Database Systems

Supply Chain Management System

WOOD
SUPPLY CHAIN
SYSTEM

Prepared By

Name	ID
Mohamed Shaban	320230188
Sandy Feras	320230158
Manal Mahmoud	320230164
Omar Ahmed	320230172

1. System Overview

The **Wood Supply Chain System** is a cloud-enabled, multi-tier supply chain management platform designed to manage the complete lifecycle of wood products from forest harvesting to customer delivery. The system supports inventory control, order processing, transportation management, compliance monitoring, and reporting.

WSCS is accessible via web browsers and mobile devices, enabling real-time collaboration among stakeholders such as forest managers, suppliers, warehouse staff, transport partners, and customers.

Key Capabilities

- Real-time stock visibility based on:
  - Wood species
  - Size and moisture level
  - Batch/lot number
  - Certification status
- Dedicated customer and supplier portals
- Integration with existing enterprise systems
- Role-based access and secure authentication

2. Stakeholders and Responsibilities

Stakeholder	Responsibilities
Admin (Staff)	Manage users, permissions, system configurations, roles, audit logs, and overall system monitoring
Forest Manager	Oversee forest zones, manage harvesting schedules, assign field workers, and maintain forest data

Transport Manager	Manage transport operations, drivers, trucks, fuel tracking, and delivery route optimization
Suppliers / Vendors	Provide raw or semi-processed wood materials; manage contracts, compliance, and deliveries
Sawmill Operator	Process harvested wood into lumber or finished goods; manage production orders and quality checks
Warehouse Manager	Manage storage, inventory, and warehouse operations including item tracking and stock alerts
Logistics Partners	Collaborate on transport planning, shipments, and last-mile delivery
Sales Officer	Manage customer sales orders, invoices, and customer relationships
Customer / Retailer	Browse product catalog, place orders, track deliveries, and view invoices or payment history

3. System Requirements

3.1 Functional Requirements

1. Forest Management

Actors: Forest Manager, Admin

- Add, edit, and remove forest areas with geo-location, tree species, and ownership data
- Schedule and monitor harvesting activities per forest zone
- Assign forest managers and workers to specific forest areas

2. Harvesting Operations

Actors: Forest Manager

- Create and track harvest batches by tree type, quantity, and date
- Generate unique batch IDs with QR codes for traceability
- Record harvest volumes, quality indicators, and harvest dates

3. Supplier Management

Actors: Admin, Forest Manager, Sales Officer

- Register, approve, and categorize suppliers
- Maintain supplier profiles, contracts, pricing, and contact details
- Rate suppliers based on delivery performance and quality
- View supplier compliance, history, and performance reports

4. Procurement and Purchase Orders

Actors: Admin, Sales Officer

- Create, approve, and track purchase orders
- Support multiple items, quantities, pricing, and delivery deadlines
- Track order lifecycle (Pending → In Progress → Delivered)
- Generate purchase summaries and cost breakdowns

5. Transportation Control

Actors: Transport Manager, Logistics Partner

- Register transport companies, trucks, and drivers
- Assign drivers and trucks using GIS-based route planning
- Monitor shipment status and capture proof of delivery
- Track fuel usage, trip logs, and driver performance

6. Sawmill and Processing Operations

Actors: Sawmill Operator

- Manage sawmill facilities and processing units
- Create processing orders to convert logs into finished goods
- Track input/output quantities, timelines, and efficiency
- Log machine usage and manpower allocation

7. Quality and Waste Management

Actors: Sawmill Operator, Quality Staff

- Conduct quality inspections for processed wood
- Record inspection results, notes, and certifications
- Track waste volume, type, and recycling methods
- Analyze waste percentage and efficiency

8. Machine Maintenance

Actors: Sawmill Operator, Admin

- Schedule preventive and corrective maintenance
- Record maintenance activities, parts used, and costs
- Generate downtime reports and alerts

9. Warehouse Management

Actors: Warehouse Manager

- Manage warehouses, capacity, and product placement
- Track shelf locations and stock levels
- Configure low-stock alerts
- Support barcode and QR code scanning

10. Inventory Management

Actors: Warehouse Manager, Admin

- Track incoming, outgoing, and damaged stock
- Maintain batch-based inventory control
- Generate stock valuation reports

11. Distribution and Shipment Tracking

Actors: Transport Manager, Logistics Partner

- Plan shipments from warehouses to customers
- Assign drivers, trucks, and routes
- Track shipment lifecycle
- Generate delivery documents and feedback reports

12. Customer and Retailer Portal

Actors: Customer / Retailer

- Register and log in securely
- Browse product catalog and pricing
- Place orders and track deliveries
- Access invoices and payment history

13. Sales Order Management

Actors: Sales Officer

- Create and manage customer orders
- Apply taxes, discounts, and delivery costs automatically
- Track order lifecycle
- Generate sales reports

14. Invoicing and Billing

Actors: Sales Officer, Admin

- Automatically generate invoices
- Support multi-currency and multi-tax formats
- Email invoices or provide downloadable PDFs
- Track unpaid and overdue invoices

15. Payment Management

Actors: Admin, Sales Officer

- Record online and manual payments
- Integrate with payment gateways
- Link payments to invoices
- Generate revenue and transaction reports

16. Employee and Role Management

Actors: Admin

- Manage employees, roles, and permissions
- Define access levels per system module
- Track employee responsibilities and performance

17. User Authentication and Access Control

Actors: Admin

- Implement JWT-based authentication
- Support two-factor authentication
- Manage password resets and session tracking
- Audit access logs

18. Audit Logs and System Activity

Actors: Admin

- Track logins, CRUD actions, approvals, and configuration changes
- Filter logs by user, department, or date
- Generate compliance audit reports

19. Reporting and Analytics Dashboard

Actors: Admin, Executives

- Unified KPI dashboard including:
    - Supplier performance
    - Inventory turnover
    - Processing efficiency
    - Sales and revenue trends
- Drill-down analytics for decision-making

3.2 Non-Functional Requirements

- **Performance:**

The system must handle high transaction volumes with fast response times. Real-time dashboards and queries should not be affected by background processing.

- **Scalability:**

The architecture must scale horizontally to support business growth, additional warehouses, partners, and increased demand.

- **Availability and Reliability:**

The system should provide high availability with minimal downtime.

- **Usability:**

The interface must be intuitive and support both Arabic (RTL) and English. Clear workflows, tooltips, and training materials should assist non-technical users.

- **Maintainability:**

The codebase should follow modular design principles and modern frameworks to allow easy updates and enhancements.

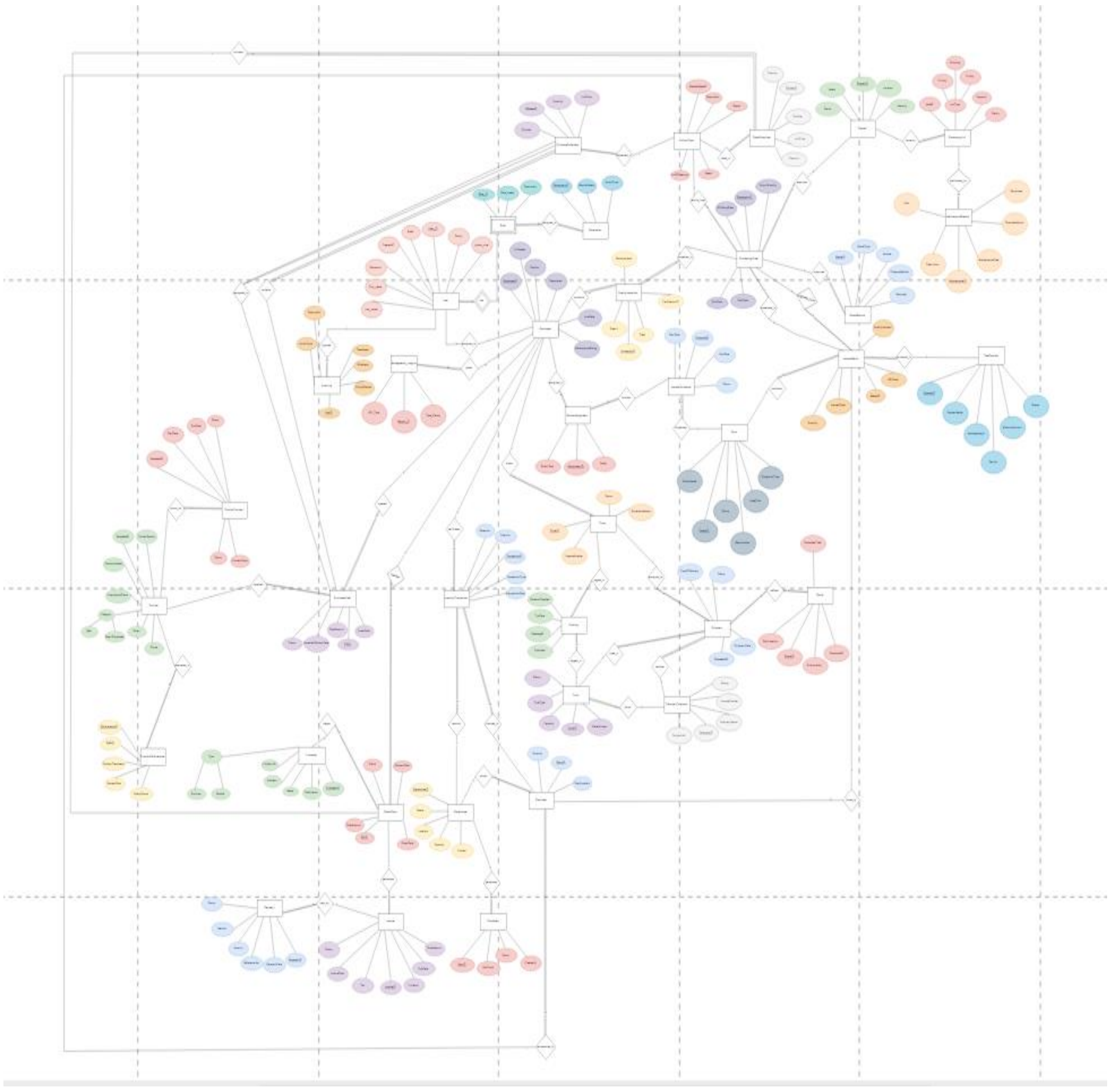
- **Reliability:**

Automatic failover, regular backups, and scheduled maintenance notifications must be supported.

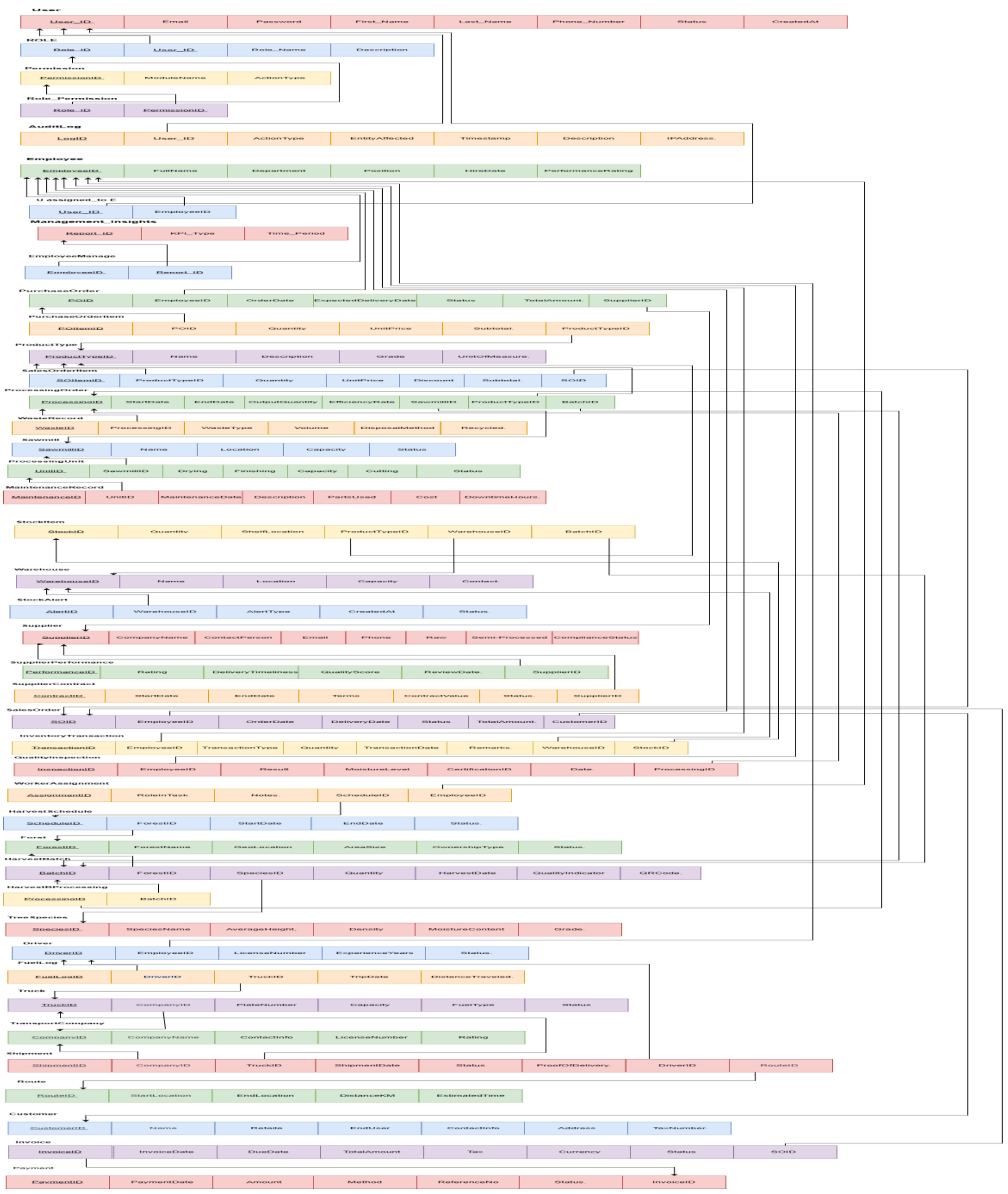
- **Localization and Context:**

WSCS is adapted for Egypt, using metric units, Arabic language support, local time zones, the Sunday–Thursday workweek, and consideration of public holidays and road restrictions.

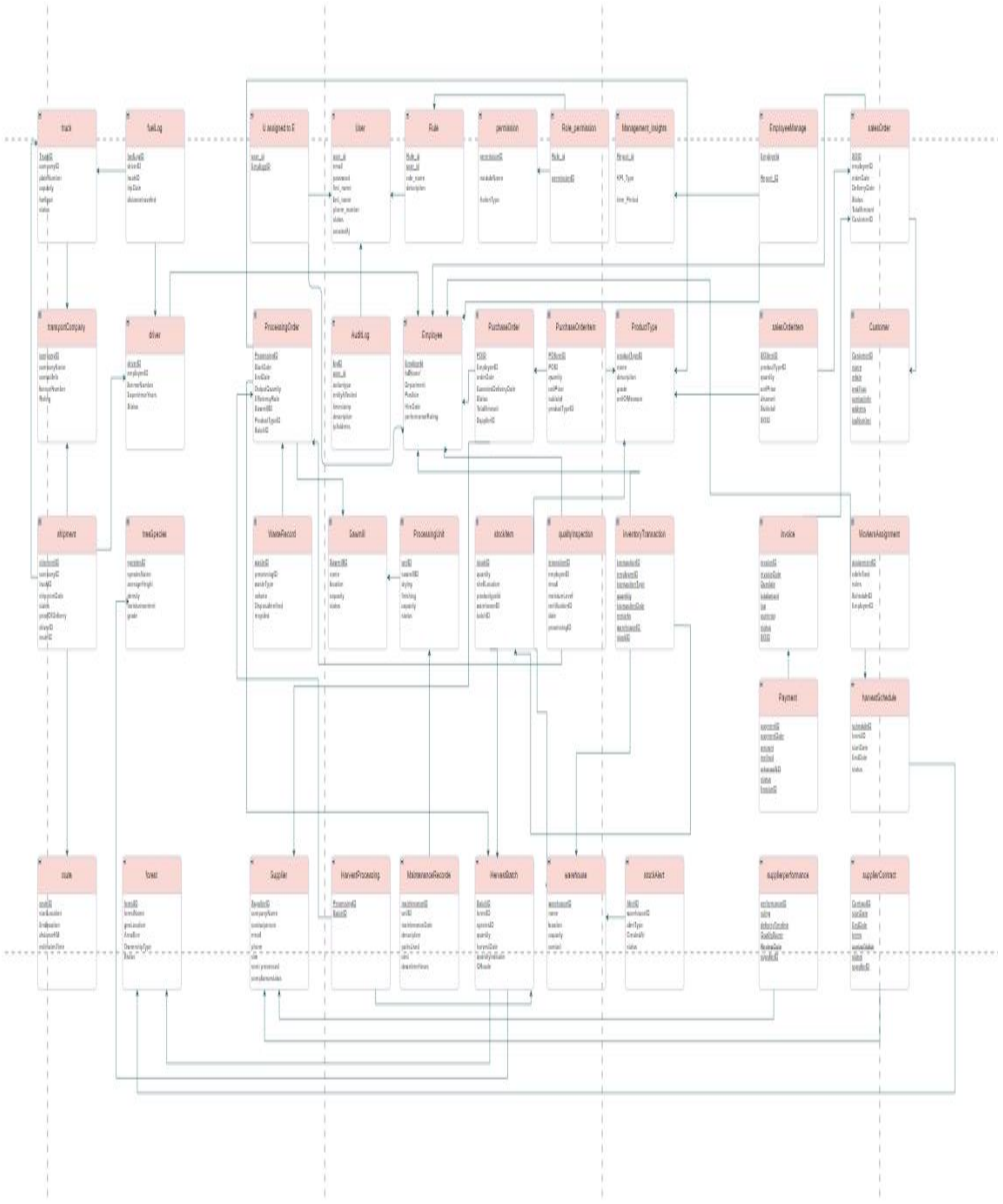
4. Entity-Relationship Diagram (ERD)



5. Database Mapping



6. Schema



7. Database Implementation & Queries

7.1 Database Schema Implementation (DDL)

The following SQL statements represent a sample of the database implementation and are provided for illustration purposes only:

7.1.1 User Table

```
CREATE TABLE "User" (

   User_ID SERIAL PRIMARY KEY,

   Email VARCHAR(255) UNIQUE NOT NULL,

   Password VARCHAR(255) NOT NULL,

   First_Name VARCHAR(100),

   Last_Name VARCHAR(100),

   Phone_Number VARCHAR(20),

   Status VARCHAR(50) DEFAULT 'active',

   CreatedAt TIMESTAMP DEFAULT CURRENT_TIMESTAMP

);
```

7.1.2 HarvestBatch Table

```
CREATE TABLE HarvestBatch (

   BatchID SERIAL PRIMARY KEY,

   ForestID INTEGER REFERENCES Forest(ForestID) ON DELETE SET NULL,

   SpeciesID INTEGER REFERENCES TreeSpecies(SpeciesID) ON DELETE SET NULL,

   ScheduleID INTEGER REFERENCES HarvestSchedule(ScheduleID) ON DELETE SET
NULL,

   Quantity DECIMAL(10,2) NOT NULL,

   HarvestDate DATE,

   QualityIndicator VARCHAR(50),
  QRCode VARCHAR(200) UNIQUE
);
```

7.1.3 Shipment Table

```
CREATE TABLE Shipment (
  ShipmentID SERIAL PRIMARY KEY,
  SOID INTEGER REFERENCES SalesOrder(SOID) ON DELETE SET NULL,
  TruckID INTEGER REFERENCES Truck(TruckID) ON DELETE SET NULL,
  DriverID  INTEGER REFERENCES Driver(DriverID)  ON DELETE SET NULL,
  CompanyID INTEGER REFERENCES TransportCompany(CompanyID) ON DELETE SET
NULL,
  RouteID INTEGER REFERENCES Route(RouteID) ON DELETE SET NULL,
  ShipmentDate DATE,
  Status VARCHAR(50),
  ProofOfDelivery  TEXT
);
```

7.1.4 QualityInspection Table

```
CREATE TABLE QualityInspection (
  InspectionID SERIAL PRIMARY KEY,
  EmployeeID INTEGER REFERENCES Employee(EmployeeID) ON DELETE SET NULL,
  ProcessingID INTEGER REFERENCES ProcessingOrder(ProcessingID) ON DELETE SET
NULL,
  POItemID INTEGER REFERENCES PurchaseOrderItem(POItemID)  ON DELETE SET NULL,
  BatchID INTEGER REFERENCES HarvestBatch(BatchID) ON DELETE SET NULL,
  Result VARCHAR(50),
  MoistureLevel  DECIMAL(5,2),
  CertificationID VARCHAR(100),

  Date DATE

);
```

7.1.5 PurchaseOrder Table

```
CREATE TABLE PurchaseOrder (

  POID SERIAL PRIMARY KEY,

  EmployeeID INTEGER REFERENCES Employee(EmployeeID) ON DELETE SET NULL,

  SupplierID INTEGER REFERENCES Supplier(SupplierID) ON DELETE SET NULL,

  OrderDate DATE NOT NULL,

  ExpectedDeliveryDate  DATE,

  Status VARCHAR(50) DEFAULT 'pending',

  TotalAmount DECIMAL(15,2)

);
```

7.1.6 ProcessingOrder

```
CREATE TABLE ProcessingOrder (

  ProcessingID SERIAL PRIMARY KEY,

  ProductTypeID  INTEGER REFERENCES ProductType(ProductTypeID)  ON DELETE SET
NULL,

  UnitID INTEGER REFERENCES ProcessingUnit(UnitID) ON DELETE SET NULL,

  StartDate DATE,

  EndDate DATE,

  OutputQuantity DECIMAL(10,2),

  EfficiencyRate DECIMAL(5,2)

);
```

7.1.7 InventoryTransaction Table

```
CREATE TABLE InventoryTransaction (
  TransactionID SERIAL PRIMARY KEY,
  EmployeeID INTEGER REFERENCES Employee(EmployeeID) ON DELETE SET NULL,
  StockID INTEGER REFERENCES StockItem(StockID) ON DELETE SET NULL,
  WarehouseID INTEGER REFERENCES Warehouse(WarehouseID) ON DELETE SET NULL,
  TransactionType VARCHAR(50) NOT NULL,
  Quantity DECIMAL(10,2) NOT NULL,
  TransactionDate TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  Remarks TEXT
);
```

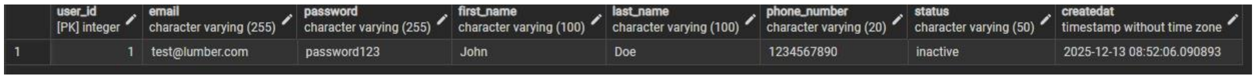
7.2 SQL Queries (DML)

7.2.1 User and Access Control Queries

This subsection presents a representative set of SQL queries used for managing users, roles, permissions, and audit logs. These queries demonstrate standard CRUD operations and enforce role-based access control within the system. Similar query patterns are applied across other system modules.

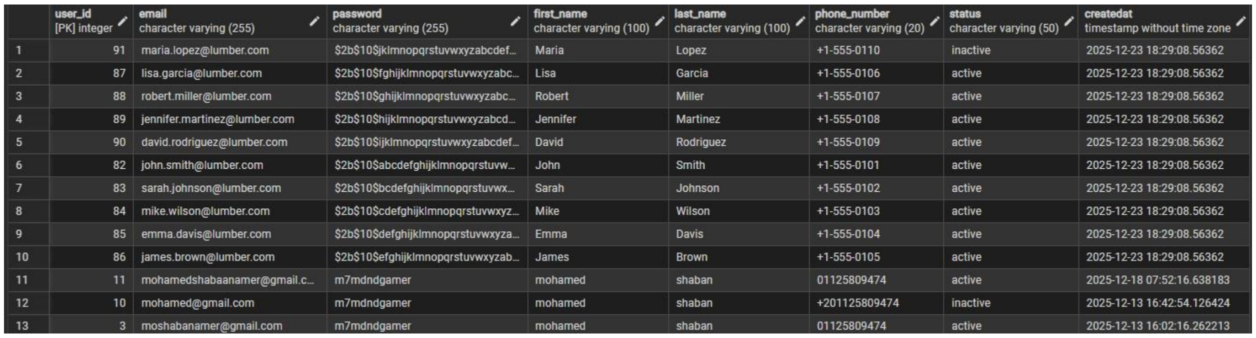
```
-- INSERT
INSERT INTO "User" (Email, Password, First_Name, Last_Name, Phone_Number, Status)
VALUES ($1, $2, $3, $4, $5, $6)
RETURNING User_ID;

-- UPDATE
```

```
UPDATE "User"

SET Email = $2, Password = $3, First_Name = $4, Last_Name = $5,

   Phone_Number = $6, Status = $7

WHERE User_ID  = $1;


-- DELETE

DELETE FROM "User"

WHERE User_ID  = $1;


-- VIEW by ID

SELECT User_ID, Email, Password, First_Name, Last_Name, Phone_Number, Status,
CreatedAt

FROM "User"

WHERE User_ID  = $1;
```



	user_id [PK] integer	email character varying (255)	password character varying (255)	first_name character varying (100)	last_name character varying (100)	phone_number character varying (20)	status character varying (50)	createdat timestamp without time zone
1	1	test@lumber.com	password123	John	Doe	1234567890	inactive	2025-12-13 08:52:06.090893

```
-- VIEW all

SELECT User_ID, Email, Password, First_Name, Last_Name, Phone_Number, Status,
CreatedAt

FROM "User"

ORDER BY CreatedAt DESC;
```



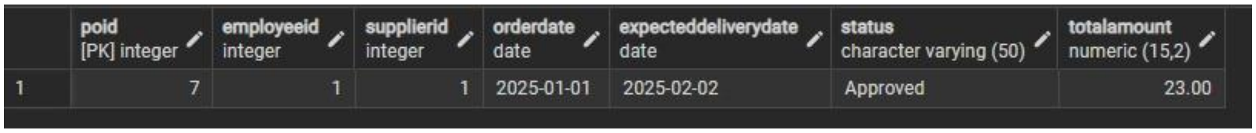
	user_id [PK] integer	email character varying (255)	password character varying (255)	first_name character varying (100)	last_name character varying (100)	phone_number character varying (20)	status character varying (50)	createdat timestamp without time zone
1	91	maria.lopez@lumber.com	$2b$10$jklmnopqrstuvwxyzabcdef…	Maria	Lopez	+1-555-0110	inactive	2025-12-23 18:29:08.56362
2	87	lisa.garcia@lumber.com	$2b$10$fghijklmnopqrstuvwxyzabc…	Lisa	Garcia	+1-555-0106	active	2025-12-23 18:29:08.56362
3	88	robert.miller@lumber.com	$2b$10$ghijklmnopqrstuvwxyzabc…	Robert	Miller	+1-555-0107	active	2025-12-23 18:29:08.56362
4	89	jennifer.martinez@lumber.com	$2b$10$hijklmnopqrstuvwxyzabcd…	Jennifer	Martinez	+1-555-0108	active	2025-12-23 18:29:08.56362
5	90	david.rodriguez@lumber.com	$2b$10$ijklmnopqrstuvwxyzabcdef…	David	Rodriguez	+1-555-0109	active	2025-12-23 18:29:08.56362
6	82	john.smith@lumber.com	$2b$10$abcdefghijklmnopqrstuvw…	John	Smith	+1-555-0101	active	2025-12-23 18:29:08.56362
7	83	sarah.johnson@lumber.com	$2b$10$bcdefghijklmnopqrstuvwx…	Sarah	Johnson	+1-555-0102	active	2025-12-23 18:29:08.56362
8	84	mike.wilson@lumber.com	$2b$10$cdefghijklmnopqrstuvwxyz…	Mike	Wilson	+1-555-0103	active	2025-12-23 18:29:08.56362
9	85	emma.davis@lumber.com	$2b$10$defghijklmnopqrstuvwxyza…	Emma	Davis	+1-555-0104	active	2025-12-23 18:29:08.56362
10	86	james.brown@lumber.com	$2b$10$efghijklmnopqrstuvwxyzab…	James	Brown	+1-555-0105	active	2025-12-23 18:29:08.56362
11	11	mohamedshabaanamer@gmail.c…	m7mdndgamer	mohamed	shaban	01125809474	active	2025-12-18 07:52:16.638183
12	10	mohamed@gmail.com	m7mdndgamer	mohamed	shaban	+201125809474	inactive	2025-12-13 16:42:54.126424
13	3	moshabanamer@gmail.com	m7mdndgamer	mohamed	shaban	01125809474	active	2025-12-13 16:02:16.262213

7.2.2 Procurement & purchase orders queries

This subsection presents representative SQL queries used for managing procurement operations and purchase orders. The queries demonstrate a master–detail relationship between purchase orders and their associated items, along with standard CRUD operations and reporting queries. Similar query patterns are applied across other transactional modules in the system.

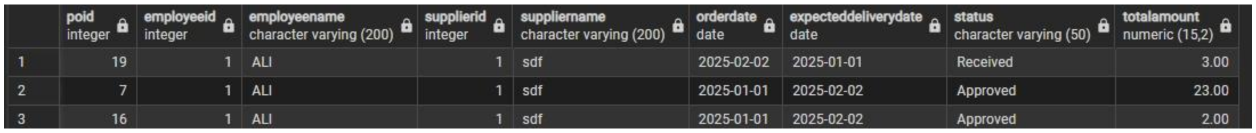
```
-- INSERT

INSERT INTO PurchaseOrder (EmployeeID, SupplierID, OrderDate, ExpectedDeliveryDate,
Status, TotalAmount)

VALUES ($1, $2, $3, $4, $5, $6)

RETURNING POID;


-- UPDATE

UPDATE PurchaseOrder

SET EmployeeID = $2, SupplierID = $3, OrderDate = $4,

   ExpectedDeliveryDate  = $5, Status = $6, TotalAmount = $7

WHERE POID = $1;


-- DELETE

DELETE FROM PurchaseOrder

WHERE POID = $1;


-- VIEW by ID

SELECT POID, EmployeeID, SupplierID, OrderDate, ExpectedDeliveryDate, Status,
TotalAmount
```

```
FROM PurchaseOrder

WHERE POID = $1;
```



	poid [PK] integer	employeeid integer	supplierid integer	orderdate date	expecteddeliverydate date	status character varying (50)	totalamount numeric (15,2)
1	7	1	1	2025-01-01	2025-02-02	Approved	23.00

```
-- VIEW all

SELECT po.POID, po.EmployeeID, e.FullName AS EmployeeName,

    po.SupplierID, s.CompanyName AS SupplierName,

    po.OrderDate, po.ExpectedDeliveryDate,  po.Status, po.TotalAmount

FROM PurchaseOrder po

LEFT JOIN Employee e ON po.EmployeeID = e.EmployeeID

LEFT JOIN Supplier s ON po.SupplierID = s.SupplierID

ORDER BY po.OrderDate DESC;
```



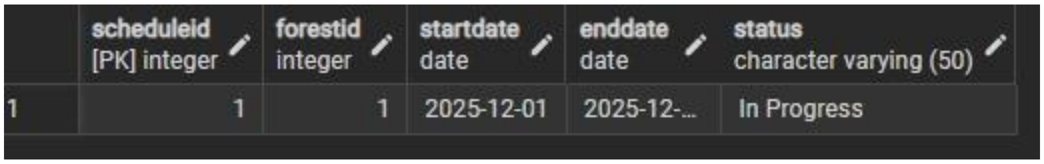
	poid integer	employeeid integer	employeename character varying (200)	supplierid integer	suppliername character varying (200)	orderdate date	expecteddeliverydate date	status character varying (50)	totalamount numeric (15,2)
1	19	1	ALI	1	sdf	2025-02-02	2025-01-01	Received	3.00
2	7	1	ALI	1	sdf	2025-01-01	2025-02-02	Approved	23.00
3	16	1	ALI	1	sdf	2025-01-01	2025-02-02	Approved	2.00

7.2.3. Forest Harvesting Operations Queries

This subsection presents representative SQL queries used to manage harvesting schedules and harvest batches. The queries support creating, updating, deleting, and retrieving harvesting plans and recorded harvest batches, ensuring accurate tracking of forest operations and harvested wood details.

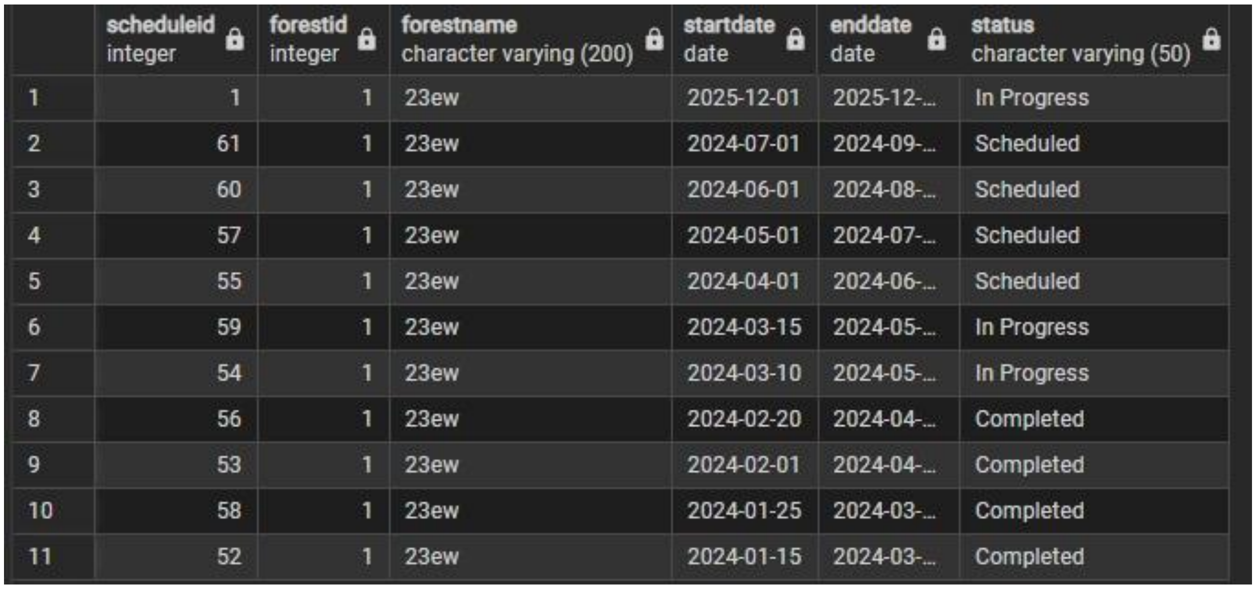
```
-- INSERT

INSERT INTO HarvestSchedule (ForestID, StartDate, EndDate, Status)

VALUES ($1, $2, $3, $4)

RETURNING ScheduleID;
```

```
-- UPDATE

UPDATE HarvestSchedule

SET ForestID = $2, StartDate = $3, EndDate = $4, Status = $5

WHERE ScheduleID = $1;


-- DELETE

DELETE FROM HarvestSchedule

WHERE ScheduleID = $1;


-- VIEW by ID

SELECT ScheduleID, ForestID, StartDate, EndDate, Status

FROM HarvestSchedule

WHERE ScheduleID = $1;
```



	scheduleid [PK] integer	forestid integer	startdate date	enddate date	status character varying (50)
1	1	1	2025-12-01	2025-12-...	In Progress

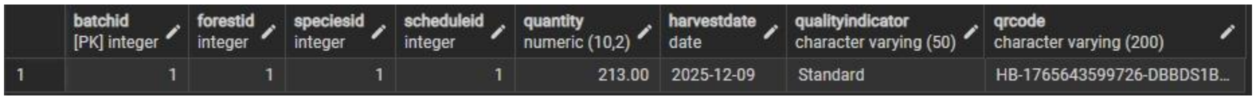
```
-- VIEW all

SELECT hs.ScheduleID, hs.ForestID, f.ForestName, hs.StartDate, hs.EndDate, hs.Status

FROM HarvestSchedule hs

JOIN Forest f ON hs.ForestID = f.ForestID

ORDER BY hs.StartDate DESC;
```



	scheduleid integer	forestid integer	forestname character varying (200)	startdate date	enddate date	status character varying (50)
1	1	1	23ew	2025-12-01	2025-12-...	In Progress
2	61	1	23ew	2024-07-01	2024-09-...	Scheduled
3	60	1	23ew	2024-06-01	2024-08-...	Scheduled
4	57	1	23ew	2024-05-01	2024-07-...	Scheduled
5	55	1	23ew	2024-04-01	2024-06-...	Scheduled
6	59	1	23ew	2024-03-15	2024-05-...	In Progress
7	54	1	23ew	2024-03-10	2024-05-...	In Progress
8	56	1	23ew	2024-02-20	2024-04-...	Completed
9	53	1	23ew	2024-02-01	2024-04-...	Completed
10	58	1	23ew	2024-01-25	2024-03-...	Completed
11	52	1	23ew	2024-01-15	2024-03-...	Completed

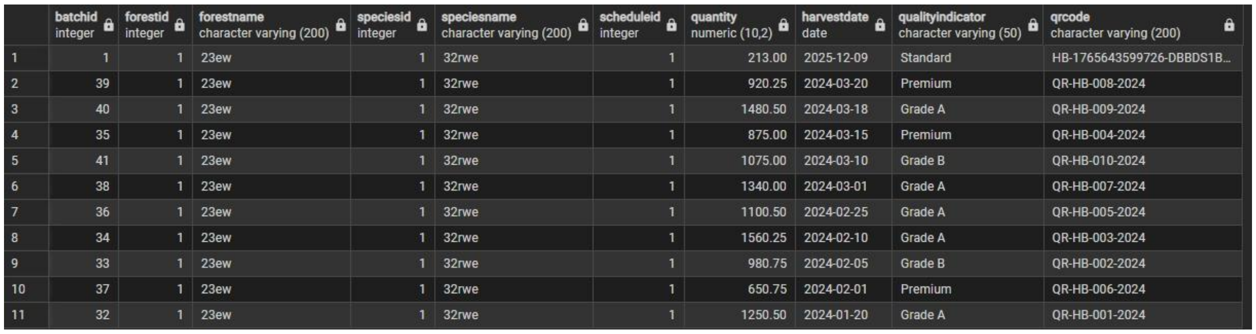
```
-- INSERT

INSERT INTO HarvestBatch (ForestID, SpeciesID, ScheduleID, Quantity, HarvestDate,
QualityIndicator, QRCode)

VALUES ($1, $2, $3, $4, $5, $6, $7)

RETURNING BatchID;


-- UPDATE

UPDATE HarvestBatch

SET ForestID = $2, SpeciesID = $3, ScheduleID = $4, Quantity = $5,

  HarvestDate = $6, QualityIndicator = $7, QRCode = $8

WHERE BatchID = $1;


-- DELETE

DELETE FROM HarvestBatch

WHERE BatchID = $1;
```

```
-- VIEW by ID

SELECT BatchID, ForestID, SpeciesID, ScheduleID, Quantity,

    HarvestDate, QualityIndicator, QRCode

FROM HarvestBatch

WHERE BatchID = $1;
```



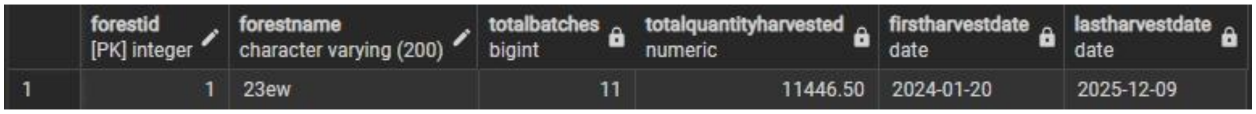
	batchid [PK] integer	forestid integer	speciesid integer	scheduleid integer	quantity numeric (10,2)	harvestdate date	qualityindicator character varying (50)	qrcode character varying (200)
1	1	1	1	1	213.00	2025-12-09	Standard	HB-1765643599726-DBBDS1B...

```
-- VIEW all

SELECT hb.BatchID, hb.ForestID, f.ForestName, hb.SpeciesID, ts.SpeciesName,

    hb.ScheduleID, hb.Quantity, hb.HarvestDate, hb.QualityIndicator, hb.QRCode

FROM HarvestBatch hb

LEFT JOIN Forest f ON hb.ForestID = f.ForestID

LEFT JOIN TreeSpecies ts ON hb.SpeciesID = ts.SpeciesID

ORDER BY hb.HarvestDate DESC;
```



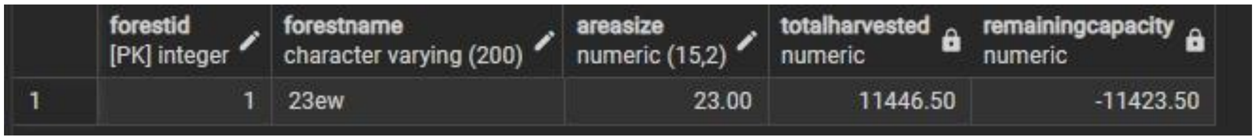
	batchid integer	forestid integer	forestname character varying (200)	speciesid integer	speciesname character varying (200)	scheduleid integer	quantity numeric (10,2)	harvestdate date	qualityindicator character varying (50)	qrcode character varying (200)
1	1	1	23ew	1	32rwe	1	213.00	2025-12-09	Standard	HB-1765643599726-DBBDS1B...
2	39	1	23ew	1	32rwe	1	920.25	2024-03-20	Premium	QR-HB-008-2024
3	40	1	23ew	1	32rwe	1	1480.50	2024-03-18	Grade A	QR-HB-009-2024
4	35	1	23ew	1	32rwe	1	875.00	2024-03-15	Premium	QR-HB-004-2024
5	41	1	23ew	1	32rwe	1	1075.00	2024-03-10	Grade B	QR-HB-010-2024
6	38	1	23ew	1	32rwe	1	1340.00	2024-03-01	Grade A	QR-HB-007-2024
7	36	1	23ew	1	32rwe	1	1100.50	2024-02-25	Grade A	QR-HB-005-2024
8	34	1	23ew	1	32rwe	1	1560.25	2024-02-10	Grade A	QR-HB-003-2024
9	33	1	23ew	1	32rwe	1	980.75	2024-02-05	Grade B	QR-HB-002-2024
10	37	1	23ew	1	32rwe	1	650.75	2024-02-01	Premium	QR-HB-006-2024
11	32	1	23ew	1	32rwe	1	1250.50	2024-01-20	Grade A	QR-HB-001-2024

```
SELECT f.ForestID, f.ForestName,

    COUNT(hb.BatchID) AS TotalBatches,

    SUM(hb.Quantity) AS TotalQuantityHarvested,

    MIN(hb.HarvestDate) AS FirstHarvestDate,
    MAX(hb.HarvestDate) AS LastHarvestDate
FROM Forest f
LEFT JOIN HarvestBatch hb ON f.ForestID = hb.ForestID
GROUP BY f.ForestID, f.ForestName
ORDER BY TotalQuantityHarvested DESC;
```



	forestid [PK] integer	forestname character varying (200)	totalbatches bigint	totalquantityharvested numeric	firstharvestdate date	lastharvestdate date
1	1	23ew	11	11446.50	2024-01-20	2025-12-09

```
SELECT f.ForestID, f.ForestName, f.AreaSize,
    COALESCE(SUM(hb.Quantity), 0) AS TotalHarvested,
    f.AreaSize - COALESCE(SUM(hb.Quantity), 0) AS RemainingCapacity
FROM Forest f
LEFT JOIN HarvestBatch hb ON f.ForestID = hb.ForestID
GROUP BY f.ForestID, f.ForestName, f.AreaSize
ORDER BY RemainingCapacity DESC;
```



	forestid [PK] integer	forestname character varying (200)	areasize numeric (15,2)	totalharvested numeric	remainingcapacity numeric
1	1	23ew	23.00	11446.50	-11423.50

7.3 Relational Algebra Expressions

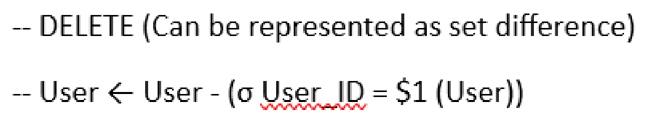
7.3.1 User and Access Control RA

This query first applies a **selection (σ)** operation to retrieve the user whose `User_ID` matches the given parameter.

π User_ID, Email, Password, First_Name, Last_Name, Phone_Number, Status, CreatedAt

(σ User_ID = $1 (User))

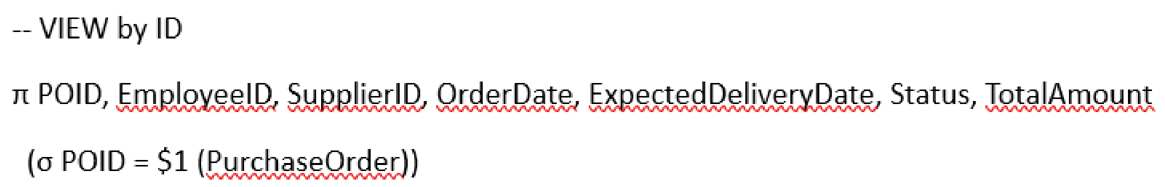
This is a delete operation represented using a **set difference** in relational algebra:



```
-- DELETE (Can be represented as set difference)

-- User ← User - (σ User_ID = $1 (User))
```

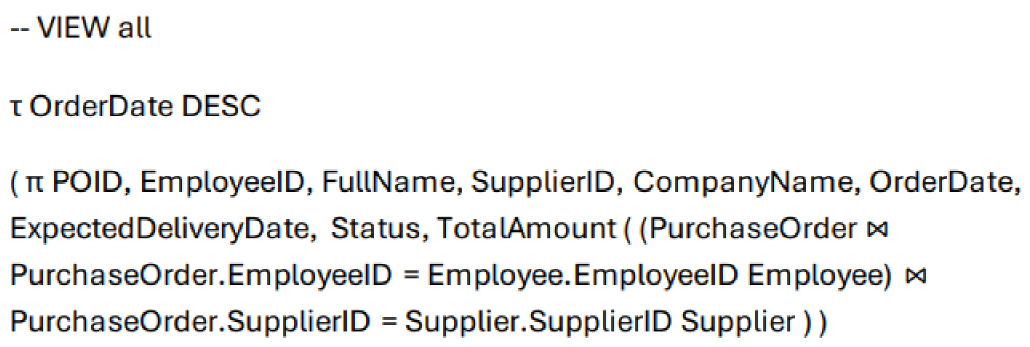
7.3.2 Procurement & purchase orders RA

Applying a **selection (σ)** operation to retrieve the purchase order by ID



```
-- VIEW by ID

π POID, EmployeeID, SupplierID, OrderDate, ExpectedDeliveryDate, Status, TotalAmount

 (σ POID = $1 (PurchaseOrder))
```

Applying a **join operation** that combines data from multiple relations to produce a comprehensive view of purchase orders:



```
-- VIEW all

τ OrderDate DESC

( π POID, EmployeeID, FullName, SupplierID, CompanyName, OrderDate,
ExpectedDeliveryDate, Status, TotalAmount ( (PurchaseOrder ⋈
PurchaseOrder.EmployeeID = Employee.EmployeeID Employee) ⋈
PurchaseOrder.SupplierID = Supplier.SupplierID Supplier ) )
```

7.3.3 Forest & Harvest Batches Details RA

This query finds forests with active harvest schedules by joining forest data with schedule data and filtering by status:

```
-- Get forest details with active harvest schedules
τ StartDate
  (π ForestID, ForestName, GeoLocation, AreaSize, ScheduleID, StartDate, EndDate,
Status
    (σ Status = $1
      (Forest ⋈ Forest.ForestID = HarvestSchedule.ForestID HarvestSchedule)))
```

This query calculates how much harvest capacity each forest has left by comparing total area size with total harvested quantity:

```
-- Get forest capacity vs harvested analysis
τ RemainingCapacity DESC
  (π ForestID, ForestName, AreaSize, TotalHarvested, RemainingCapacity
    (γ ForestID, ForestName, AreaSize; COALESCE(SUM(Quantity), 0) → TotalHarvested,
      (AreaSize - COALESCE(SUM(Quantity), 0)) → RemainingCapacity
     (Forest ⟕ Forest.ForestID = HarvestBatch.ForestID HarvestBatch)))
```

This query finds harvest batches filtered by quality rating and includes related forest/species info:

```
τ HarvestDate DESC
  (π BatchID, ForestID, ForestName, SpeciesID, SpeciesName, Quantity, HarvestDate,
QualityIndicator, QRCode
    (σ QualityIndicator = $1
      ((HarvestBatch ⋈ HarvestBatch.ForestID = Forest.ForestID Forest)
        ⋈ HarvestBatch.SpeciesID = TreeSpecies.SpeciesID TreeSpecies)))
```

8. Data Dictionary

8.1 User Table

Col Name	Description

User_ID	Unique identifier for each user
Email	User email address
Password	Encrypted user password
First_Name	User first name
Last_Name	User last name
Phone_Number	Contact phone number
Status	Account status (active/inactive)
CreatedAt	Account creation timestamp

8.2 Employee Table

Col Name	Description
EmployeeID	Unique identifier for employee
FullName	Employee full name
Department	Department name
Position	Job position
HireDate	Date of hiring
PerformanceRating	Performance evaluation score

8.3 Supplier Table

Col Name	Description
SupplierID	Unique supplier identifier
CompanyName	Supplier company name
ContactPerson	Main contact person
Email	Supplier email
Phone	Supplier phone number
ComplianceStatus	Compliance verification status
Raw	Indicates raw material supplier
Semi_Processed	Indicates semi-processed supplier

8.4 ProductType Table

Col Name	Description
ProductTypeID	Unique product type identifier
Name	Product name
Description	Product description
Grade	Product quality grade
UnitOfMeasure	Measurement unit (e.g., cubic meter)

8.5 PurchaseOrder Table

Col Name	Description
POID	Purchase order identifier
EmployeeID	Employee who created the order
SupplierID	Supplier associated with the order
OrderDate	Date of order creation
ExpectedDeliveryDate	Expected delivery date
Status	Order status (pending, delivered, etc.)
TotalAmount	Total order value

8.6 Forest Table

Col Name	Description
ForestID	Unique identifier for each forest
ForestName	Name of the forest
GeoLocation	Geographic location of the forest
AreaSize	Total forest area size
OwnershipType	Ownership classification (e.g., public/private)
Status	Current forest status

8.7 HarvestSchedule Table

Col Name	Description
ScheduleID	Unique harvest schedule identifier

ForestID	Reference to the associated forest
StartDate	Harvesting start date
EndDate	Harvesting end date
Status	Schedule status

8.8 HarvestBatch Table

Col Name	Description
BatchID	Unique identifier for harvest batch
ForestID	Forest where harvesting occurred
SpeciesID	Tree species harvested
ScheduleID	Related harvesting schedule
Quantity	Quantity harvested
HarvestDate	Date of harvesting
QualityIndicator	Quality assessment result
QRCode	Unique QR code for batch traceability

8.9 Warehouse Table

Col Name	Description
WarehouseID	Unique warehouse identifier
Name	Warehouse name
Location	Physical warehouse location
Capacity	Maximum storage capacity
Contact	Warehouse contact information

8.10 SalesOrder Table

Col Name	Description
SOID	Sales order identifier
EmployeeID	Employee who created the order
CustomerID	Customer placing the order
OrderDate	Order creation date
DeliveryDate	Scheduled delivery date
Status	Order status
TotalAmount	Total sales order value

9. System Design and Architecture

Our system follows a **three-tier architecture** consisting of a frontend layer, backend layer, and database layer. This separation of concerns improves maintainability, scalability, and system organization.

9.1 Frontend Layer

The frontend of the system is developed using **React.js**, which provides a dynamic and responsive user interface. React enables the creation of reusable components and supports efficient state management, allowing users to interact smoothly with system features such as dashboards, forms, and data tables. The graphical user interface (GUI) is accessed through a web browser and communicates with the backend through HTTP requests.

9.2 Backend Layer

The backend of the system is implemented using **Go (Golang)**. It is responsible for handling business logic, processing client requests, enforcing system rules, and managing communication with the database. The backend exposes APIs that allow the frontend to perform operations such as user authentication, order management, inventory tracking, and reporting. This layer acts as an intermediary between the user interface and the database.

9.3 Database Layer

The system uses **PostgreSQL** as the Database Management System (DBMS). PostgreSQL stores all persistent data, including users, roles, suppliers, orders, inventory, and transactions. The relational nature of PostgreSQL ensures data integrity through constraints, relationships, and normalization, making it suitable for complex supply chain operations.

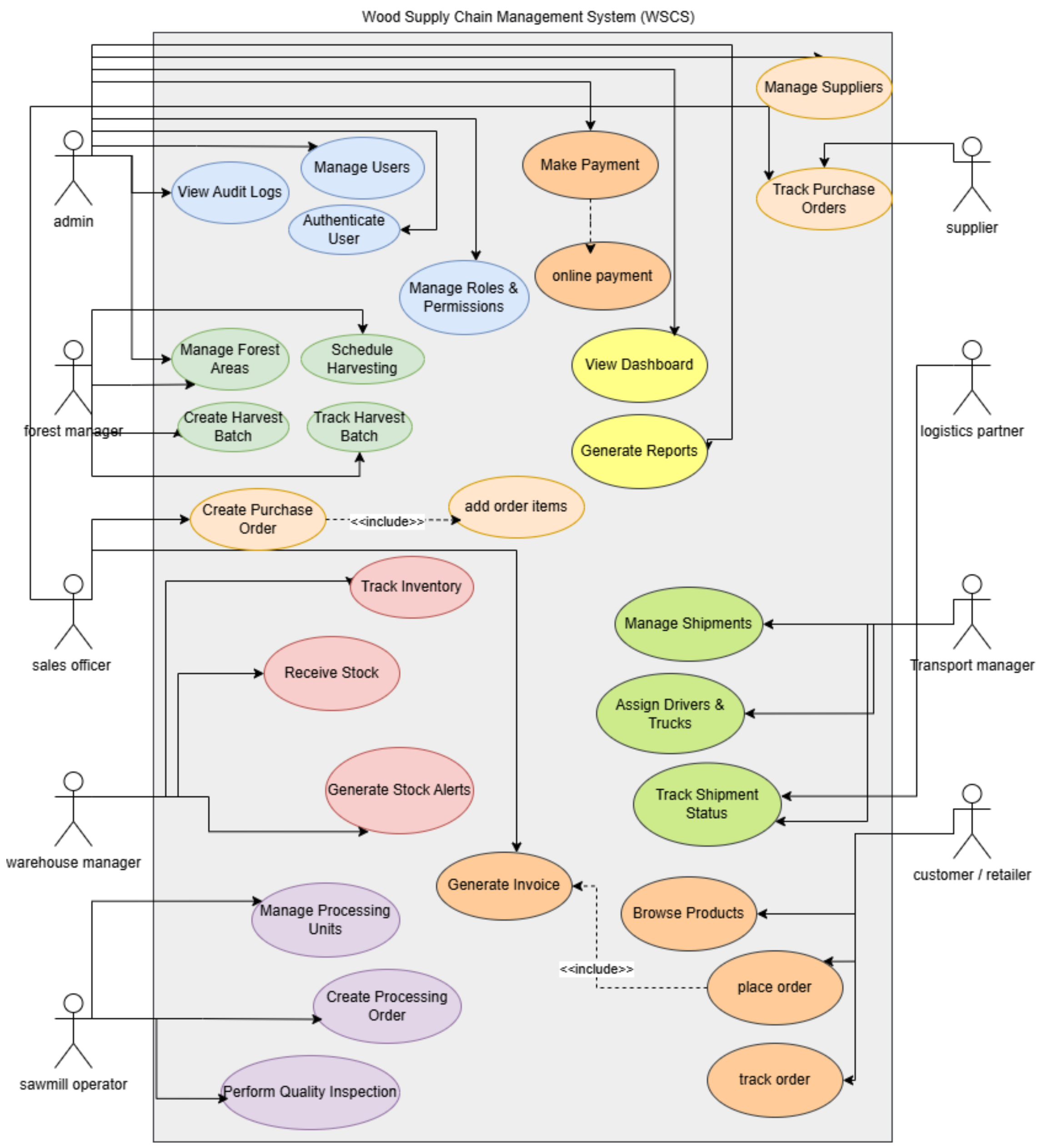
9.4 System Communication Flow

The frontend communicates with the backend using **RESTful API calls**, sending and receiving data in structured formats. The backend processes requests, applies validation and business logic, and interacts with the PostgreSQL database to retrieve or update data before returning responses to the frontend.

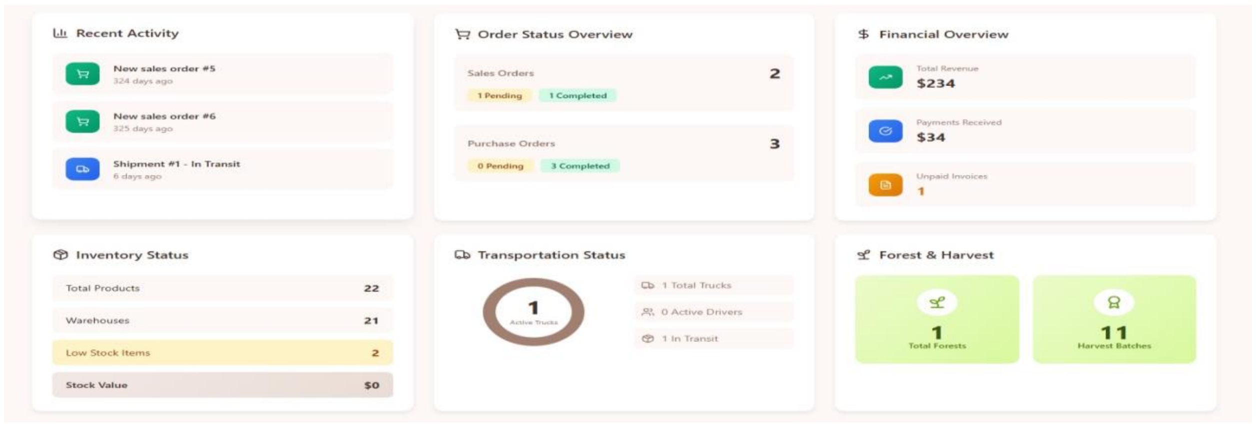
10. Analytical Dashboards & Visualizations

This section presents additional visual components developed to enhance system monitoring and decision making.

10.1 Use Case Diagram



10.2 Dashboard



Recent Activity

New sales order #5
324 days ago

New sales order #6
325 days ago

Shipment #1 - In Transit
6 days ago

Order Status Overview

Sales Orders 2
1 Pending 1 Completed

Purchase Orders 3
0 Pending 3 Completed

Financial Overview

Total Revenue
$234

Payments Received
$34

Unpaid Invoices
1

Inventory Status

Total Products	22
Warehouses	21
Low Stock Items	2
Stock Value	$0

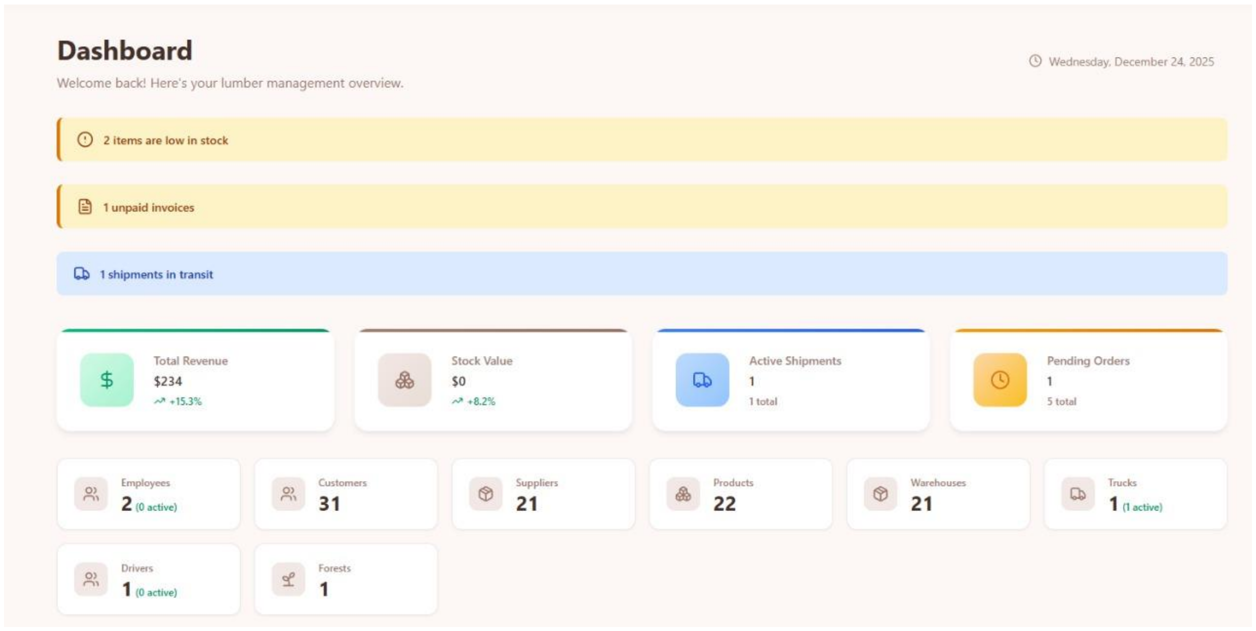
Transportation Status

1 Active Trucks

1 Total Trucks
0 Active Drivers
1 In Transit

Forest & Harvest

1 Total Forests
11 Harvest Batches



Dashboard

Welcome back! Here's your lumber management overview.

Wednesday, December 24, 2025

2 items are low in stock

1 unpaid invoices

1 shipments in transit

Total Revenue
$234
+15.3%

Stock Value
$0
+8.2%

Active Shipments
1
1 total

Pending Orders
1
5 total

Employees
2 (0 active)

Customers
31

Suppliers
21

Products
22

Warehouses
21

Trucks
1 (1 active)

Drivers
1 (0 active)

Forests
1

11. Conclusion & Future Work

11.1 Conclusion

This project presented the design and implementation of a comprehensive Wood Supply Chain Management System that supports the full lifecycle of supply chain operations, from forest management to customer delivery. The system applies a structured relational database design, role-based access control, and a multi-tier architecture to ensure data integrity, scalability, and secure system operation. Analytical dashboards further enhance visibility and support data-driven decision making.

11.2 Future Work

Potential enhancements to the system include:

- Integration of predictive analytics to forecast demand and optimize inventory levels
- Development of mobile applications to support field workers and transport staff
- Integration with IoT devices for real-time tracking of shipments, machinery, and inventory
- Expansion of business intelligence dashboards with advanced KPIs and drill-down analytics
- Implementation of AI-based optimization for transportation routing and resource allocation
- Support for integration with external enterprise systems and government compliance platforms