**National University of Computer &  
Emerging Sciences Karachi Campus**  
  
  
***Software Design Specification  
For  
Supplier Management System***

**Course: Software Design and Architecture  
 Section: BSE-4B  
  
 21K-3906 Aiman  
 21k-3914 Rania  
 21K-3907 Huda  
  
  
  
  
  
  
  
1. System Architecture**

**Presentation layer:**   
This layer will be responsible for the user interface of the system. It will include the design and implementation of various GUI components, such as menus, forms, and dialogs. The presentation layer will be implemented using Java Swing or JavaFX, which are GUI frameworks for Java.

**Application layer:** This layer will be responsible for the business logic of the system. It will include the implementation of algorithms and rules that govern the behavior of the system. The application layer will be divided into the following subsystems.

**Supplier management subsystem:**   
This subsystem will handle the functionalities related to managing suppliers, such as adding new suppliers, removing existing suppliers, updating supplier information, and viewing supplier details.

**Delivery management subsystem:**This subsystem will handle the functionalities related to managing deliveries, such as creating delivery reports, tracking delivery status, and updating delivery information.

**Payment management subsystem:**This subsystem will handle the functionalities related to managing payments, such as collecting payments from suppliers, issuing payments to suppliers, and generating payment reports.

**Data layer:**   
This layer will be responsible for the storage and retrieval of data used by the system. It will include the database schema, data access logic, and any data manipulation or transformation logic that may be required. The data layer will be implemented using MySQL as the DBMS and JDBC as the API for accessing the database.

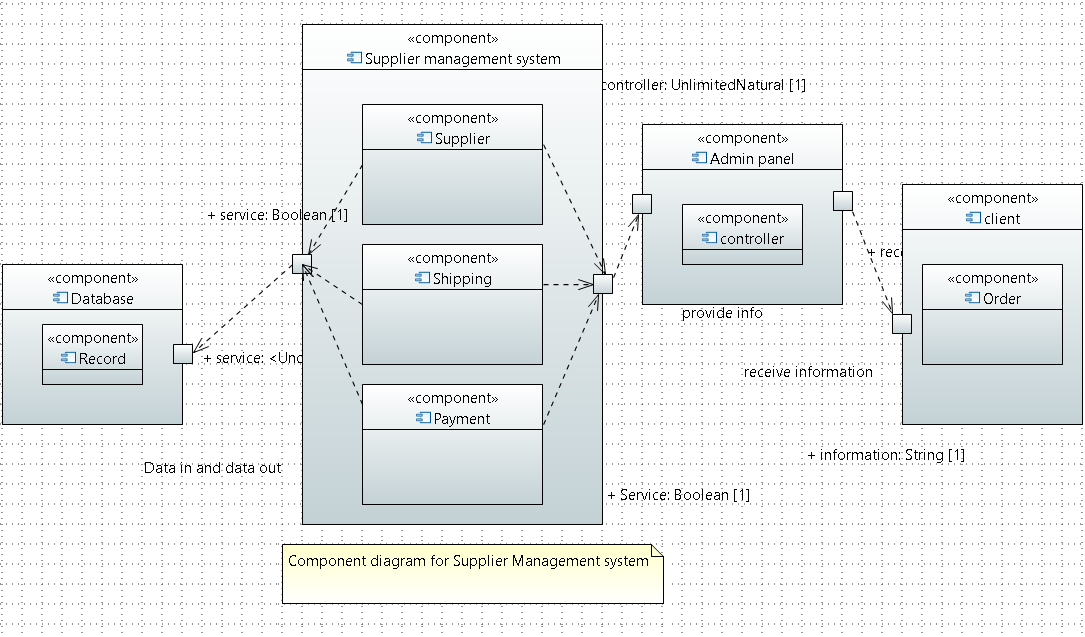
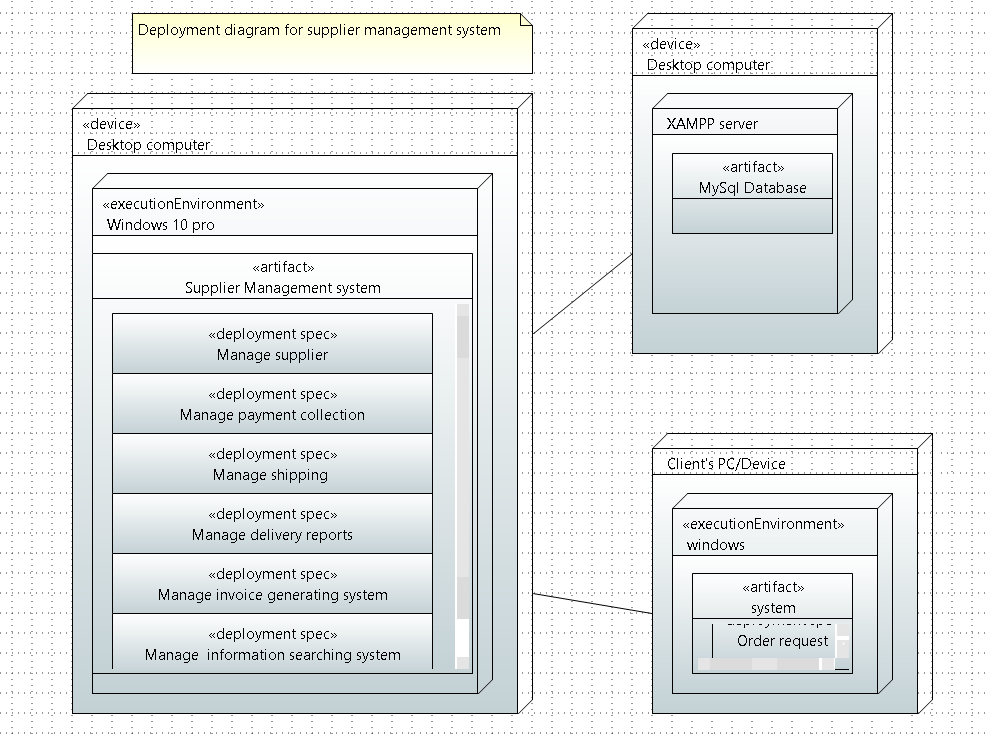
**Integration layer:**   
This layer will be responsible for managing the communication and interaction between the various subsystems and external systems. It will include messaging systems, APIs, or other mechanisms for exchanging data and information. In this case, the integration layer could be implemented using Java servlets, which provide a way to handle HTTP requests and responses between the client and server.

**Infrastructure layer:**   
This layer will be responsible for providing the underlying hardware and software infrastructure required to support the system. It will include servers, databases, network connections, and other resources. In this case, the infrastructure layer could be implemented using XAMPP, which provides an integrated web server, database server, and other tools for developing and testing web applications.

**1.1. System Level Architecture   
  
System Decomposition:**The supplier management system can be decomposed into several main components, including the Presentation layer, Application layer, Data layer, Integration layer, and Infrastructure layer.  
  
**Relationship between elements:**The Presentation layer will provide the user interface for the system, and will communicate with the Application layer. The Application layer will contain the business logic and will communicate with the Data layer to perform data operations. The Integration layer will manage communication between different subsystems and external systems, while the Infrastructure layer will provide the underlying hardware and software infrastructure.

**Interfaces to external systems:**The supplier management system will interface with external systems such as the database server and any other systems that might provide data to or consume data from the system.

**Physical design issues:**The Presentation layer will execute on the client-side, while the Application layer, Data layer, and Integration layer will execute on the server-side. The Infrastructure layer will provide the physical infrastructure to support the system.

**Global design strategies:**The supplier management system will use robust error handling strategies to handle errors at every level of the system. The system will also be designed to be scalable and easy to maintain, with modularity and abstraction built into the system design.  
  
  
**Component Diagram**  
  
In this diagram, the Admin Panel is at the top, the client component provides information to the Admin component. The Supplier Management Component is responsible for managing the supplier data, while the Shipping Component generates delivery reports based on that data. The Payment Component handles payment transactions and updates the payment status for each supplier, and the Database Component stores all the supplier data and payment information.  
  
  
  
  
  
**Deployment Diagram  
**

The deployment diagram in the supplier management system shows the physical deployment of the software components on the hardware nodes. The hardware nodes represent the physical machines or devices on which the software components are deployed.

The system has three main components: the client component, the server component, and the database component.

The client component is deployed on the user's device or computer, represented by the "Client Device" node. The client component provides the user interface and communicates with the server component to perform various actions such as adding, updating, or deleting a supplier.

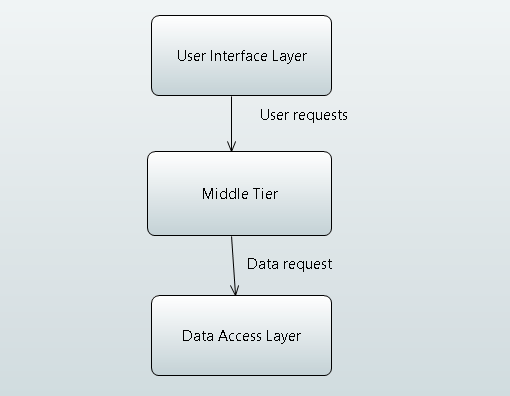
The server component is deployed on a separate machine or device, represented by the "Server Machine" node. The server component acts as a middleware between the client component and the database component. It receives requests from the client component, processes them, and sends the appropriate response back to the client component. The server component also manages the security and access control of the system.

The database component is deployed on a separate machine or device, represented by the "Database Server" node. The database component stores all the data related to the suppliers, delivery reports, and payment records. It receives requests from the server component to retrieve or update the data and sends the appropriate response back to the server component.

The deployment diagram also shows the communication paths between the components using various connectors such as client-server and database-server connectors. These connectors represent the protocols and communication mechanisms used by the components to exchange data and messages.

Overall, the deployment diagram provides a high-level view of how the software components are physically deployed and how they communicate with each other to provide the desired functionality of the supplier management system.  
 **1.2. Software Architecture  
  
User Interface Layer:**This layer is responsible for providing an interface for users to interact with the system. It includes the graphical user interface (GUI) and the user input/output processing logic. The User Interface Layer interacts with the Middle Tier layer.

**Middle Tier:**This layer is responsible for processing user requests, handling business logic, and interacting with the Data Access Layer to retrieve and store data. It also includes the application processing logic that sits between the User Interface and the Data Access Layers. The Middle Tier layer interacts with the User Interface Layer and the Data Access Layer.

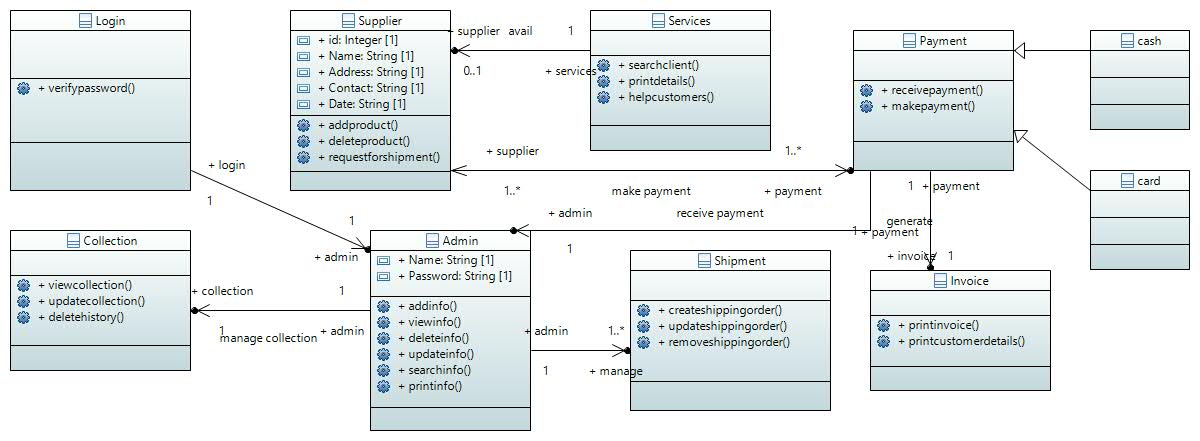
**Data Access Layer:**This layer is responsible for accessing and manipulating data from the underlying database. It includes the database connectivity logic and the database access processing logic. The Data Access Layer interacts with the Middle Tier.  
  
  
In this architecture, the User Interface Layer communicates with the Middle Tier layer through user requests. The Middle Tier processes the requests and interacts with the Data Access Layer to retrieve or store data from the database. The Data Access Layer is responsible for accessing the database and returning the requested data to the Middle Tier for processing. The processed data is then returned to the User Interface Layer for display to the user.  
  
  
***2. Design Strategy*** In designing the supplier management system, we aimed to create a flexible and scalable architecture that can accommodate future system extensions or enhancements. To achieve this, we used a modular approach that separates the different layers of the system, such as the user interface layer, the middle tier, and the data access layer.

One key decision was to use a Model-View-Controller (MVC) architecture for the user interface layer, which separates the presentation logic (View) from the business logic (Controller) and the data storage and retrieval (Model). This approach allows for easier maintenance and modifications of the user interface without affecting the underlying functionality of the system.

In terms of system reuse, we designed the system to be modular and reusable, with components that can be easily integrated into other systems as needed. We also prioritized the use of open-source technologies to ensure that the system can be easily adapted and modified by other developers.

For data management, we chose to use a relational database management system (RDBMS) with MySQL as the database engine. This allowed us to store and manage large amounts of data efficiently and with data consistency.

In terms of concurrency and synchronization, we implemented multi-threading to ensure that the system can handle multiple requests and users simultaneously without sacrificing performance. We also used synchronization mechanisms to prevent conflicts and maintain data integrity.

Overall, our design strategy aimed to create a flexible and scalable architecture that prioritizes modularity, reusability, and adaptability while ensuring efficient data management and user-friendly interface paradigms.  
  
***3. Detailed System Design   
  
3.1. Class Diagram***The class diagram consists of 10 classes, each of which plays a specific role in the system. The first class is the "Log In" class which allows the admin to log in to the system and perform various tasks. The "Admin" class is responsible for managing the supplier information. This includes adding, updating, and deleting supplier details.

The "Supplier" class enables the supplier to add or delete products and request shipment. The "Collection" class enables the admin to view collection history and update or delete collection records. The "Shipment" class is responsible for creating, removing, and updating shipment details.

The "Payment" class is responsible for handling payment-related tasks such as receiving payments. There are two classes that generalize the payment process - "Cash Payment" and "Card Payment". Finally, the "Invoice" class is responsible for generating invoices and printing customer details.

The attributes and functions of each class have been defined to ensure proper interaction between different modules. For example, the "Supplier" class can access the "Shipment" class to request shipment, while the "Admin" class can access the "Collection" class to view or update collection history.  
  
Overall, the class diagram provides a clear and comprehensive representation of the system's functionality, allowing developers to better understand how different components of the system interact with each other.  
  
**3.2. Detailed GUI Design**

This section describes the detailed design of the graphical user interfaces (GUIs) that will be implemented in the Supplier Management System. The GUIs are designed to provide a user-friendly and efficient way for the system users to interact with the system and perform the necessary tasks. The following subsections provide a detailed description of each GUI component and its functionality:

**Login GUI:**This GUI will be used to authenticate the users and grant them access to the system. It will have input fields for username and password and a submit button to log in.

**Dashboard GUI:** This GUI will be the main interface for the system users. It will display an overview of the system's functionalities and provide quick access to the most commonly used features such as managing suppliers, shipping, and payments.

**Supplier Management GUI:**This GUI will allow the admin to manage the suppliers in the system. It will have features to add new suppliers, update their information, view their details, and delete them if necessary.

**Shipping Management GUI:**This GUI will allow the admin to manage the shipping orders and related information. It will have features to create new shipping orders, generate delivery reports, update and delete existing orders, and manage shipping costs.

**Payment Management GUI:** This GUI will allow the admin to manage the payment-related tasks. It will have features to accept payments, generate invoices, track outstanding payments, view payment history, and print invoices.

**Report Generation GUI:**This GUI will allow the admin to generate various types of reports related to suppliers, shipping, and payments. It will have options to select the report type, date range, and other parameters as required.

Each GUI component will have a user-friendly layout, clear labels and instructions, and appropriate error handling mechanisms to ensure a smooth user experience. The GUIs will be implemented using modern design principles and best practices to ensure compatibility with various devices and platforms.  
**3.3. Database Design**

The supplier management system requires a database to store various information such as supplier details, shipping information, payment details, etc. The database schema consists of several tables with relationships between them.

The main tables in the database include the following:

**Suppliers:**   
This table stores all the details of the suppliers such as supplier ID, name, contact details, etc.

**Products:**   
This table stores the details of the products supplied by the suppliers such as product ID, name, description, etc.

**Orders:**   
This table stores the details of the orders placed by the customers such as order ID, date, customer ID, etc.

**Shipping:**This table stores the details of the shipping such as shipping ID, shipping date, shipping cost, etc.

**Payment:**This table stores the details of the payment such as payment ID, payment date, amount, etc.

The relationships between these tables are as follows:

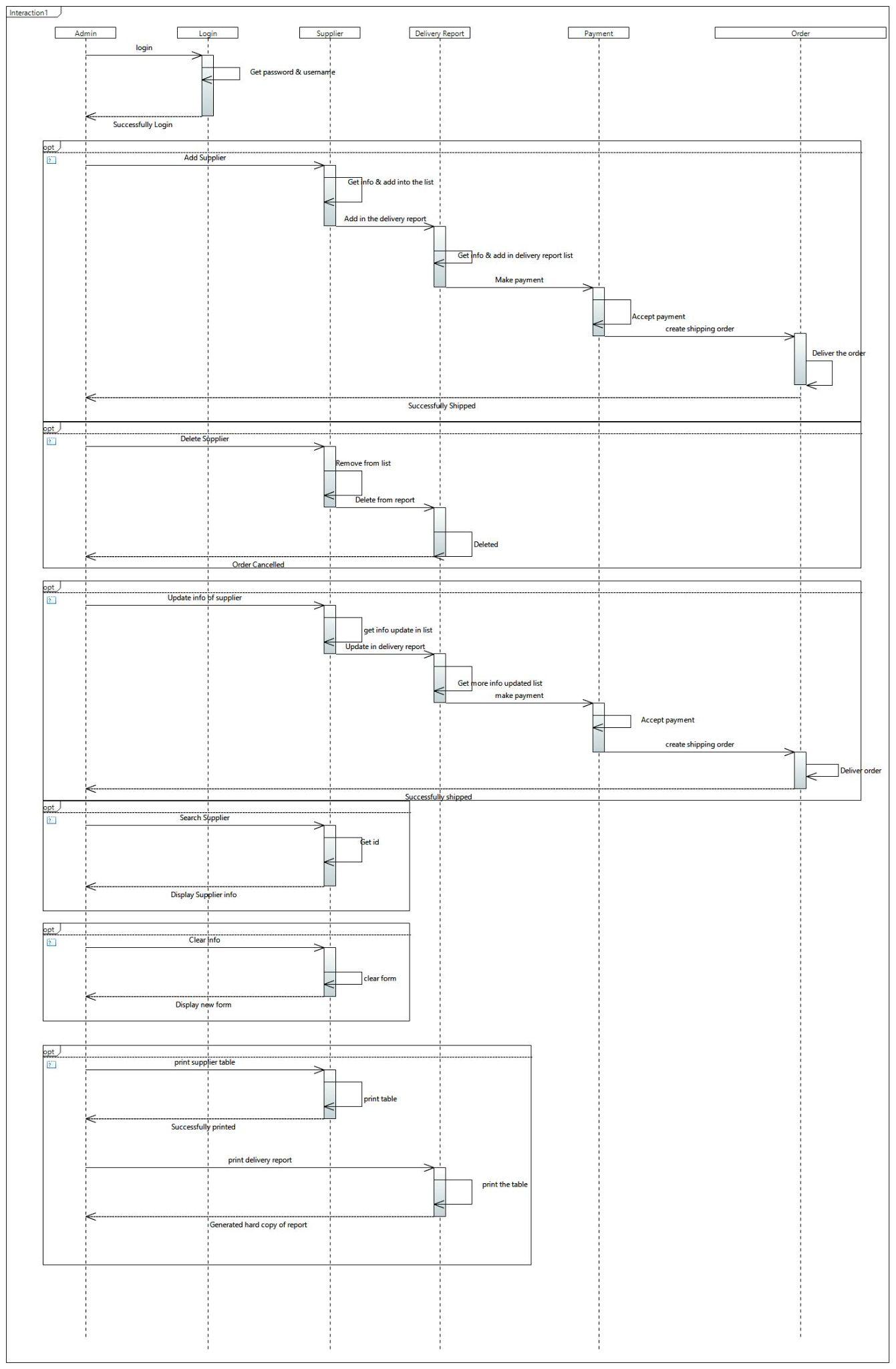
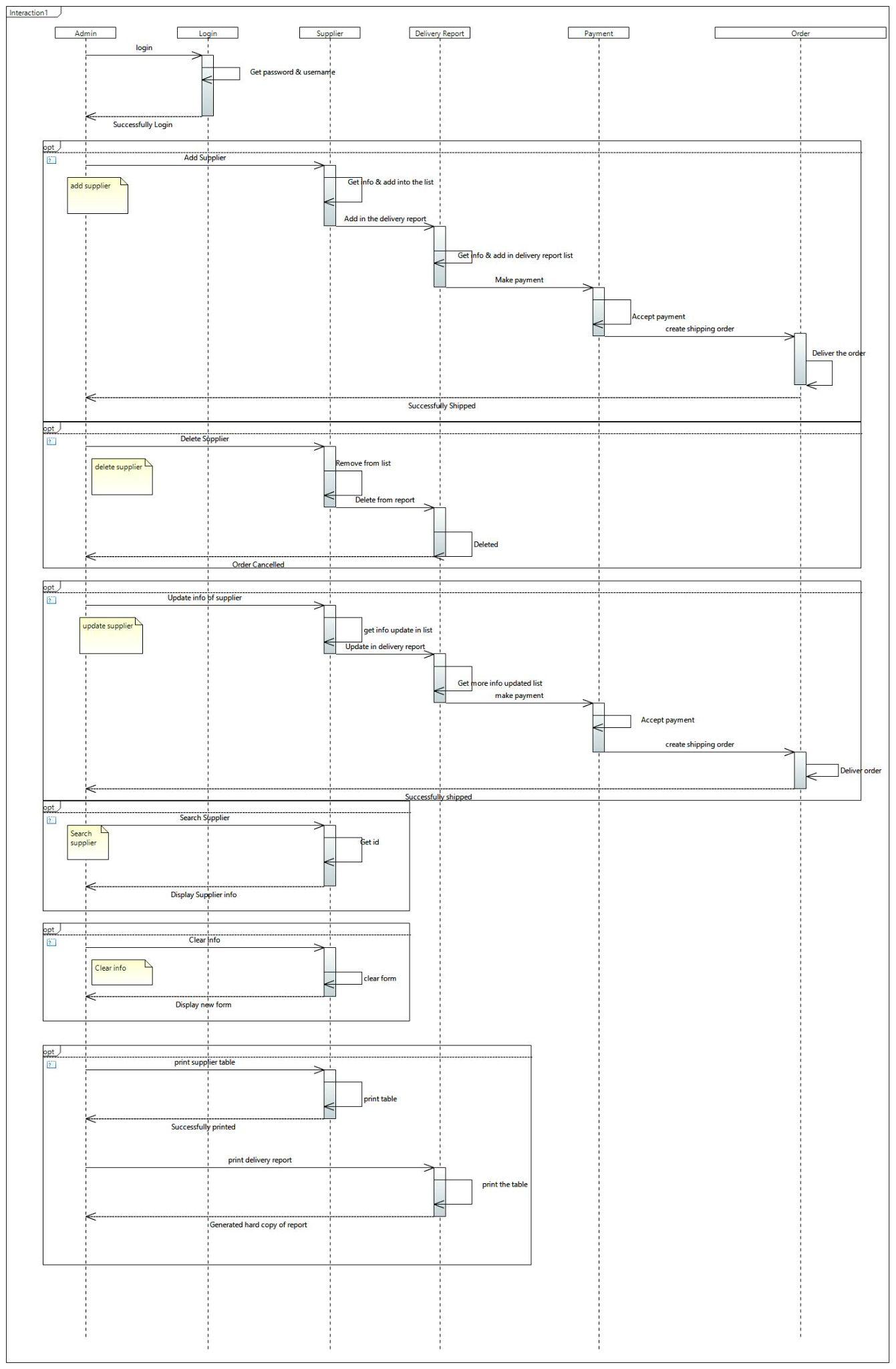
* Suppliers and Products have a one-to-many relationship as one supplier can supply many products.
* Orders and Products have a many-to-many relationship as one order can have many products and one product can be part of many orders.
* Orders and Customers have a one-to-many relationship as one customer can place many orders.
* Orders and Shipping have a one-to-one relationship as one order can have only one shipping record.
* Orders and Payment have a one-to-many relationship as one order can have many payments.

The data types and constraints for each table have been chosen based on the nature of the data they store. For example, the supplier ID is of type integer and is set as the primary key for the Suppliers table. The Orders table has a foreign key constraint on the Customers table to ensure that only valid customer IDs are entered.

Indexes have been created on certain columns for performance reasons. For example, the Products table has an index on the product name column to speed up searches for a particular product.

Data migration and backup strategies have also been implemented to ensure that data is not lost in case of any system failures or disasters. Regular backups are taken and stored securely.

Finally, security measures have been implemented to protect the data from unauthorized access. Access to the database is restricted to authorized personnel only and user authentication mechanisms are in place to prevent unauthorized access.  
  
**4. Application Design   
  
4.1. Sequence Diagram**

*  
***4.2. Sequence Diagram Explanation** The sequence diagram shows the interactions between the different objects involved in the system. The objects in this case are the admin, supplier, and delivery person.

The first sequence starts with the admin object creating a new supplier object. The admin provides the necessary details such as the supplier's name, contact information, and product details. This information is then passed to the supplier object, which is created and added to the list of suppliers.

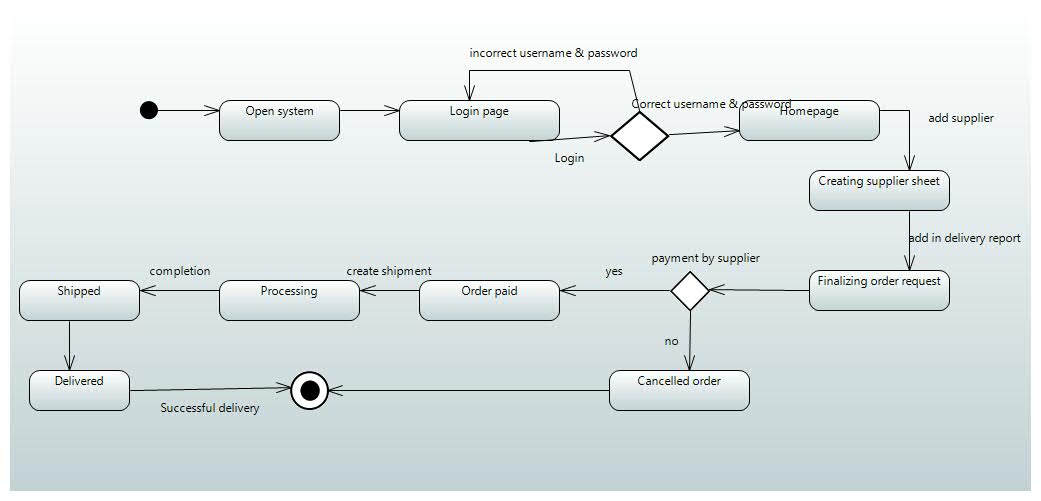
The next sequence shows the admin updating the information of an existing supplier. The admin selects the supplier whose information needs to be updated, makes the necessary changes and then saves the changes. The updated information is then passed to the supplier object, which updates its information accordingly.

The third sequence shows the admin removing a supplier from the system. The admin selects the supplier to be removed and then sends a request to the supplier object. The supplier object removes itself from the list of suppliers and sends a confirmation message back to the admin.

The fourth sequence shows the delivery person delivering the products to a specific supplier. The delivery person selects the supplier and then requests the supplier object to provide the necessary product details. The supplier object returns the product details, which the delivery person uses to make the delivery. Once the delivery is complete, the delivery person updates the status of the delivery in the supplier object.

The final sequence shows the admin generating a delivery report. The admin selects the supplier for which the report needs to be generated and then sends a request to the supplier object. The supplier object returns the necessary details, which are used by the admin to generate the report.

Overall, the sequence diagram provides a clear and detailed representation of the interactions between the different objects in the system, allowing for a better understanding of the system's functionality.  
  
  
  
**4.3. State Diagram**

****4.4. State Diagram Explanation**The state transition diagram for the supplier management system represents the different states of the supplier's information and the transitions between them. It consists of four states:

New: This state represents the initial state of the supplier's information when it is first added to the system.

Pending Approval: This state represents the supplier's information that is waiting for approval from the admin before it can be used in the system.

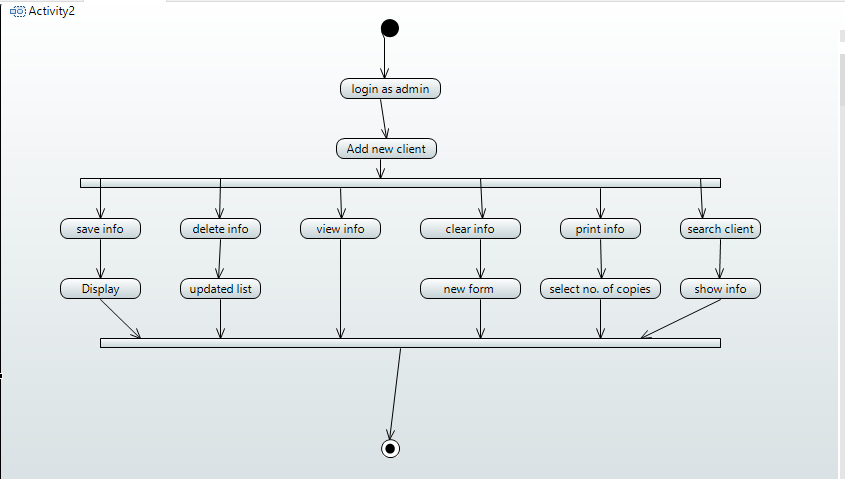
Approved: This state represents the supplier's information that has been approved by the admin and can now be used in the system for ordering products and managing shipments.

Rejected: This state represents the supplier's information that has been rejected by the admin and cannot be used in the system.

The transitions between these states are triggered by certain events or actions. For example, when a supplier is added to the system, it transitions from the New state to the Pending Approval state. Similarly, if the admin approves the supplier's information, it transitions from the Pending Approval state to the Approved state. If the admin rejects the supplier's information, it transitions from the Pending Approval state to the Rejected state.

Overall, the state transition diagram provides a clear understanding of the various states and transitions that the supplier's information can go through in the system. This helps in ensuring that the supplier management process is well-defined and streamlined.  
  
**4.5. Activity Diagram**

***Manage supplier***

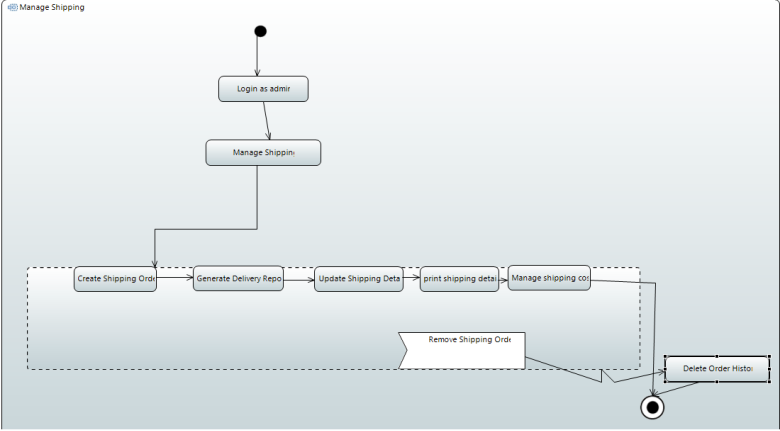
  
  
**4.5.1. Activity Diagram Explanation**  
The activity diagram starts with the admin selecting the "Manage Supplier" option from the main menu. This action triggers the "Manage Supplier" activity. The first activity in this process is to "View Supplier Information". The admin can choose to view either all the supplier information or search for specific supplier information using the "Search Supplier" activity.

Once the admin has viewed the supplier information, they can perform different operations such as "Add Supplier", "Update Supplier Information", or "Delete Supplier". These activities can be performed by selecting the corresponding options from the "Supplier Information" activity.

If the admin selects the "Add Supplier" activity, they will be prompted to enter the details of the new supplier such as name, address, contact information, etc. Once the admin has entered all the details, the new supplier information is added to the supplier database.

If the admin selects the "Update Supplier Information" activity, they will be prompted to select the supplier they want to update and then edit the necessary information. Once the admin has made the changes, the updated supplier information is saved in the database.  
If the admin selects the "Delete Supplier" activity, they will be prompted to select the supplier they want to delete. Once the admin has selected the supplier, the system will delete the supplier information from the database.

Finally, after performing any of the above activities, the admin can choose to "Save Supplier Information" to save any changes made to the supplier database. The "Manage Supplier" activity is then completed, and the admin is returned to the main menu.  
  
  
**4.6. Activity Diagram**   
 ***Manage shipping***

  
  
**4.6.1. Activity Diagram Explanation**  
The activity diagram starts with the "Manage Shipping" initial node, which leads to the "Create Shipping Order" activity. The admin creates a new shipping order by entering the relevant information such as the supplier name, shipping address, delivery date, and other details. The system then generates a unique shipping order ID and saves the order in the database.

The next activity is "Generate Delivery Report," which is initiated by the admin after the shipping order has been delivered to the customer. The admin enters the shipping order ID and the delivery date, and the system generates a delivery report, which contains information about the order, such as the delivery date, the recipient's name, and signature, if required.

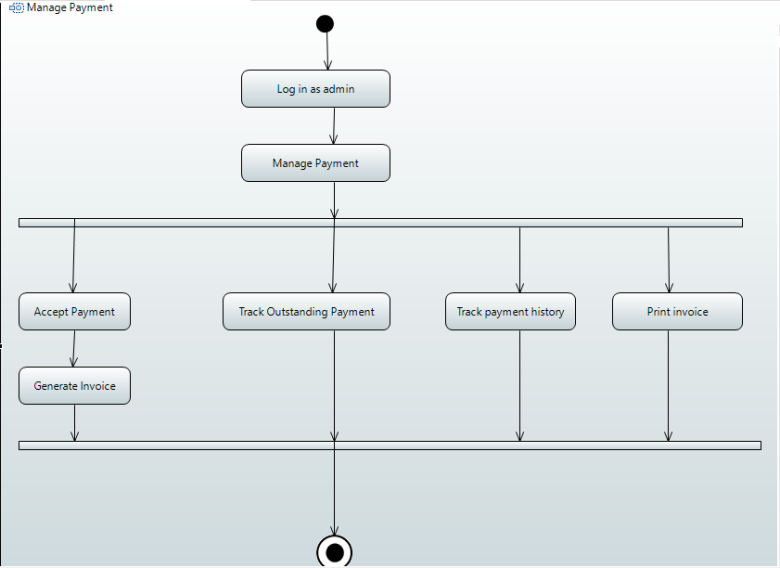
The next activity is "Update Shipping Information," which allows the admin to update any information related to a specific shipping order, such as the delivery address, the delivery date, or the recipient's name. The admin enters the shipping order ID and selects the option to update the relevant information. The system updates the order information in the database.

The "Delete Shipping Information" activity allows the admin to delete a shipping order. The admin enters the shipping order ID and selects the option to delete the order. The system removes the order from the database.

The "Save Shipping Information" activity allows the admin to save any changes made to a specific shipping order. The admin enters the shipping order ID and selects the option to save the changes. The system updates the order information in the database.

The "Print Shipping Details" activity allows the admin to print a report containing all the details of a specific shipping order, such as the shipping order ID, supplier name, delivery address, delivery date, and recipient name. The admin enters the shipping order ID and selects the option to print the report. The system generates a report and displays it on the screen, which the admin can then print.

Finally, the "Manage Shipping Cost" activity allows the admin to manage the shipping cost for each order. The admin can view the shipping cost for each order and update it if necessary. The system retrieves the shipping cost from the database and displays it to the admin. The admin can then update the shipping cost and save it in the database.

The activity diagram ends with the "Manage Shipping" final node.  
  
  
**4.7. Activity Diagram**   
 ***Manage payment  
*4.7.1. Activity Diagram Explanation**The activity diagram for managing payment shows the steps involved in managing payment for the supplier management system. The process begins with the acceptance of payment, which can be done through various payment methods such as cash, check, or credit card. Once payment is accepted, the system generates an invoice that includes details of the payment amount and payment method. The invoice is then sent to the supplier or customer for payment.

The system then tracks the outstanding payment, which is any payment that has not been received or cleared by the due date. The system sends reminders to the supplier or customer to make the payment, and the payment status is updated accordingly.

The payment history is also tracked, which includes all the payments made by the supplier or customer. This information is important for record-keeping and can be used for future reference.

Finally, the system can print the invoice for the supplier or customer as proof of payment. The activity diagram also shows the various decision points in the process, such as whether payment has been received or not, and whether the payment is overdue or not. This helps the system to take appropriate actions and keep track of the payment process for efficient managemen***t.  
  
  
5. References***

* Smith, J. (2018). Supplier Management Best Practices. New York: McGraw Hill Education.
* Martin, R. (2015). Clean Architecture: A Craftsman's Guide to Software Structure and Design. Upper Saddle River, NJ: Prentice Hall.
* Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1995). Design Patterns: Elements of Reusable Object-Oriented Software. Reading, MA: Addison-Wesley.
* Oracle Corporation. (2021). Java Platform, Standard Edition 16 Documentation. Retrieved from https://docs.oracle.com/en/java/javase/16/
* PostgreSQL Global Development Group. (2021). PostgreSQL 13.4 Documentation. Retrieved from https://www.postgresql.org/docs/13/
* Papyrus. (2021). User Guide. Retrieved from https://www.eclipse.org/papyrus/ug-distribution/PapyrusUsersGuide.html  
    
    
  ***6. Appendices***Attaching the use cases for supplier management system for the better understanding of the system.  
  