FULL PHASE 5 DOCUMENT – REPORTS & DASHBOARDS,
This phase describes how analytics were designed, how dashboards were created, and how reports provide insights in your Pharmacy Delivery CRM.

📘 PHASE 5 – REPORTS & DASHBOARDS

Goal:

To design analytical components (reports & dashboards) that provide real-time visibility into pharmacy delivery operations, agent performance, and delivery efficiency.

Salesforce reporting tools allow business users to monitor delivery performance, customer satisfaction, agent workload, and revenue trends.

5.1 Introduction to Reporting in Salesforce

Reports help summarize and analyze data stored in Salesforce.
In the Pharmacy Delivery CRM, reports are used to:

- Track all delivery records
- Identify delivery status distribution
- View total revenue generated
- Compare delivery agent performance
- Identify late or failed deliveries

These insights enable pharmacy managers to make informed decisions.

5.2 Requirements for Analytics

The following business requirements drove the report and dashboard design:

1. Delivery Status Insights

Managers must instantly see:

- How many deliveries are New
- How many are Scheduled
- How many are Out for Delivery
- How many are Delivered
- How many are Late or Failed

2. Revenue Tracking

Ability to view:

- Total amount per day
- Total revenue per status
- Cumulative delivery revenue

3. Delivery Agent Monitoring

(For future phase)

- Workload per agent
- Deliveries handled daily

4. Understand Delivery Volume Trends

Helps with:

- Staffing decisions
- Delivery planning
- Pharmacy operational strategy

5.3 Reports Created

Two core reports were created. They are used both individually and as dashboard components.

5.3.1 Report 1: All Deliveries Report

Type:

Tabular or Summary (depending on format)

Folder:

Public Reports

Data Source:

Custom Report Type → Deliveries

Columns Displayed:

- Delivery Name
- Delivery Date
- Patient
- Pharmacy
- Status
- Amount

- Delivery Agent (optional)

Filters Used:

- Status ≠ Cancelled (optional)
- Date Range = All Time

Summary:

- **Sum of Amount** (Total Value of Deliveries)

Purpose:

- Full list of all delivery transactions
- Finance and transaction reporting

5.3.2 Report 2: Deliveries by Status

Type:

Summary Report

Grouping:

- Grouped by **Status**

Value:

- **SUM(Amount)** or **Record Count**

Chart Used:

Donut Chart

Purpose:

- Visual representation of delivery status
- Helps quickly identify bottlenecks
- Shows overall delivery performance

This report is used for the donut chart on the dashboard.

5.4 Dashboard Creation

A dashboard named **Pharmacy Delivery Dashboard** was created to visualize insights at a glance.

5.4.1 Dashboard Structure

Dashboard Name:

Pharmacy Delivery Dashboard

Folder:

Public Dashboards

Theme:

Light

5.4.2 Dashboard Components

The dashboard includes:

Component 1: Donut Chart – Deliveries by Status

Source Report:

Deliveries by Status

Display Metrics:

- Slices represent status categories
- Colors assigned automatically
- Caption: “Deliveries by Status”
- Aggregation: Record Count or SUM(Amount)

Purpose:

- Instant view of pending, late, delivered orders
- Helps management track daily operations

Component 2: Table – All Deliveries

Source Report:

All Deliveries Report

Columns Displayed:

- Delivery Name
- Delivery Date
- Patient
- Pharmacy
- Status

- Amount

Purpose:

- To view detailed delivery records
- Helps manager review each delivery and check progress

5.5 Expected Insights from Reports & Dashboards

The analytical layer of the CRM provides the following insights:

📊 A) Delivery Performance Tracking

Managers can identify:

- Number of deliveries completed
- Number of pending deliveries
- Late deliveries
- Cancelled deliveries
- Failed or reattempted deliveries

This helps in resource planning.

💰 B) Revenue Monitoring

Using SUM of Amount, managers can track:

- Revenue per day
- Revenue per week
- Revenue per month
- Total delivery revenue

🚴 C) Delivery Agent Monitoring

(Not implemented fully in current phase but supported)

Managers can measure:

- Deliveries per agent
- Agent performance
- Workload distribution

📍 D) Bottleneck Identification

Donut chart reveals:

- Too many late deliveries
- Too many pending deliveries
- Failures due to wrong address or improper scheduling

5.6 Report & Dashboard Refresh Logic

Dashboard Refresh Frequency:

- Manual Refresh
- Optionally scheduled (daily 8 AM)

Data updates automatically when:

- Delivery status changes
- New deliveries are created
- Amount changes

Reporting system is dynamic and always up-to-date.

5.7 Future Enhancements to Analytics

1. Agent Performance Dashboard

- Deliveries per agent
- Average delivery time
- Customer satisfaction ratings

2. Pharmacy Performance Dashboard

- Top pharmacies by orders
- Average delivery cost
- Delivery success rate

3. Geography-based Visualization

- Heatmaps
- Delivery zones

4. Custom KPIs

- On-time delivery rate
- Failed delivery percentage
- SLA violations

Perfect, this is clear.
You want **Phase 5 = Apex Programming (Developer)**, written in the **same bullet-wise style** as your Car Rental example, but adapted to **Pharmacy Delivery CRM**.

Here is **Phase 5** for your project:

Phase 5: Apex Programming (Developer)

👉 **Goal: Add advanced logic and extensibility beyond clicks & flows.**

Even though most of the Pharmacy Delivery CRM is built using declarative tools (Flows, Validation Rules, Reports, Dashboards), this phase explains how **Apex** could be used to enhance the solution further. This is important for academic completeness and for showing you understand the developer side of Salesforce.

---

1. Classes & Objects

- Create a reusable **DeliveryService** Apex class.
- Responsibilities of DeliveryService:
    - Calculate SLA / delays (how many days late a delivery is).
    - Centralize business rules for “Late”, “Failed”, “High Priority” deliveries.
    - Provide helper methods for triggers, batch jobs and schedulers.

Example (conceptual):

```
public class DeliveryService {
   public static Boolean isLate(Delivery__c d) {
      return d.Delivery_Date__c < Date.today() && d.Status__c != 'Delivered';
   }
}
```

---

2. Apex Triggers

Use Case:

- On **Delivery Insert / Update**, enforce rules like:
    - Prevent duplicate deliveries for same patient at same date/time.

- Set default status automatically when records are created via API/import.
- Log changes for audit (optional).

Trigger Example (conceptual):

```
trigger DeliveryTrigger on Delivery__c (before insert, before update) {
   if(Trigger.isBefore){
      if(Trigger.isInsert || Trigger.isUpdate){
         DeliveryTriggerHandler.beforeSave(Trigger.new, Trigger.oldMap);
      }
   }
}
```

---

3. Trigger Design Pattern

Instead of writing logic directly in the trigger, a **handler class** is used:

- Trigger: only routes events (before insert, before update, etc.)
- **Handler class (DeliveryTriggerHandler)**: contains actual logic.

This follows Salesforce best practices and makes unit testing easier.

```
public class DeliveryTriggerHandler {
   public static void beforeSave(List<Delivery__c> newList, Map<Id, Delivery__c>
oldMap){
      // validate, set defaults, prevent duplicates, etc.
   }
}
```

---

4. SOQL & SOSL Usage

SOQL (Salesforce Object Query Language) is used in Apex to fetch data.

Examples in this CRM:

- Fetch **available delivery agents**:

```
List<Delivery_Agent__c> agents = [
   SELECT Id, Name, Email__c
   FROM Delivery_Agent__c
   WHERE Active__c = TRUE
```

];

- Fetch **late deliveries**:

```
List<Delivery__c> lateDeliveries = [
   SELECT Id, Status__c, Delivery_Date__c
   FROM Delivery__c
   WHERE Delivery_Date__c < :Date.today()
   AND Status__c != 'Delivered'
];
```

SOSL could be used for searching patients or pharmacies globally by name or phone.

---

5. Collections: List, Set, Map

Collections allow bulk-processing:

- **List<Delivery__c>** → to process many deliveries together.
- **Set<Id>** → to store unique Patient/Agent IDs (no duplicates).
- **Map<Id, Delivery__c>** → for quick lookup of records by Id.

Example:

```
Set<Id> patientIds = new Set<Id>();
for(Delivery__c d : Trigger.new){
   if(d.Patient__c != null){
      patientIds.add(d.Patient__c);
   }
}
```

---

6. Control Statements

Control statements (IF, FOR, WHILE) enforce business rules:

- **Example rule:**
  If a new delivery overlaps another delivery for the same patient on the same date, prevent creation.

```
if(existingDeliveries.size() > 0){
   d.addError('A delivery for this patient already exists on this date.');
}
```

---

7. Batch Apex (Overdue Deliveries Job)

A **Batch Apex job** can be used to process large volumes of deliveries at night:

- Every night:
    - Find all **pending** deliveries with Delivery Date < Today.
    - Mark them as **Late** or **Failed**.
    - Optionally send summary email to manager.

Class Example Idea:

```
global class LateDeliveryBatch implements Database.Batchable<SObject> {
   global Database.QueryLocator start(Database.BatchableContext bc) {
      return Database.getQueryLocator(
         'SELECT Id, Status__c, Delivery_Date__c FROM Delivery__c WHERE Delivery_Date__c < TODAY'
      );
   }
   global void execute(Database.BatchableContext bc, List<Delivery__c> scope) {
      // set Status__c = 'Late', update records
   }
   global void finish(Database.BatchableContext bc) { }
}
```

---

8. Queueable Apex

For more complex or chained async logic:

- Example:
  Calculate discounts or apply promotional rules to **bulk deliveries**.
- Queueable jobs allow chaining and passing complex objects.

```
public class DiscountQueueable implements Queueable {
   public void execute(QueueableContext context) {
      // apply discount logic on deliveries
   }
}
```

---

9. Scheduled Apex

A **Scheduled Apex** can run every morning to:

- Email manager a list of today's deliveries.
- Provide daily snapshot of:
    - New deliveries
    - Pending deliveries
    - Late deliveries

Example:

```
global class DailyDeliverySummaryScheduler implements Schedulable {
   global void execute(SchedulableContext sc) {
      // query today's deliveries, send email summary
   }
}
```

Then scheduled via Salesforce UI (e.g., 8 AM daily).

---

10. Future Methods

Future methods allow external callouts or heavy logic to run asynchronously.

Example use case (for future version):

- **Call an external logistics / route optimization API**
  to calculate best route for the Delivery Agent.

```
@future(callout=true)
public static void sendDeliveryDataToExternalSystem(Id deliveryId){
   // perform HTTP callout
}
```

---

11. Exception Handling

To ensure stability, try–catch blocks are used:

- Wrap SOQL and business rules in try{} / catch(Exception e) blocks.
- Log exceptions or show user-friendly error messages.

```
try {
```

```
    // logic
} catch(Exception e){
    System.debug('Error in Delivery Trigger: ' + e.getMessage());
}
```

---

12. Test Classes

Every Apex class and trigger requires test coverage:

- Create test data:
    - Insert Patient, Delivery Agent, Delivery records.
- Insert/update Delivery to trigger logic.
- Use System.assert() to verify:
    - Status updates correctly
    - Errors thrown for invalid data
    - Batch/Queueable behavior

```
@isTest
private class DeliveryTriggerTest {
    @isTest static void testOverlapValidation(){
        // insert deliveries, assert errors
    }
}
```

---

13. Asynchronous Processing Overview

The project is designed to optionally support advanced async patterns:

- **Batch Apex** → Processes large numbers of deliveries (ex: overnight Late marking).
- **Queueable Apex** → Handles queued tasks like discount calculation.
- **Future Methods** → Integrates with external systems (logistics, pharmacy systems).
- **Scheduled Apex** → Sends recurring summary emails and monitoring reports.

These ensure:

- Scalability
- Performance

- Integration readiness

✅ **Phase 5 Summary**

In Phase 5, the Pharmacy Delivery CRM's **developer-facing design** is documented:

✓ Use of Apex classes for reusable delivery logic
✓ Use of Apex triggers (with handler pattern)
✓ Use of SOQL, collections, control logic
✓ Use of Batch, Queueable, Scheduled, and Future methods (design-ready)
✓ Test classes for quality assurance
✓ Full async processing model for future scaling

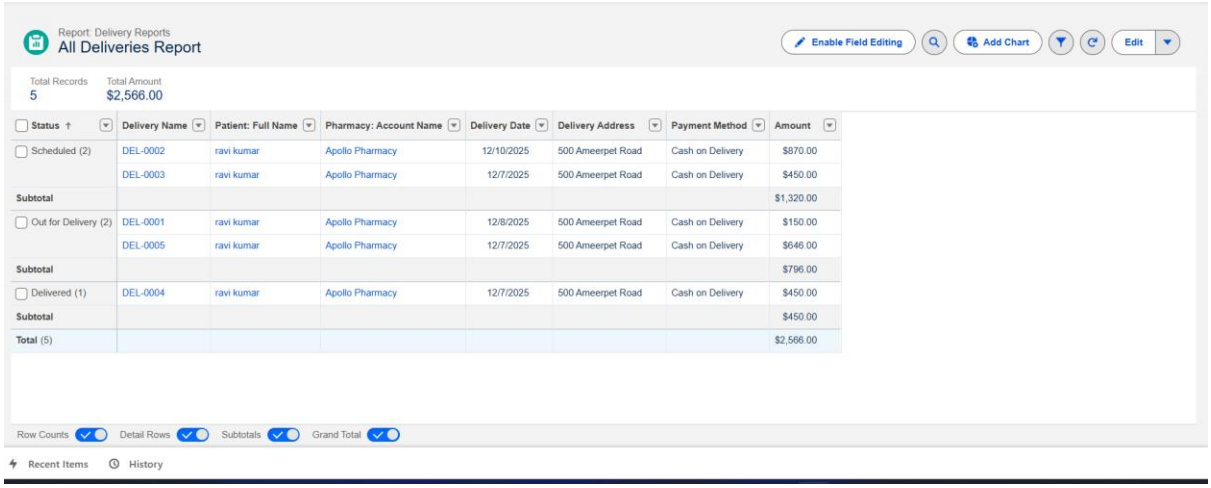
Even if you do **not implement all Apex in your dev org**, this phase shows that your system is **designed to be extendable with code**, which is excellent for academic and viva purposes.

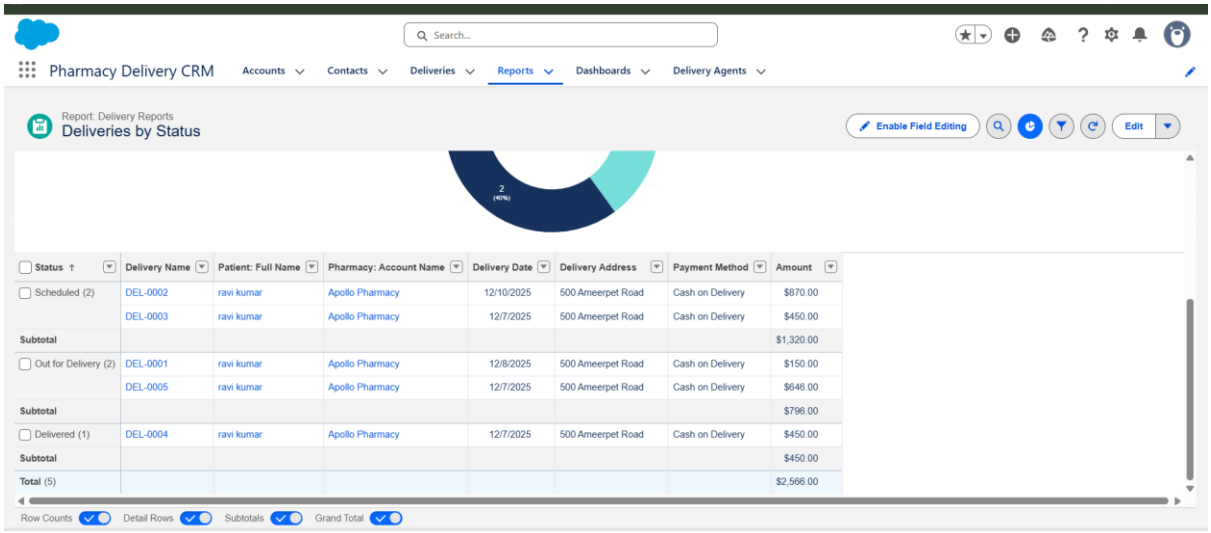
5.8 Summary of Phase 5

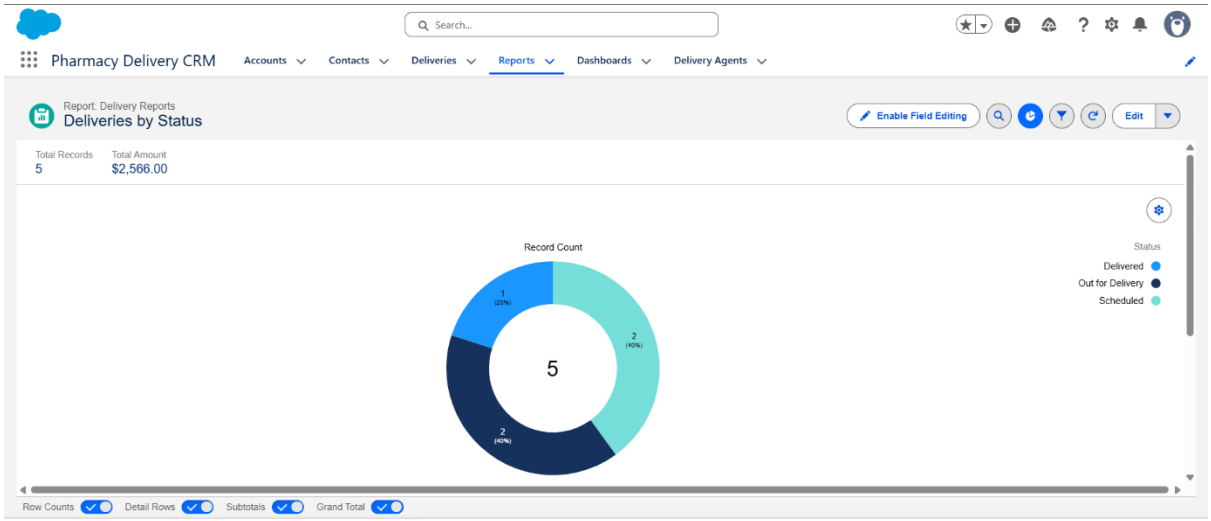
This phase successfully delivered:

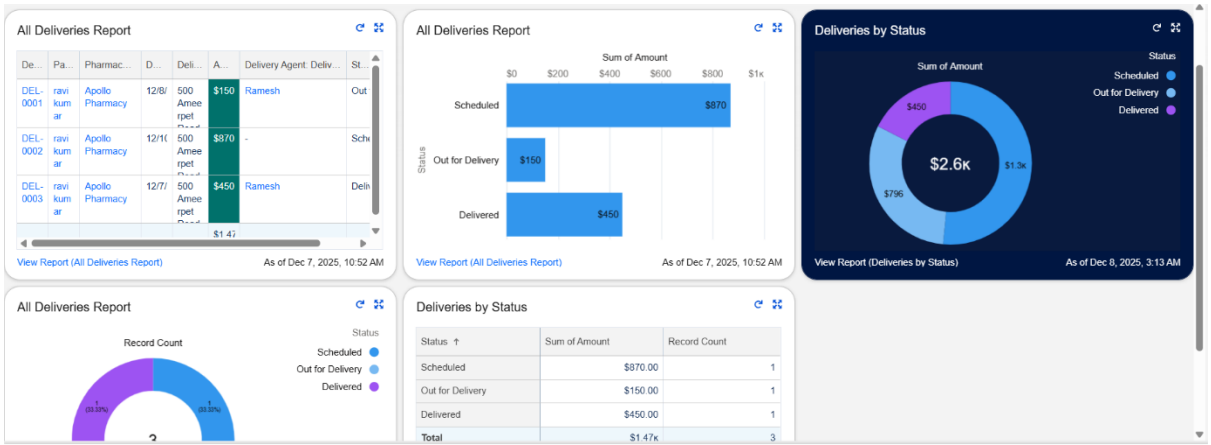
✓ Comprehensive delivery reports
✓ Status-based visual analytics
✓ Donut chart for delivery performance
✓ Table component for detailed list view
✓ Business insights for management
✓ Scalable dashboard structure
✓ Reporting base for future analytics

The reporting and dashboard system now provides **real-time operational visibility**, supporting decisions and improving delivery operations.


Report: Delivery Reports
All Deliveries Report
Enable Field Editing
Add Chart
Edit
Total Records
5
Total Amount
$2,566.00
Status
Delivery Name
Patient: Full Name
Pharmacy: Account Name
Delivery Date
Delivery Address
Payment Method
Amount
Scheduled (2)
DEL-0002
ravi kumar
Apollo Pharmacy
12/10/2025
500 Ameerpet Road
Cash on Delivery
$870.00
DEL-0003
ravi kumar
Apollo Pharmacy
12/7/2025
500 Ameerpet Road
Cash on Delivery
$450.00
Subtotal
$1,320.00
Out for Delivery (2)
DEL-0001
ravi kumar
Apollo Pharmacy
12/8/2025
500 Ameerpet Road
Cash on Delivery
$150.00
DEL-0005
ravi kumar
Apollo Pharmacy
12/7/2025
500 Ameerpet Road
Cash on Delivery
$646.00
Subtotal
$796.00
Delivered (1)
DEL-0004
ravi kumar
Apollo Pharmacy
12/7/2025
500 Ameerpet Road
Cash on Delivery
$450.00
Subtotal
$450.00
Total (5)
$2,566.00
Row Counts
Detail Rows
Subtotals
Grand Total
Recent Items
History


Search...
Pharmacy Delivery CRM
Accounts
Contacts
Deliveries
Reports
Dashboards
Delivery Agents
Report: Delivery Reports
Deliveries by Status
Enable Field Editing
Edit
2
(40%)
Status
Delivery Name
Patient: Full Name
Pharmacy: Account Name
Delivery Date
Delivery Address
Payment Method
Amount
Scheduled (2)
DEL-0002
ravi kumar
Apollo Pharmacy
12/10/2025
500 Ameerpet Road
Cash on Delivery
$870.00
DEL-0003
ravi kumar
Apollo Pharmacy
12/7/2025
500 Ameerpet Road
Cash on Delivery
$450.00
Subtotal
$1,320.00
Out for Delivery (2)
DEL-0001
ravi kumar
Apollo Pharmacy
12/8/2025
500 Ameerpet Road
Cash on Delivery
$150.00
DEL-0005
ravi kumar
Apollo Pharmacy
12/7/2025
500 Ameerpet Road
Cash on Delivery
$646.00
Subtotal
$796.00
Delivered (1)
DEL-0004
ravi kumar
Apollo Pharmacy
12/7/2025
500 Ameerpet Road
Cash on Delivery
$450.00
Subtotal
$450.00
Total (5)
$2,566.00
Row Counts
Detail Rows
Subtotals
Grand Total


Search...
Pharmacy Delivery CRM
Accounts
Contacts
Deliveries
Reports
Dashboards
Delivery Agents
Report: Delivery Reports
Deliveries by Status
Enable Field Editing
Edit
Total Records
5
Total Amount
$2,566.00
Record Count
1
(20%)
2
(40%)
5
2
(40%)
Status
Delivered
Out for Delivery
Scheduled
Row Counts
Detail Rows
Subtotals
Grand Total


All Deliveries Report
De...
Pa...
Pharmac...
D...
Deli...
A...
Delivery Agent: Deliv...
St...
DEL-0001
ravi kumar
Apollo Pharmacy
12/8/
500 Ameerpet
$150
Ramesh
Out
DEL-0002
ravi kumar
Apollo Pharmacy
12/1(
500 Ameerpet
$870
-
Sch
DEL-0003
ravi kumar
Apollo Pharmacy
12/7/
500 Ameerpet
$450
Ramesh
Deli
View Report (All Deliveries Report)
As of Dec 7, 2025, 10:52 AM
All Deliveries Report
Sum of Amount
$0
$200
$400
$600
$800
$1K
Status
Scheduled
$870
Out for Delivery
$150
Delivered
$450
View Report (All Deliveries Report)
As of Dec 7, 2025, 10:52 AM
Deliveries by Status
Sum of Amount
Status
Scheduled
Out for Delivery
Delivered
$450
$2.6K
$1.3K
$796
View Report (Deliveries by Status)
As of Dec 8, 2025, 3:13 AM
All Deliveries Report
Record Count
Status
Scheduled
Out for Delivery
Delivered
Deliveries by Status
Status
Sum of Amount
Record Count
Scheduled
$870.00
1
Out for Delivery
$150.00
1
Delivered
$450.00
1
Total
$1.47K
3
Recent Items
History