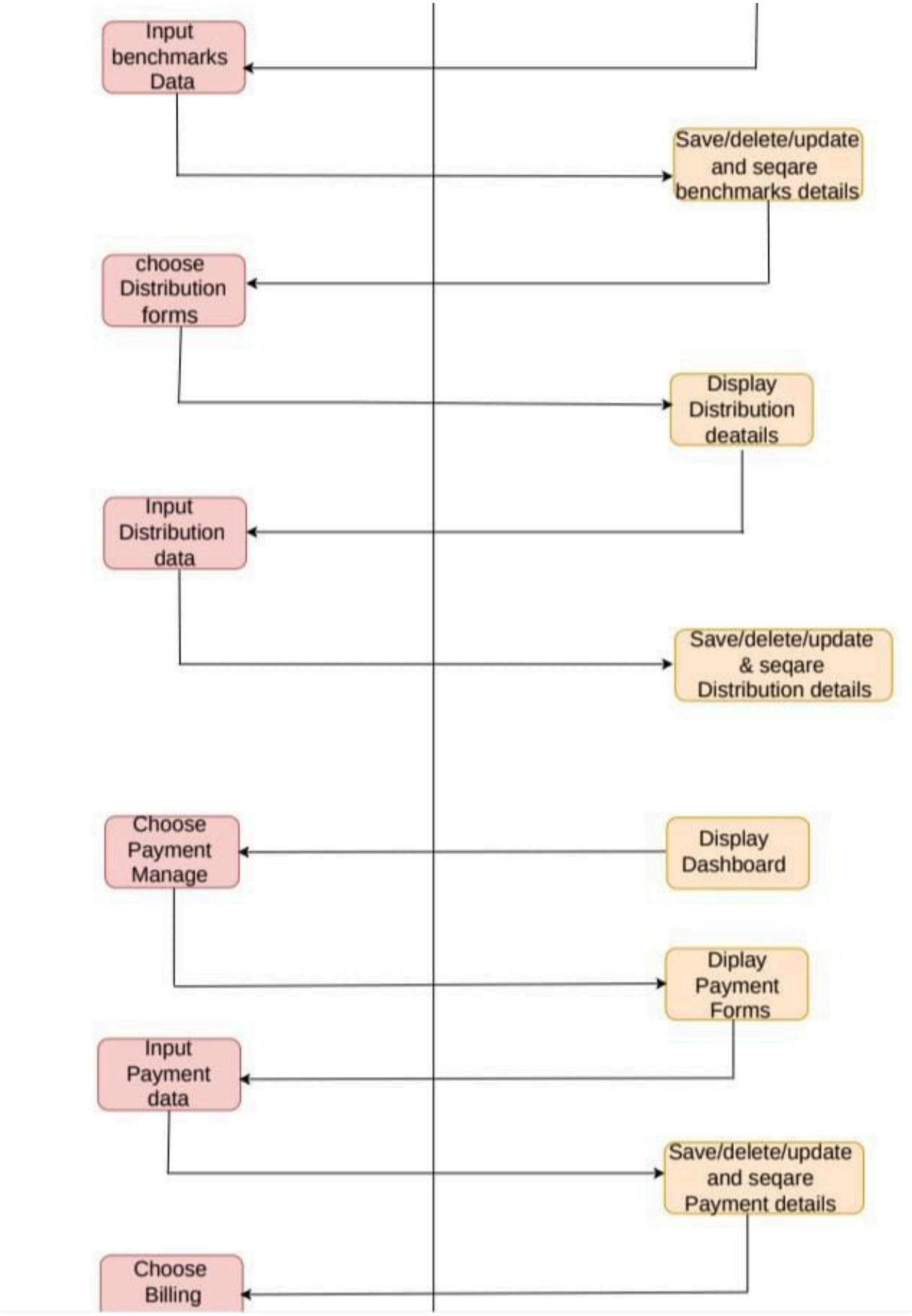

Choose Financial Manage
Display Dashboard
Diplay Financial Forms
Input financial Data
Save/delete/update and seqare financial details
Choose IRR Forms
Display IRR forms
Input IRR Data
Save/delete/update and seqare IRR details
Choose Amortization forms
Display Amortization: deatails
Input Amortization: data
Save/delete/update and seqare Amotization details

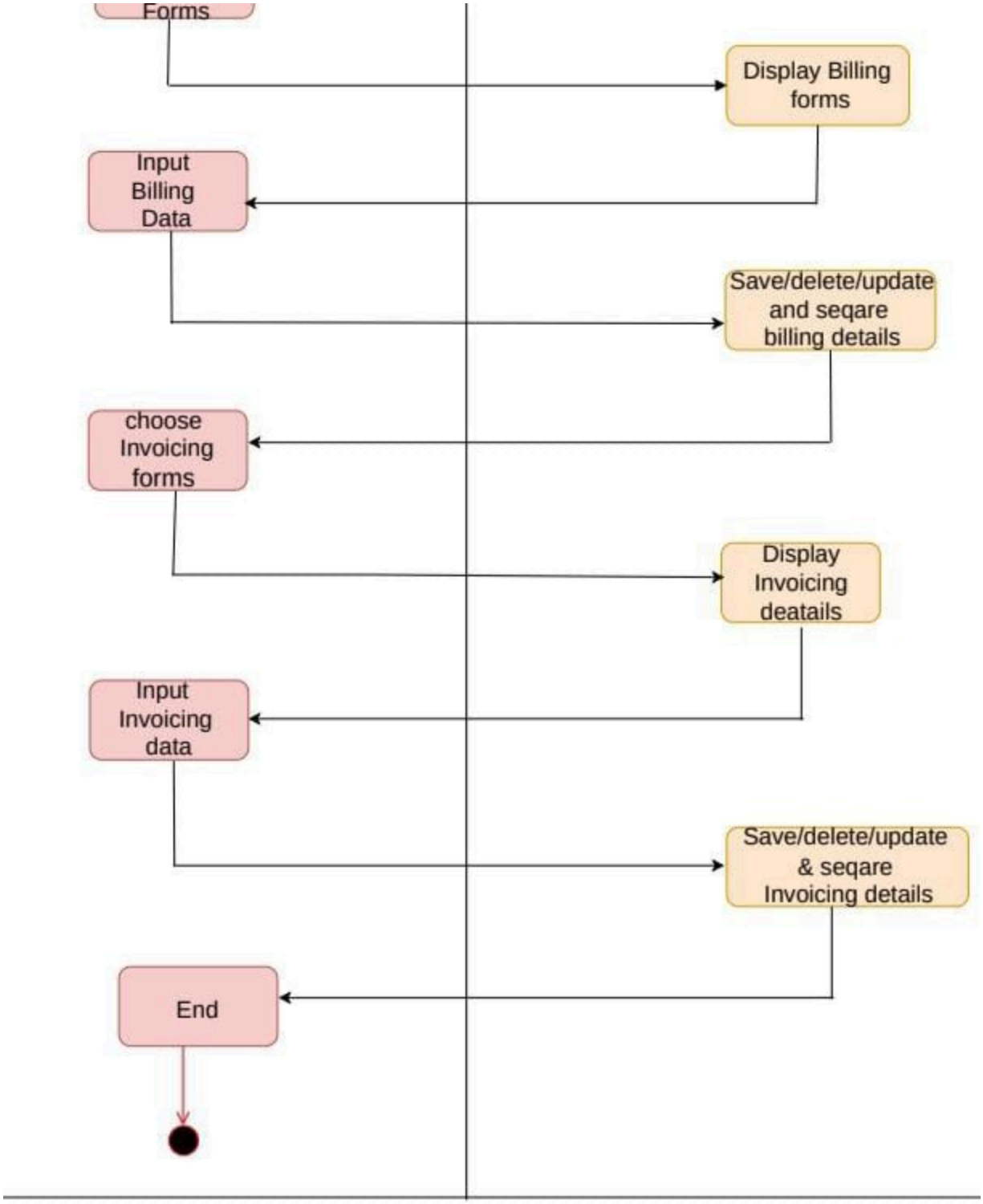

Input logistic data
Save/delete/update and seqare logistic details
Choose Supplier Manage
Display Dashboard
Diplay Supplier Forms
Input Supplier Data
Save/delete/update and seqare Supplier details
Choose Procurement Forms
Display Procurement forms
Input Procurement Data
Save/delete/update and seqare Procurementdetails
Choose sourcing forms
Display


Sourcing deatails
Input Sourcing data
Save/delete/update and seqare Sourcing details
Choose Buyer Manage
Display Dashboard
Diplay Buyer Forms
Input Buyer Data
Save/delete/update and seqare Buyer details
Choose negotiation Forms
Display negotiation forms
Input negotiation Data
Save/delete/update and seqare negotiation details
Choose Compliance forms


Display Compliance deatails
Input Compliance data
Save/delete/update and seqare Compliance details
Choose Order Manage
Display Dashboard
Diplay Order Forms
Input Order Data
Save/delete/update and seqare Order details
Choose Inventory Forms
Display Inventory forms
Input Inventory Data
Save/delete/update and seqare Inventory details


Choose Fulfillment forms
Display Fulfillment deatails
Input Fulfillment data
Save/delete/update and seqare Fulfillment details
Choose Shipment Manage
Display Dashboard
Diplay Shipment Forms
Input Shipment data
Save/delete/update and seqare Shipment details
Choose Logistic Forms
Display Logistic forms
Input Logistic Data


Input benchmarks Data
Save/delete/update and seqare benchmarks details
choose Distribution forms
Display Distribution deatails
Input Distribution data
Save/delete/update & seqare Distribution details
Choose Payment Manage
Display Dashboard
Diplay Payment Forms
Input Payment data
Save/delete/update and seqare Payment details
Choose Billing


Forms
Display Billing forms
Input Billing Data
Save/delete/update and seqare billing details
choose Invoicing forms
Display Invoicing deatails
Input Invoicing data
Save/delete/update & seqare Invoicing details
End

SourceURL:file:///home/praween/Downloads/Diagrams.docx

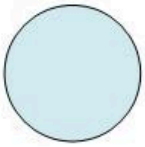
DFD Diagram

A Data Flow Diagram (DFD) is a graphical representation of the flow of data through a system. It is a powerful tool in software engineering for modeling the data flow and processes within a system. DFDs are used to visualize and document the flow of data between different components of a system, such as processes, data stores, and external entities.

DFDs are used to model the flow of data through a system and the processes that manipulate the data. They help software engineers, developers, and stakeholders to understand the data flow and processes within a system and to design and implement software systems more effectively. DFDs are also used as a communication tool to discuss and document the data flow and processes within a system, facilitating collaboration among team members and stakeholders.

Overall, DFDs are a valuable tool in software engineering for modeling the flow of data through a system and the processes that manipulate the data. They provide a visual representation of the data flow and processes within a system and help to ensure that the system meets the needs of its users.

Components of a DFD:

Notation	De Marco & Yourdon	Gane and Sarson
External Entity		
Process		
Data Store		
Data Flow		

Explanation of the components of a DFD:

1.External Entity: An external entity represents an external system, user, or process that interacts with the system being modeled. It is used to model the inputs and outputs of the system. In a DFD, external entities are represented as rectangles with the name of the entity inside. For example, in a banking system, an external entity might be a "Customer" or a "Bank Employee."

2.Process: A process represents a specific action or task that takes place within the system being modeled. It describes the transformation of data from input to output. In a DFD, processes are represented as circles with the name of the process inside. For example, in a banking system, a process might be"Withdraw Money" or "Transfer Funds."

3.Data Store: A data store represents a repository of data within the system being modeled. It is used to model the storage and retrieval of data. In a DFD, data stores are represented as rectangles with the name of the data store inside. For example, in a banking system, a data store might be "Account Information" or "Transaction History."

4.Data Flow: A data flow represents the flow of data between different components of the system. It is used to model the movement of data from one component to another. In a DFD, data flows are represented as arrows connecting the different components. For example, an arrow from an external entity to a process represents the input of data into the process, while an arrow from a process to a data store represents the storage of data in the data store.

DFD Diagram are three types

- DFD level 0
- DFD level 1
- DFD level 2

DFD Level 0

A Data Flow Diagram (DFD) Level 0 is the highest-level view of a system's processes, data flows, and external entities. It provides a broad overview of the system's functionality and the interactions between its components. DFD Level 0 is also known as a Context Diagram because it shows the context in which the system operates, including the external entities that interact with the system.

DFD Level 0 is used to provide a high-level view of the system's functionality and the interactions between its components. It helps software engineers, developers, and stakeholders to understand the overall structure of the system and the context in which it operates. DFD Level 0 is also used as a communication tool to discuss and document the system's functionality and interactions, facilitating collaboration among team members and stakeholders.

Overall, DFD Level 0 is a valuable tool in software engineering for providing a high-level view of a system's functionality and the interactions between its components. It provides a broad overview of the system's structure and helps to ensure that the system meets the needs of its users.

Explanation of the components of a DFD Level 0:

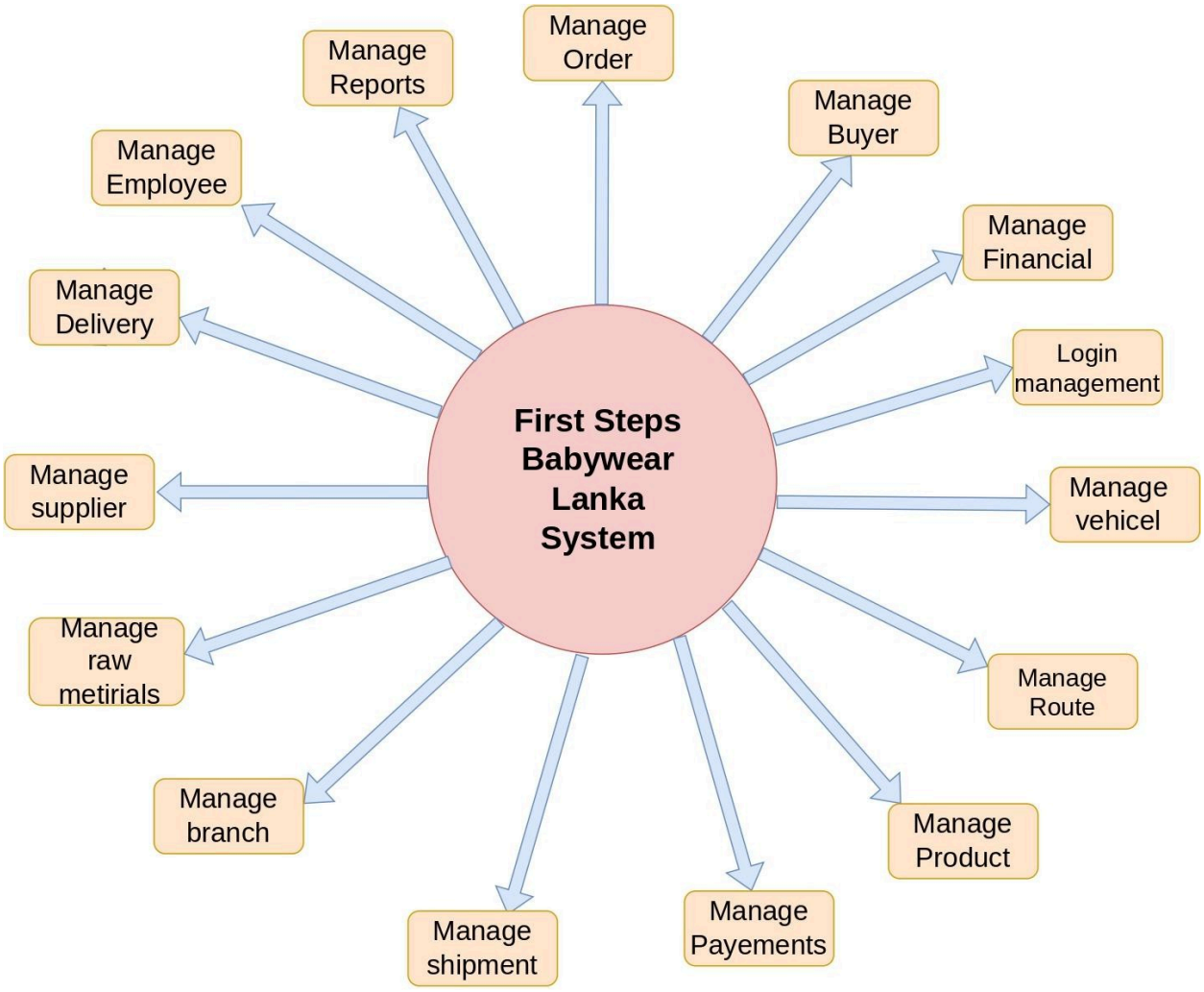
1.External Entities: External entities are entities that interact with the system but are not part of the system itself. They represent the inputs and outputs of the system. In a DFD Level 0, external entities are represented as rectangles with the name of the entity inside. For example, in a banking system, external entities might include "Customer," "Bank Employee," and "ATM."

2.Processes: Processes are the actions or tasks that take place within the system. They represent the transformation of data from input to output. In a DFD Level 0, processes are represented as circles with the name of the process inside. For example, in a banking system, processes might include "Withdraw Money," "Transfer Funds," and "Check Balance."

3.Data Flows: Data flows represent the movement of data between different components of the system. They show how data flows from one component to another. In a DFD Level 0, data flows are represented as arrows connecting the different components. For example, an arrow from an external entity to a process represents the input of data into the process, while an arrow from a process to an external entity represents the output of data from the process.

4.Data Stores: Data stores represent repositories of data within the system. They show where data is stored and retrieved. In a DFD Level 0, data stores are represented as rectangles with the name of the data store inside. For example, in a banking system, data stores might include "Account Information" and "Transaction History."

This is our DFD level 0 Diagram



DFD Level 1

A Data Flow Diagram (DFD) Level 1 is a more detailed view of a system's processes, data flows, and external entities. It provides a more granular view of the system's functionality and the interactions between its components compared to a DFD Level 0. DFD Level 1 is also known as a Decomposed DFD because it breaks down the processes and data flows into more detailed components.

DFD Level 1 is used to provide a more detailed view of the system's functionality and the interactions between its components. It helps software engineers, developers, and stakeholders to understand the detailed structure of the system and the interactions between its components. DFD Level 1 is also used as a communication tool to discuss and document the system's functionality and interactions, facilitating collaboration among team members and stakeholders.

Overall, DFD Level 1 is a valuable tool in software engineering for providing a detailed view of a system's functionality and the interactions between its components. It provides a more granular view of the system's structure and helps to ensure that the system meets the needs of its users.

Explanation of the components of a DFD Level 1:

1.External Entities: External entities are entities that interact with the system but are not part of the system itself. They represent the inputs and outputs of the system. In a DFD Level 1, external entities are represented as rectangles with the name of the entity inside. For example, in a banking system, external entities might include "Customer," "Bank Employee," and "ATM."

2.Processes: Processes are the actions or tasks that take place within the system. They represent the transformation of data from input to output. In a DFD Level 1, processes are represented as circles with the name of the process inside. For example, in a banking system, processes might include "Withdraw Money," "Transfer Funds," and "Check Balance."

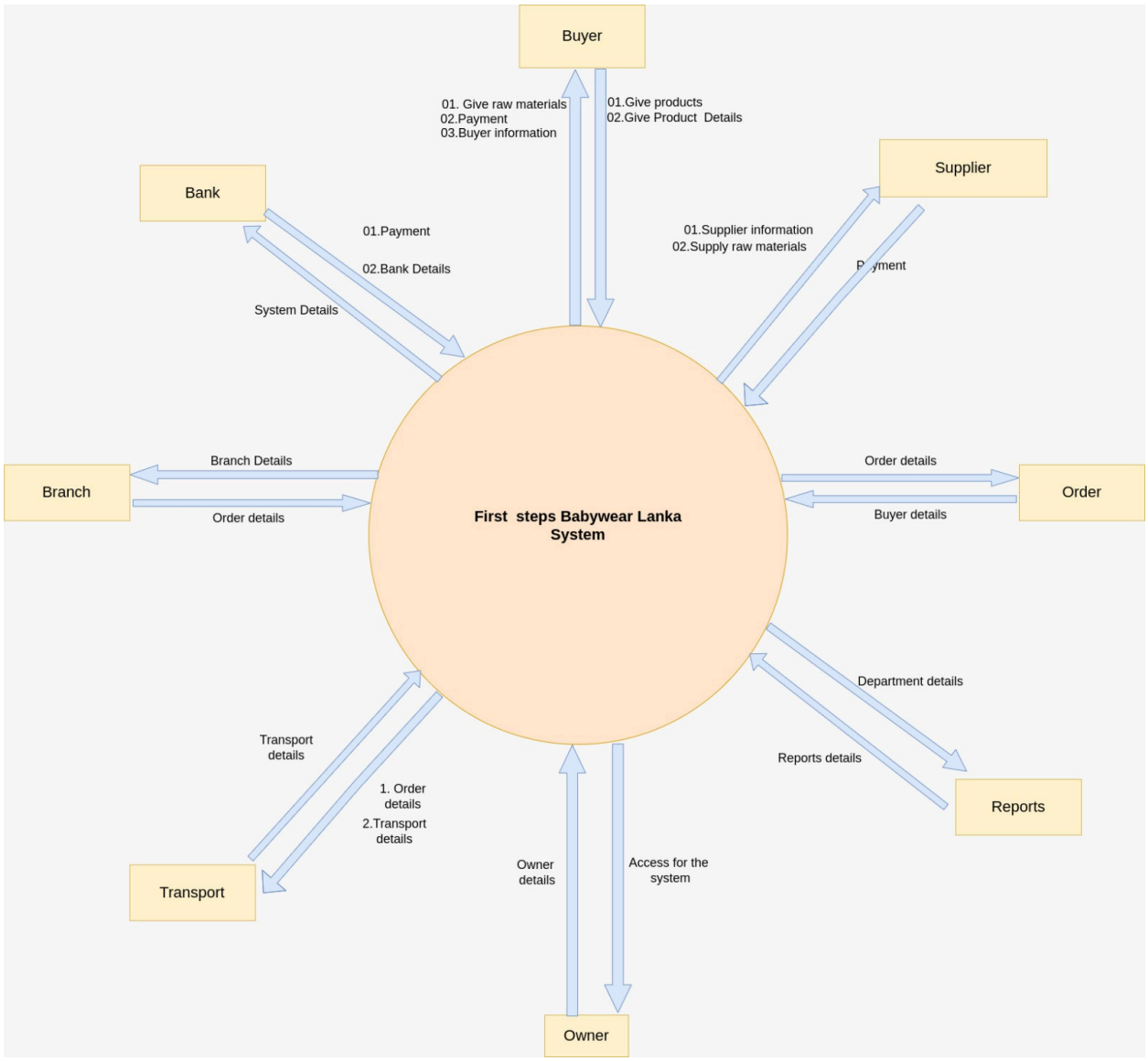
3.Data Flows: Data flows represent the movement of data between different components of the system. They show how data flows from one component to another. In a DFD Level 1, data flows are represented as arrows connecting the different components. For example, an arrow from an external entity to a process represents the input of data into the process, while an arrow from a process to an external entity represents the output of data from the process.

4.Data Stores: Data stores represent repositories of data within the system. They show where data is stored and retrieved. In a DFD Level 1, data stores are represented as rectangles with the name of the data store inside. For example, in a banking system, data stores might include "Account Information" and "Transaction History."

5.Child Diagrams: Child diagrams are used to represent more detailed views of the processes and data flows within a system. They provide a more granular

view of the system's functionality. In a DFD Level 1, child diagrams are represented as rectangles with the name of the child diagram inside. For example, a child diagram might include more detailed processes and data flows related to the "Withdraw Money" process.

This is our DFD Level 1 Diagram



DFD Level 2

DFD Level 2 is used to provide a more detailed view of the system's functionality and the interactions between its components. It helps software engineers, developers, and stakeholders to understand the detailed structure of the system and the interactions between its components. DFD Level 2 is also used as a communication tool to discuss and document the system's functionality and interactions, facilitating collaboration among team members and stakeholders.

Overall, DFD Level 2 is a valuable tool in software engineering for providing a detailed view of a system's functionality and the interactions between its components. It provides a more granular view of the system's structure and helps to ensure that the system meets the needs of its users.

A Data Flow Diagram (DFD) Level 2 is a more detailed view of a system's processes, data flows, and external entities compared to a DFD Level 1. It provides a more granular view of the system's functionality and the interactions between its components. DFD Level 2 is also known as a Decomposed DFD because it breaks down the processes and data flows into more detailed components.

Explanation of the components of a DFD Level 2:

1.External Entities: External entities are entities that interact with the system but are not part of the system itself. They represent the inputs and outputs of the system. In a DFD Level 2, external entities are represented as rectangles with the name of the entity inside. For example, in a banking system, external entities might include "Customer," "Bank Employee," and "ATM."

2.Processes: Processes are the actions or tasks that take place within the system. They represent the transformation of data from input to output. In a DFD Level 2, processes are represented as circles with the name of the process inside. For example, in a banking system, processes might include "Withdraw Money," "Transfer Funds," and "Check Balance."

3.Data Flows: Data flows represent the movement of data between different components of the system. They show how data flows from one component to another. In a DFD Level 2, data flows are represented as arrows connecting the different components. For example, an arrow from an external entity to a process represents the input of data into the process, while an arrow from a process to an external entity represents the output of data from the process.

4.Data Stores: Data stores represent repositories of data within the system. They show where data is stored and retrieved. In a DFD Level 2, data stores are represented as rectangles with the name of the data store inside. For example, in a banking system, data stores might include "Account Information" and "Transaction History."

2. **Transportation Routes:**

International Shipping: Fabric and other raw materials are transported from the Indian fabric mill to Sri Lanka through international shipping channels. The choice of shipping routes and logistics partners is crucial to ensure timely and cost-effective deliveries.

- **Inland Transportation:** Upon arrival in Sri Lanka, the materials are transported inland to the Panadura head office and other branches. This involves coordination with local transportation services, considering factors such as distance, road conditions, and potential bottlenecks.

3. **Head Office Operations:**

- **Centralized Operations:** The Panadura head office serves as the central hub for key operations, including cutting, printing, and embroidering. Raw materials received from the fabric mill are distributed to different sections for processing.

- **Manufacturing Processes:** The cutting, printing, and embroidering sections at the head office contribute to the creation of finished products. These processes require a constant and timely supply of raw materials to meet production schedules.

4. **Branch Operations:**

    - **Sewing and Quality Checks:** The sewing of clothes is distributed among the nine branches. After the clothes are sewn, a rigorous quality check is conducted at each branch to ensure that the finished products meet the company's standards before export.

    - **Export to UK and US:** Once the quality checks are completed, the finished products are exported to the company's buyers in the UK and US. The transport system plays a crucial role in ensuring that the exported goods reach their destinations within the specified timelines.

5. **GPS Tracking System:**

    - **Current System:** The existing transport system relies on GPS technology for tracking the movement of vehicles carrying raw materials and finished products. However, challenges related to driver adherence and scheduling deviations have been identified.

    - **Planned Software Solution:** In response to the identified challenges, the company plans to implement a software solution that will enhance control and monitoring capabilities. This software aims to provide real-time tracking, optimize routes, and address the issues affecting the efficiency of the current transport system.

6. **Integration with Operations:**

- **Compatibility:** The new software solution is intended to seamlessly integrate with existing operational systems, such as inventory management and order processing. This integration ensures that the transport system aligns with overall company processes.

JV	**LOCATION**	**DISTANCE FROM PANADURA**
SISALU	MEDIRIGIRIYA	248 KM
RISALKA	GALENBINDUNUWEWA	255 KM
RICHLIGHT	RATHNAPURA	75KM
F&F	YATIYANTOTA	81 KM
JUMEIRAH	MONARAGALA	316 KM
DAG	MONARAGALA	298 KM
DSL	KARANDENIYA	74 KM
DAG	THALALLA	152 KM
ISABELLAA	PALLEKELE	176 KM
RITZ	NIKAWERATIYA	168 KM
NORWOOD	POLGAHAWELA	104KM

LORRY TRANSPORT PROPOSAL

Capacity	Monday	Tuesday	Wednesday	Thursday	Friday
20'	Sisalu | Risalka				Jumeirah| DAG
20'	Jumeirah| DAG				Sisalu | Risalka
20'		Jumeiragh| DAG			
18 1/2		Sisalu | Risalka		Jumeirah | DAG	
08 1/2	Embellishment Plant	Embellishment Plant	Embellishment Plant	Embellishment Plant	Embellishment Plant
14 1/2	Richlight | F&F		Sisalu | Risalka		Richlight | F&F
14 1/2		Richlight | F&F		Richlight | F&F	

14 1/2	Isabella | Ritz N	DSL	Isabella | Ritz N	DSL	Isabella | Ritz N

Functional requirements outline the specific features and capabilities that the Transport Management Software (TMS) should have to meet the needs of First Steps Babywear Lanka. These requirements are essential for the effective functioning of the software and addressing the challenges identified in the current transport system. The functional requirements for the TMS include:

1. Real-Time Tracking:

    - Objective: Provide real-time visibility into the location and status of vehicles transporting raw materials and finished products.
    - Requirements:
        - GPS integration for accurate tracking.
        - Map-based visualization of vehicle routes.
        - Update frequency for real-time monitoring.

2. Route Optimization:

    - Objective: Optimize transportation routes to minimize delays and reduce costs.
    - Requirements:
        - Algorithmic route optimization based on traffic, distance, and other factors.
        - Integration with real-time traffic and weather data.
        - Ability to dynamically adjust routes based on changing conditions.

3. Driver Performance Monitoring:

    - Objective: Monitor and analyze driver behavior to ensure adherence to schedules and safe driving practices.

- Requirements:
    - Driver behavior analytics, including speed, breaks, and adherence to schedules.
    - Performance metrics for individual drivers.
    - Incentive and penalty tracking based on driver performance.

4. Communication Tools:
    - Objective: Facilitate communication among stakeholders involved in the transportation process.
    - Requirements:
        - Two-way communication between drivers and central control.
        - Alerts and notifications for critical events.
        - Messaging platform for collaboration between drivers, logistics, and management.

5. Integration with Existing Systems:
    - Objective: Seamlessly integrate the TMS with existing operational systems to ensure data consistency and accuracy.
    - Requirements:
        - Application Programming Interfaces (APIs) for integration.
        - Data synchronization with inventory management and order processing systems.
        - Compatibility with databases used by the company.

6. User Roles and Permissions:
    - Objective: Define roles and permissions to control access to different features and data within the TMS.
    - Requirements:
        - Role-based access control (RBAC).
        - Different levels of access for drivers, logistics personnel, and management.
        - Authentication mechanisms to ensure secure access.

7. Mobile Application for Drivers:

    - Objective: Provide drivers with a mobile application for real-time updates, navigation assistance, and communication.
    - Requirements:
        - GPS navigation for optimized routes.
        - Driver logbook for recording working hours.
        - User-friendly interface for easy interaction.

8. Reporting and Analytics:
    - Objective: Generate reports and analytics to monitor key performance indicators and make data-driven decisions.
    - Requirements:
        - Customizable reports for management.
        - Performance metrics, including on-time delivery rates and fuel efficiency.
        - Trend analysis and predictive modeling for future planning.

9. Training Module:
    - Objective: Provide training modules for users to ensure effective utilization of the TMS.
    - Requirements:
        - Online training materials and tutorials.
        - User guides and documentation.
        - Certification or assessment for user proficiency.

10. Security Measures:
    - Objective: Ensure the security and integrity of data within the TMS.
    - Requirements:
        - Data encryption for sensitive information.
        - Secure login mechanisms.
        - Audit trails for monitoring user activities.

11. Scalability:
    - Objective: Design the TMS to scale with the growing needs of the company.
    - Requirements:
        - Ability to handle increased data loads.
        - Compatibility with future hardware and software upgrades.
        - Performance testing to ensure scalability.

12. User Interface (UI) Customization:
    - Objective: Allow users to customize the TMS interface based on their preferences.
    - Requirements:
        - Personalized dashboards for different user roles.
        - Configurable settings for map views and data displays.
        - Accessibility features for diverse user needs.

13. Audit Trail:
    - Objective: Maintain an audit trail to track changes, activities, and system events within the TMS.
    - Requirements:
        - Record of user actions and system events.
        - Timestamps for all entries in the audit trail.
        - Secure storage of audit trail data.

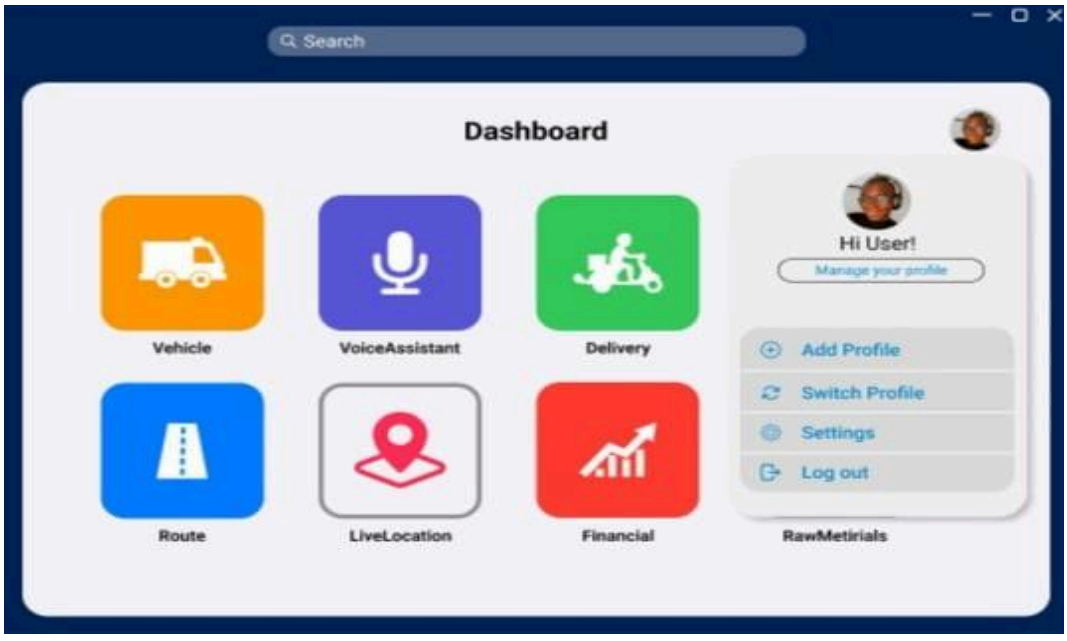
14. Automated Alerts and Notifications:
    - Objective: Provide automated alerts and notifications for critical events or deviations in the transport system.
    - Requirements:
        - Configurable alert settings for different scenarios.
        - Immediate notifications for delays or issues.
        - Alerts for route deviations or unexpected events.

15. Offline Mode:

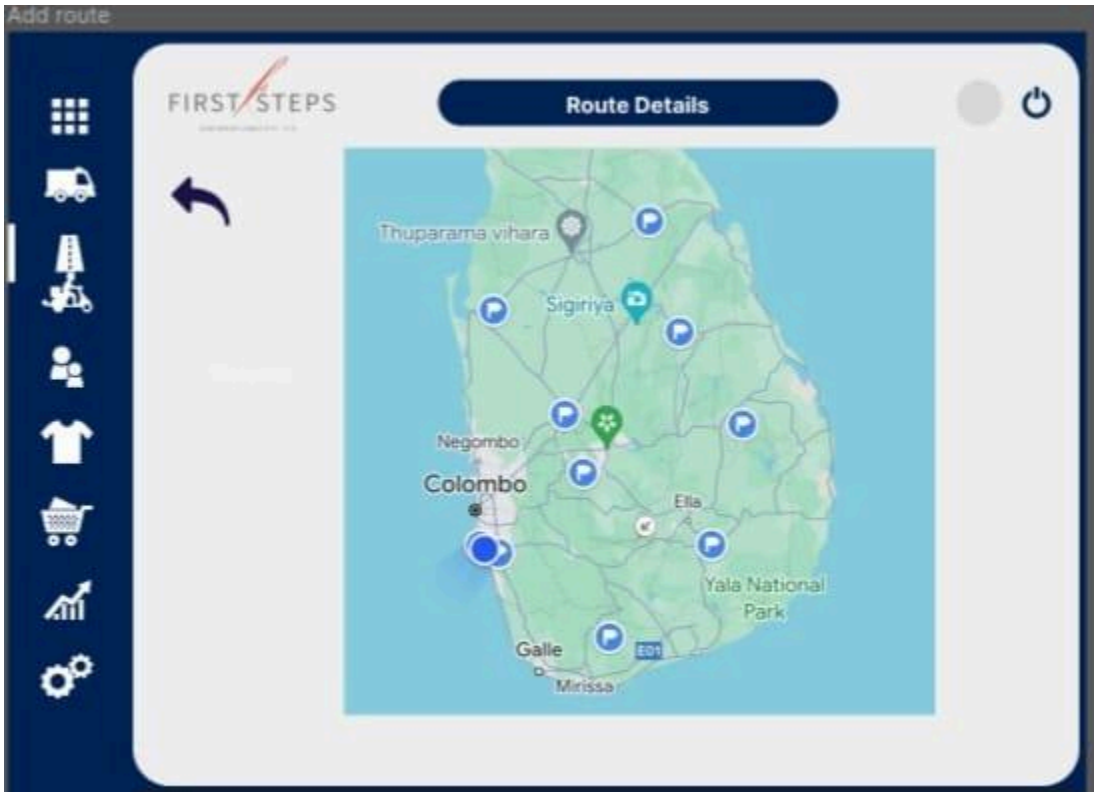
    - Objective: Ensure that the TMS can function in offline mode, allowing continuous operations in areas with limited connectivity.
    - Requirements:
        - Offline data storage and synchronization capabilities.
        - Automatic data upload when connectivity is restored.

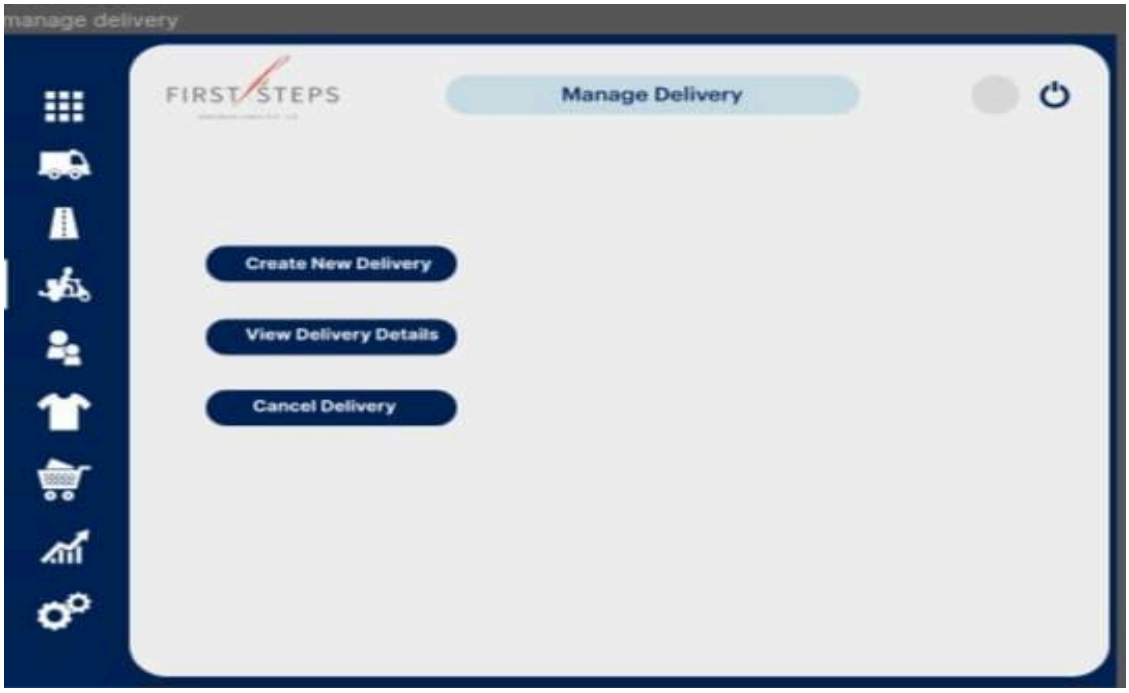
11. Mockup diagram for prototype

- **Sketch of each frames**




manage route
FIRST STEPS
Manage Route
Route Details
View Route Details
JV
LOCATION
DISTANCE FROM PANADURA
SISALU
MEDIRIGIRIYA
248KM
RISALKA
GALENBINDUNUWEWA
255KM
RICHLIGHT
RATHNAPURA
75KM
F&F
YATIYANTOTA
81KM
JUMEIRAH
MONARAGALA
316KM
DAG
MONARAGALA
298KM
DSL
KRANDENIYA
74KM
DAG
THALALLA
152KM
ISABELLAA
PALLEKELE
176KM
RITZ
NIKAWERATIYA
168KM
NORWOOD
POLGAHAWELA
104KM


Add route
FIRST STEPS
Route Details
Thuparama vihara
Sigiriya
Negombo
Colombo
Ella
Yala National Park
Galle
Mirissa
E01

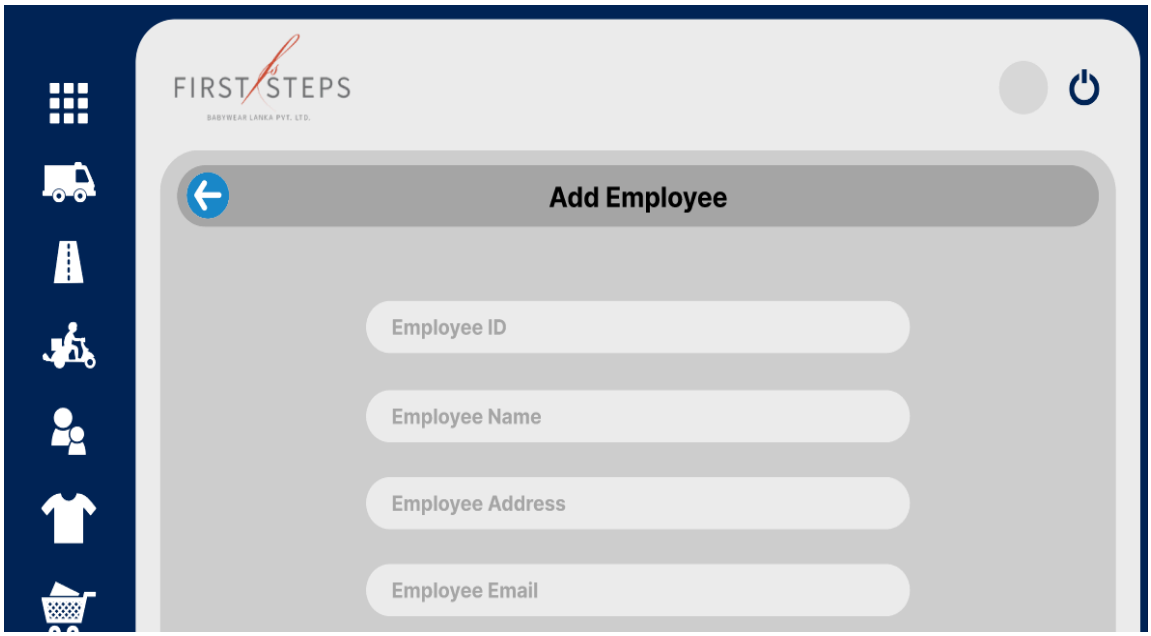

manage delivery
FIRST STEPS
Manage Delivery
Create New Delivery
View Delivery Details
Cancel Delivery

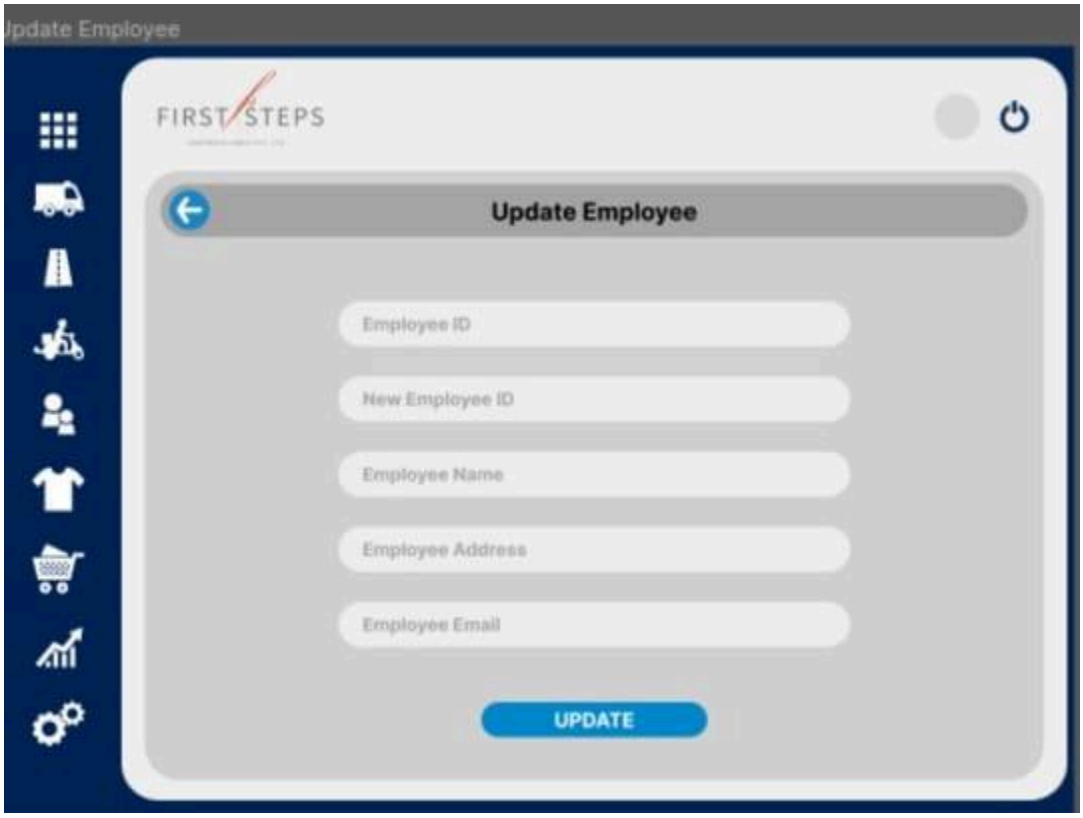

new delivery
FIRST STEPS
Create New Delivery
Select Driver
D001 L.G.Nimal Select
D002 M.Randika Select
D003 K.Supun Select
D004 W.E.Kamal Select
Delivery Id
Location
ADD


view delivery
FIRST STEPS
View Delivery Details

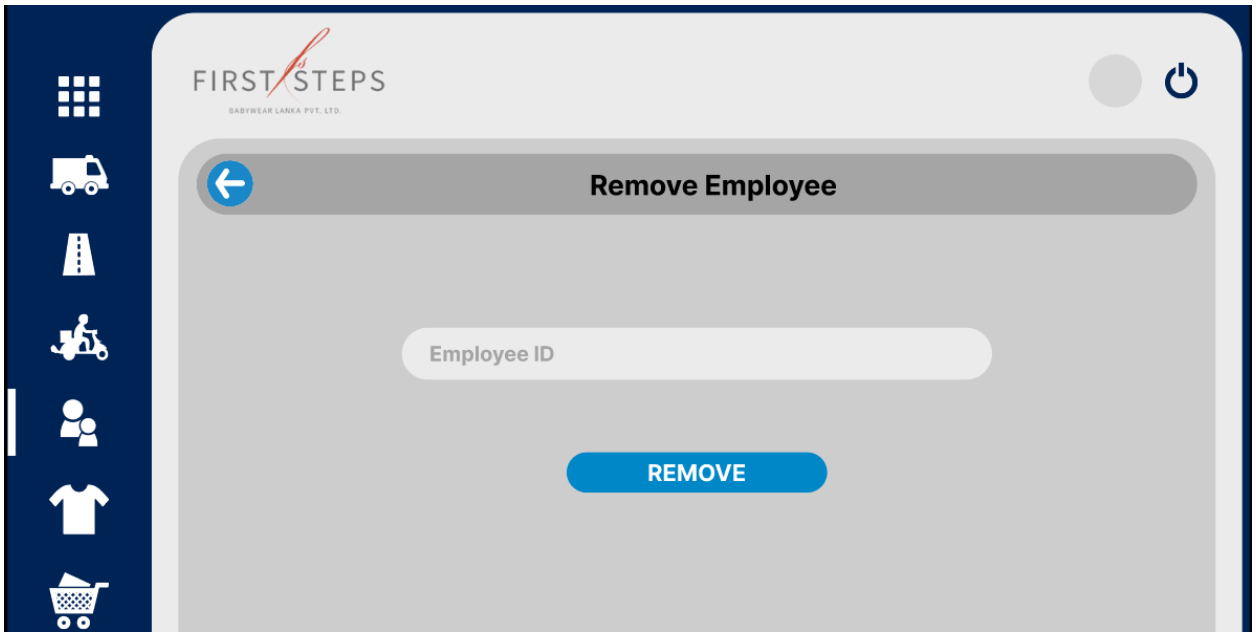

cancel delivery
FIRST STEPS
Cancel Delivery
Delivery Id
Cancel

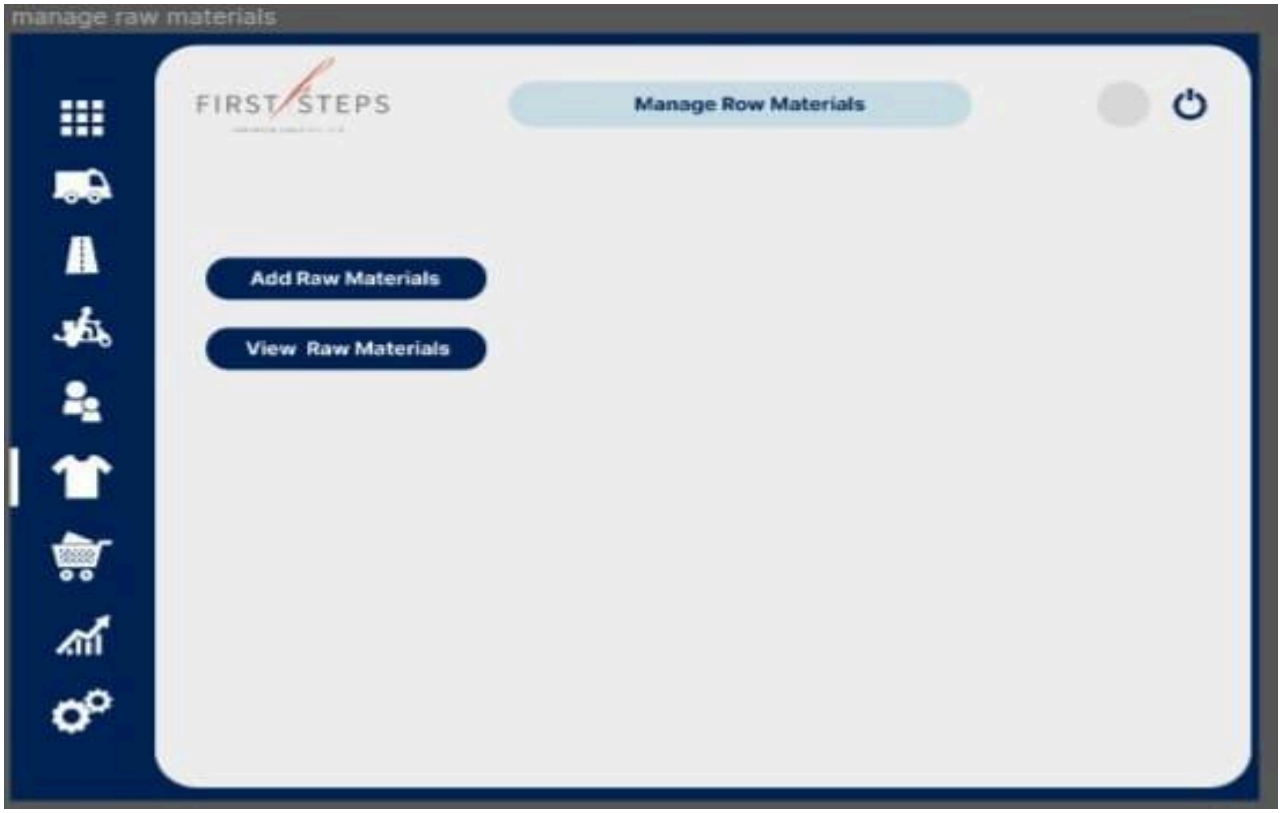

manage employee
FIRST STEPS
Manage Employee
Add Employee
Update Employee
View Employee
Remove Employee


FIRST STEPS
Add Employee
Employee ID
Employee Name
Employee Address
Employee Email

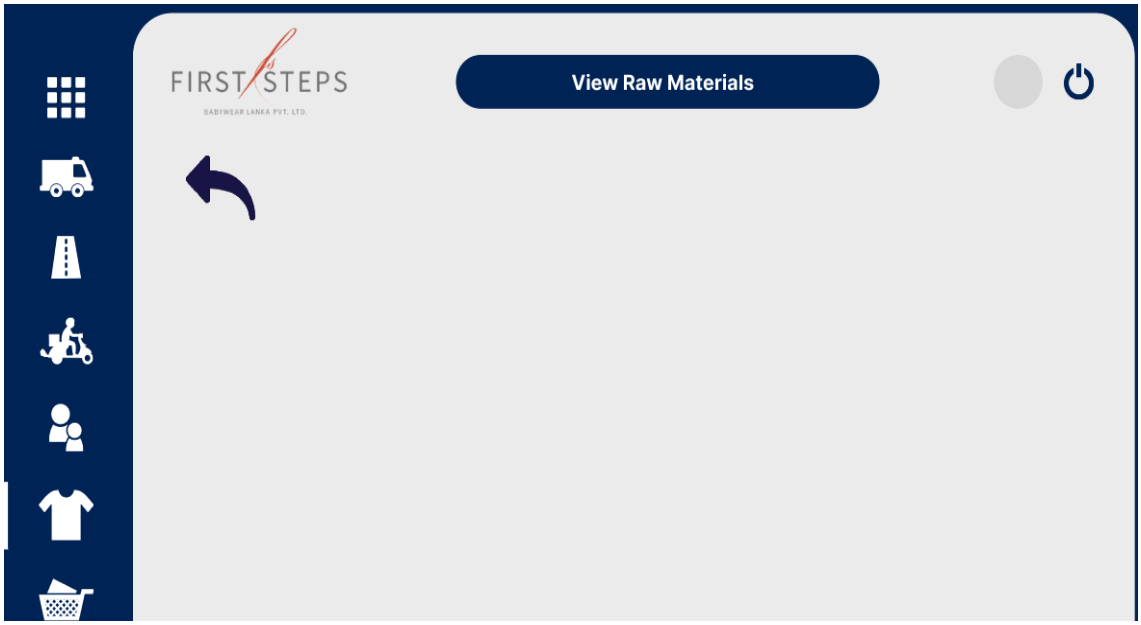

Update Employee
FIRST STEPS
Update Employee
Employee ID
New Employee ID
Employee Name
Employee Address
Employee Email
UPDATE

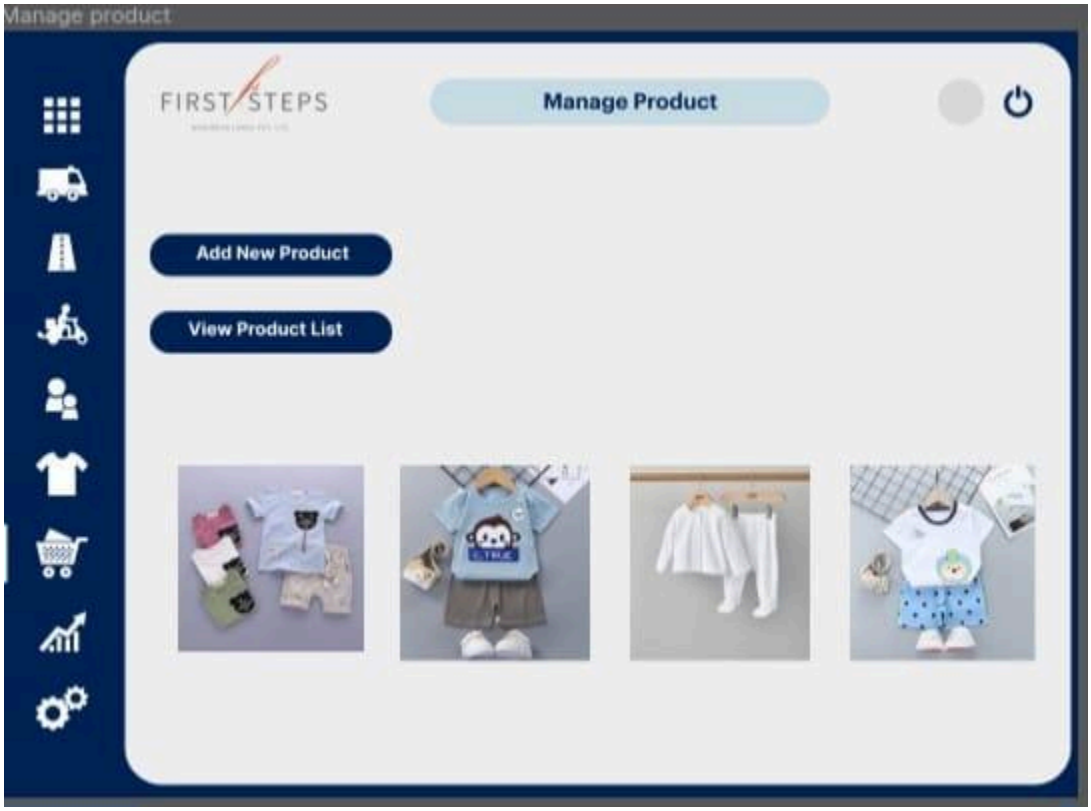

view Employee
FIRST STEPS
View Employee


FIRST STEPS
Remove Employee
Employee ID
REMOVE


manage raw materials
FIRST STEPS
Manage Row Materials
Add Raw Materials
View Raw Materials


add raw materials
FIRST STEPS
Add Raw Materials
Name :
Quantity:
Price :


FIRST STEPS
View Raw Materials



12. Implementation and development requirement

Implementing a Transport Management Software (TMS) involves translating the gathered requirements into a tangible, functional system. Below are the key implementation and development requirements for the TMS:

1. Technology Stack:

- Define the technology stack, including programming languages, frameworks, and databases.
- Ensure compatibility with existing infrastructure and technology used within the organization.

2. Database Design:

    - Design a robust database structure to store and manage data related to vehicles, routes, drivers, and transportation activities.
    - Ensure data integrity, normalization, and efficient retrieval for reporting.

3. User Interface (UI) and User Experience (UX):

    - Design an intuitive and user-friendly interface for various user roles (drivers, logistics personnel, management).
    - Prioritize ease of navigation, clear data visualization, and responsiveness across devices.

4. Real-Time Tracking System:

    - Integrate GPS technology for accurate real-time tracking of vehicles.
    - Implement map-based visualization to display live routes and locations.

5. Route Optimization Algorithm:

    - Develop an algorithm for route optimization based on factors such as traffic, distance, and delivery priorities.
    - Ensure adaptability to changing conditions and efficient rerouting options.

6. Communication Features:

    - Implement two-way communication between drivers and the central control system.
    - Integrate alerts and notifications for critical events, deviations, or delays.

7. Integration with Existing Systems:

    - Develop APIs for seamless integration with existing systems, such as inventory management and order processing.
    - Ensure data synchronization to maintain consistency across different platforms.

8. Driver Performance Monitoring:

    - Implement features to monitor driver behavior, including speed, breaks, and adherence to schedules.
    - Develop performance metrics dashboards for individual drivers.

9. Security Measures:

    - Incorporate robust security measures to protect sensitive transportation and logistics data.

- Implement secure login mechanisms, data encryption, and role-based access control

10. Reporting and Analytics:

- Develop customizable reporting features for monitoring key performance indicators (KPIs).
- Implement trend analysis and predictive modeling for informed decision-making.

11. Mobile Application for Drivers:

- Design and develop a mobile application for drivers with GPS navigation, logbook features, and communication tools.
- Ensure compatibility with common mobile platforms (iOS, Android).

12. Scalability and Performance:

- Design the system to be scalable to accommodate future growth.
- Conduct performance testing to ensure the system can handle peak loads and large datasets.

13. Training Module:

- Develop an online training module with tutorials and user guides for different user groups.

- Include certification or assessment mechanisms to ensure user proficiency.

14. Continuous Monitoring and Support:

- Implement monitoring tools for continuous tracking of system performance.
- Establish a support system to address technical issues and provide assistance to users.

15. Data Backup and Recovery:

- Develop a robust data backup and recovery system to prevent data loss in case of system failures.
- Implement regular backup schedules and testing procedures.

16. Compliance with Regulations:

- Ensure that the TMS complies with relevant regulations and standards in the transportation and logistics industry.
- Regularly update the system to address any changes in regulatory requirements.

17. Documentation:

- Maintain comprehensive documentation for the TMS, including technical specifications, user manuals, and system architecture.

- Provide documentation for future reference and training purposes.

By addressing these implementation and development requirements, First Steps Babywear Lanka can build a robust Transport Management Software that aligns with the needs of its logistics and transportation operations. Regular testing, user feedback, and updates are crucial throughout the development process to ensure the system's effectiveness and adaptability.

13. Running Environment requirement

Ensuring the TMS operates effectively requires attention to the hardware, software, and network environments. The following outlines the running environment requirements for the successful deployment and operation of the TMS:

1. Hardware Requirements:
    - Server Infrastructure:
        - Deploy dedicated servers to host the TMS, ensuring sufficient processing power, memory, and storage capacity.
        - Consider cloud-based solutions for scalability and flexibility.

- End-User Devices:
  - Ensure that end-user devices, such as desktop computers and laptops, meet minimum hardware specifications to run the TMS user interface smoothly.
  - Specify requirements for mobile devices (smartphones, tablets) if the TMS includes a mobile application.

- GPS Devices:
  - Ensure compatibility with GPS devices installed in vehicles.
  - Specify any hardware requirements for GPS devices, considering accuracy and real-time tracking capabilities.

2. Software Requirements:

- Operating Systems:
  - Specify compatible operating systems for server deployment (e.g., Windows Server, Linux distributions).
  - Ensure compatibility with common end-user operating systems (Windows, macOS, Linux) and mobile platforms (iOS, Android).

- Database Management System (DBMS):

        - Define the supported DBMS (e.g., MySQL, PostgreSQL, MongoDB) and version.
        - Ensure efficient database operations and data retrieval for TMS functionalities.

    - Web Browsers:
        - Specify supported web browsers for accessing the TMS interface.
        - Ensure compatibility with widely used browsers such as Google Chrome, Mozilla Firefox, and Microsoft Edge.

    - Mobile Application Compatibility:
        - Specify compatibility requirements for the mobile application, including supported operating systems and device specifications.

3. Network Infrastructure:

    - Internet Connectivity:
        - Ensure a reliable and high-speed internet connection for both server hosting and end-user access.
        - Specify bandwidth requirements for optimal performance.

    - Firewall and Security Measures:

        - Define firewall configurations to allow secure data transfer between the TMS and GPS devices.
        - Implement security measures to protect against unauthorized access and data breaches.

    - VPN (Virtual Private Network):
        - If applicable, define compatibility with VPNs for secure remote access.
        - Ensure that VPN configurations do not hinder TMS functionality.

4. Integration Requirements:

    - APIs and Middleware:
        - Specify integration requirements with other systems, applications, or middleware.
        - Define communication protocols and API specifications for data exchange.

5. Environmental Considerations:
    - Temperature and Humidity:
        - Specify recommended temperature and humidity levels for server rooms or data centers hosting the TMS servers.
        - Ensure that hardware components are within manufacturer-recommended environmental conditions.

    - Power Supply:
        - Implement redundant power supply systems to minimize downtime.
        - Specify power requirements and backup systems for uninterrupted TMS operation.

6. Security Measures:
    - Authentication and Authorization:
        - Define user authentication methods (e.g., username/password, multi-factor authentication).
        - Implement role-based access control to manage user permissions effectively.

    - Data Encryption:
        - Enforce data encryption measures to protect sensitive information during transmission and storage.
        - Specify encryption protocols and standards.

7. Scalability and Redundancy:

    - Scalability Requirements:
        - Plan for scalability by defining procedures for scaling up the infrastructure to accommodate increased load and data volume.
        - Implement load balancing mechanisms if needed.

- Redundancy and Failover:
  - Define redundancy requirements to ensure system availability in the event of hardware or network failures.
  - Implement failover mechanisms for critical components.

8. Monitoring and Logging:

- Monitoring Tools:
  - Implement monitoring tools to track system performance, resource utilization, and potential issues.
  - Define thresholds for alerts and notifications based on monitored metrics.

- Logging Requirements:
  - Specify logging mechanisms to record system events, user activities, and errors.
  - Define log retention policies and access controls for log files.

9. Backup and Recovery:

- Backup Strategy:
  - Establish a regular backup strategy for database and system configurations.
  - Define backup schedules, storage locations, and recovery procedures.

- Disaster Recovery Plan:

        - Develop a disaster recovery plan outlining procedures for recovering the TMS in case of catastrophic events.
        - Conduct periodic testing of disaster recovery mechanisms.

10. User Training and Support:

- Training Environment:
    - Create a training environment to familiarize users with the TMS interface and functionalities.
    - Specify training materials, documentation, and resources.

- Support Mechanisms:
    - Establish a support system, including helpdesk services, to address user queries and technical issues promptly.
    - Define support hours and escalation procedures.

By clearly defining these running environment requirements, First Steps Babywear Lanka ensures the smooth deployment, operation, and maintenance of the Transport Management Software. Regular monitoring, updates, and adherence to environmental specifications contribute to the long-term success of the TMS in enhancing the efficiency of the transport system.

14. Expected project results

The expected project results for the implementation of the Transport Management Software (TMS) at First Steps Babywear Lanka encompass various aspects of improved efficiency, enhanced control, and positive impacts on the company's transportation and logistics operations. Here are the anticipated project outcomes:

1. Optimized Transportation Processes:
    - The TMS is expected to optimize transportation processes by implementing efficient route planning, minimizing delays, and enhancing overall logistics coordination.

2. Real-Time Visibility and Control:
    - With the implementation of the TMS, real-time visibility into the location and status of vehicles will be achieved. This will empower the management team to have better control over transportation activities.

3. Improved Route Optimization:
    - The TMS's route optimization algorithm is anticipated to result in more cost-effective and time-efficient routes, reducing fuel consumption and transportation costs.

4. Enhanced Driver Performance:
    - The TMS's features for monitoring driver behavior and performance are expected to lead to improved adherence to schedules, increased safety, and overall better driver performance.

5. Effective Communication:
    - The TMS's communication tools are expected to facilitate seamless communication between drivers, logistics personnel, and management, leading to quicker responses to issues and improved collaboration.

6. Increased Customer Satisfaction:
    - By streamlining transportation processes and ensuring timely deliveries, the TMS is likely to contribute to increased customer satisfaction, reinforcing positive relationships with clients.

7. Accurate Data for Decision-Making:
    - The reporting and analytics features of the TMS are anticipated to provide accurate and actionable data, enabling data-driven decision-making for continuous improvement.

8. Cost Savings and Efficiency Gains:
    - Through optimized routes, improved driver performance, and better overall logistics coordination, the TMS is expected to contribute to cost savings and operational efficiency gains.

9. Scalability for Future Growth:
    - The TMS is designed to be scalable, allowing it to adapt to the company's future growth, increased data volumes, and additional requirements.

10. Compliance with Regulatory Standards:
    - The TMS is expected to ensure compliance with regulatory standards and requirements in the transportation and logistics industry, reducing the risk of penalties and legal issues.

11. Reduced Delays and Downtime:
    - The implementation of the TMS is anticipated to reduce delays in transportation activities, minimizing downtime and ensuring a more streamlined and predictable supply chain.

12. Enhanced User Proficiency:
    - The training module for the TMS is designed to enhance user proficiency, ensuring that employees can effectively utilize the system and maximize its benefits.

13. Improved Security Measures:

- The TMS's security measures, including data encryption and access controls, are expected to enhance the overall security of transportation and logistics data.

14. Positive Impact on Company Reputation:
    - Achieving operational efficiency and delivering on-time services is likely to positively impact the company's reputation, attracting potential customers and partners.

15. Continuous Improvement Culture:

    - The TMS is expected to contribute to the establishment of a culture of continuous improvement, where insights from data analytics and user feedback are used to refine and enhance transportation processes.

    - Overall, the successful implementation of the Transport Management Software is anticipated to bring about positive transformation, making First Steps Babywear Lanka's transportation and logistics operations more efficient, cost-effective, and adaptable to future challenges and growth.

12. Schedule

TASK SCHEDULE : -

Finished Tasks

	Task	Expected days to complete
01	Field visit	01 Day
02	Requirement gathering part	01 Days

- Planned Tasks

Schedule of the project

	Task	Expected days to complete
01	Prototype • UX/UI	02 Day
02	Frontend • Cording part	03 Days
03	Backend	03 Days

13.References:

https://firststepslanka.com/

14.Job role and budget

Budget for SRS(Software Requirements Specification)

This is the budget for software development project of First Steps Babywear Lanka

Reason	Explanation	Cost(LKR)
01.Project Manager	For Manage Project and Other Costs in his role.	Rs.400000.00
02.SSE	Coding and Leading Project and Other Costs in his role.	Rs325000.00
03.BA	Project Get Information and Other Cost in his role	Rs.150000.00
04.QA	Validation Project and Other Cost in his role	Rs.275000.00
05.Full-Stack Developers	Coding Project and Other Cost in his role	Rs.375000.00
06.Tech Lead	For Leading Project and Other Cost in his role	Rs.350000.00
08.UI/UX	For Designing Project and Other Costs in his role	Rs.200000.00
09.Front-End Developer	For Develop Project and Other Costs in his role.	Rs250000.00
10.Developers	For Develop Project and Other Costs in his role.	Rs.150000.00
11.Back-End Developers	For Develop Project and Other Costs in his role.	Rs.400000.00
12.Internet Cost	1 Month Mobitel , Dialog WiFi Bill	Rs.5733
13.Intellij IDEA Ultimate	Software Buying Cost	Rs.25000.00
14.Adobe AI	Software Buying Cost	Rs.7526.00
Total Cost:		Rs.2913259.00