

Tubestream Central for MITME

Technical Design Document

Version 1.0



REVISION HISTORY

Ver.	Date	Prepared by	Reviewed by	Approved by	Description
1.0	06/30/2025	Agustinus Winata	Vincent Sew Hee		Initial document creation

DETAILED CHANGE LOG

Version No.	Changes Made (Description)
1.0	Initial document creation – no change to log.

DOCUMENT SIGN-OFF

	Prepared by (Inosoft)	Reviewed and Approved by (TDC)	Reviewed and Approved by (BU/MISI)
Date	06/30/2025		
Signature			
Name	Agustinus Winata	Vincent Sew Hee	

TABLE OF CONTENT

1. Introduction	33
2. Specific Configuration	33
2.1 Server Configuration	33
2.2 Database Configuration	34
2.3 Authentication and Security	34
2.4 Logging and Monitoring.....	35
2.5 API Configuration	35
3. Coding Standard.....	35
3.1 Basic Coding Standard.....	35
3.2 Namespace and Use Declarations	36
3.3 Classes, Properties, and Methods	36
3.4 Control Structure.....	37
3.5 Coding Example.....	37
3.6 Security Consideration	37
3.7 Code Readability and Maintainability.....	37
3.8 Common Mistakes	38
3.9 Migration from PSR-2 to PSR-12.....	38
3.10 Tools for Enforcing PSR-12.....	38
3.11 Best Practice.....	40
3.12 Error Handling and Logging	42
3.13 Performance Optimization.....	43
3.14 Advanced OOP Concept	43
3.15 Design Pattern in PHP.....	43
3.16 Dependency Injection and Service Containers	43
3.17 Testing and Debugging Best Practices	44
3.18 Continuous Integration and Deployment (CI/CD).....	44
3.19 Documentation and Code Comment	44
4. Interaction Protocols.....	44
4.1 Client-Server Communication	44
4.2 API Request and Response Flow	44
4.3 Authentication Flow.....	44
5. Component Design.....	44
5.1 Quality & HSE	45
5.1.1 NCR	45
5.1.1.1 User Interface	46
5.1.1.1.1 NCR Form.....	46
5.1.1.1.2 Detail NCR	46
5.1.1.2 Security.....	46
5.1.1.3 Application Services.....	46
5.1.1.3.1 Initial Data Retrieval.....	46
5.1.1.3.2 NCR Created.....	46
5.1.1.3.3 NCR Lifecycle Events.....	46
5.1.1.3.4 Mill Audit Service	47
5.1.1.4 Database	47
5.1.2 Inspection Report.....	48

5.1.2.1 User Interface.....	48
5.1.2.1.1 Create Inspection.....	49
5.1.2.1.2 Inspect Screening.....	49
5.1.2.1.3 Review Inspection	49
5.1.2.1.4 Move Rack.....	49
5.1.2.2 Security.....	49
5.1.2.3 Application Service	49
5.1.2.3.1 EventInspectionCreated.....	49
5.1.2.3.2 Notification	49
5.1.2.3.3 EventInspectionReviewed	49
5.1.2.3.4 EventScreeningCompleted.....	50
5.1.2.3.5 EventInspectionCompleted.....	50
5.1.2.3.6 YardService	50
5.1.2.3.7 LedgerService	50
5.1.2.3.8 InspectionService	50
5.1.2.3.9 UserService	50
5.1.2.3.10 ItemTallyService.....	50
5.1.2.4 Database.....	50
5.1.3 Quarantine Lot.....	51
5.1.3.1 User Interface.....	51
5.1.3.1.1 Create Quarantine Form	51
5.1.3.1.2 New Quarantine Form	51
5.1.3.1.3 New Release Quarantine Form	52
5.1.3.1.4 Release Quarantine Form	52
5.1.3.2 Security.....	52
5.1.3.3 Application Services	52
5.1.3.3.1 EventQuarantineCreated.....	52
5.1.3.3.2 EventAssignQuarantine.....	52
5.1.3.3.3 EventAssignRelease.....	52
5.1.3.3.4 EventReleaseQuarantine	52
5.1.3.3.5 YardService	53
5.1.3.3.6 LedgerService	53
5.1.3.3.7 QuarantineService	53
5.1.3.3.8 InspectionService	53
5.1.3.4 Database.....	53
5.2 Commercial.....	53
5.2.1 Account Balance.....	53
5.2.1.1 User Interface.....	54
5.2.1.2 Security.....	54
5.2.1.3 Application Service	54
5.2.1.3.1 User Service	54
5.2.1.3.2 Customer Service	54
5.2.1.3.3 Account Balance Service.....	54
5.2.1.3.4 Itemvims Service	54
5.2.1.3.5 Item Card Balance Service.....	54
5.2.1.3.6 Item Service.....	54
5.2.1.3.7 Ordervims Service.....	54

5.2.1.3.8 Forecast Service	54
5.2.1.4 Database.....	54
5.2.2 Commitment.....	55
5.2.2.1 User Interface.....	55
5.2.2.1.1 Create Commitment Form	55
5.2.2.1.2 Close Short Commitment	56
5.2.2.2 Security	56
5.2.2.3 Application Service	56
5.2.2.3.1 Create Commitment	56
5.2.2.3.1.1 User Service.....	56
5.2.2.3.1.2 Customer Service Data.....	56
5.2.2.3.1.3 CustomerField Service.....	56
5.2.2.3.1.4 Delivery Term Service	56
5.2.2.3.1.5 Item Service.....	56
5.2.2.3.1.6 Range Service	56
5.2.2.3.1.7 Item VMI Service	56
5.2.2.3.1.8 ItemCardBalance Service	56
5.2.2.3.1.9 OrderVMI Service	56
5.2.2.3.1.10 Commitment Service.....	56
5.2.2.3.1.11 Notification Service	56
5.2.2.3.2 Close Short Commitment.....	57
5.2.2.3.2.1 User Service.....	57
5.2.2.3.2.2 Commitment Service.....	57
5.2.2.3.2.3 ItemCardBalance Service.....	57
5.2.2.3.2.4 OrderVMI Service	57
5.2.2.3.2.5 Notification Service	57
5.2.2.4 Database	57
5.2.3 Forecast.....	58
5.2.3.1 User Interface.....	58
5.2.3.1.1 Create Forecast Form	58
5.2.3.1.2 Dashboard Forecast.....	58
5.2.3.1.3 Sidebar Forecast.....	58
5.2.3.2 Security	58
5.2.3.3 Application Services	59
5.2.3.3.1 Form Data and Filtering	59
5.2.3.3.1.1 Customer Service	59
5.2.3.3.1.2 Item Service.....	59
5.2.3.3.1.3 Get Dashboard Forecast.....	59
5.2.3.3.1.4 Advance Filter Forecast	59
5.2.3.3.2 Forecast Operation.....	59
5.2.3.3.2.1 Submit Forecast → SaveForecast().....	59
5.2.3.3.2.2 Move Forecast → MoveForecast().....	59
5.2.3.3.2.3 Duplicate Forecast → DuplicateForecast()	59
5.2.3.3.2.4 Delete Forecast → DeleteForecast().....	59
5.2.3.3.2.5 Import Forecast → SubmitImportForecast()	59
5.2.3.3.2.6 Delete Forecast by Well → DeleteForecastByForecast() ..	59

5.2.3.3.2.7 Get Sidebar	59
Forecast → getForecastListWithPeriodByGroup()	59
5.2.3.3.3 Supporting Logic	59
5.2.3.3.3.1 OrderVims Service	59
5.2.3.4 Database	59
5.2.4 Ownership Transfer	60
5.2.4.1 User Interface	60
5.2.4.1.1 Create Ownership Transfer Form.....	60
5.2.4.1.2 Toggle Invoice Ownership Transfer Form	61
5.2.4.2 Security	61
5.2.4.3 Application Services	61
5.2.4.3.1 Create Ownership Transfer.....	61
5.2.4.3.1.1 Customer Service	61
5.2.4.3.1.2 PrivilegeAccessibility Service.....	61
5.2.4.3.1.3 User Service.....	61
5.2.4.3.1.4 StoreBusiness Service	61
5.2.4.3.1.5 CustomField Service	61
5.2.4.3.1.6 WellName Service.....	61
5.2.4.3.1.7 CollectionPlan Service	61
5.2.4.3.1.8 Item Service.....	61
5.2.4.3.1.9 CustomFieldItem Service	61
5.2.4.3.1.10 Ledger Service	61
5.2.4.3.1.11 SaveOwnershipTransfer Service.....	61
5.2.4.3.1.12 EventSaveOwnershipTransfer	61
5.2.4.3.1.13 Notification Service.....	62
5.2.4.3.2 Toggle Invoice Ownership Transfer Form.....	62
5.2.4.3.2.1 OwnershipTransfer Service.....	62
5.2.4.3.2.2 Commitment Service.....	62
5.2.4.3.2.3 Item Service	62
5.2.4.3.2.4 AccBalance Service	62
5.2.4.3.2.5 CreateAccbalance Service	62
5.2.4.3.2.6 ToggleInvoice Service	62
5.2.4.3.2.7 EventToggleInvoice	62
5.2.4.3.2.8 User Service	62
5.2.4.4 Database	62
5.3 Procurement	63
5.3.1 Lot Info	63
5.3.1.1 User Interface	63
5.3.1.1.1 Create Lot Info Form.....	63
5.3.1.1.2 List Table Lot Info.....	63
5.3.1.1.3 Detail Lot Info	63
5.3.1.2 Security	63
5.3.1.3 Application Service	64
5.3.1.3.1 LotInfo Service.....	64
5.3.1.3.2 MDR Service.....	64
5.3.1.3.3 Event Service.....	64
5.3.1.3.4 Notification Service.....	64

5.3.1.4 Database.....	64
5.3.2 Mill Set Up.....	65
5.3.2.1 User Interface.....	65
5.3.2.2 Security.....	65
5.3.2.3 Controller.....	65
5.3.2.3.1 Mill Tag Added.....	65
5.3.2.3.2 Ino Mill Advisor Model.....	65
5.3.2.3.3 Ino Itemtrans Model.....	65
5.3.2.3.4 Ino Cpo.....	65
5.3.2.4 Database.....	66
5.4 Inventory	66
5.4.1 Stock on Hand	66
5.4.1.1 User Interface.....	66
5.4.1.1.1 List TableSOH	66
5.4.1.1.2 List Detail Ledger	66
5.4.1.1.3 View Transaction in Process	66
5.4.1.2 Security.....	66
5.4.1.3 Application Services	67
5.4.1.3.1 LedgerService.....	67
5.4.1.3.2 InspectionService	67
5.4.1.3.3 TransferService	67
5.4.1.3.4 QuarantineService	67
5.4.1.3.5 ProductionService.....	67
5.4.1.3.6 ReturnService	67
5.4.1.4 Database.....	67
5.4.2 Transfer	68
5.4.2.1 User Interface.....	68
5.4.2.1.1 Save Transfer	68
5.4.2.1.2 Create Dispatch	68
5.4.2.1.3 Create Receive	68
5.4.2.2 Security	69
5.4.2.3 Application Services	69
5.4.2.3.1 User Service.....	69
5.4.2.3.2 Transfer Service	69
5.4.2.3.3 EventTransferCreated	69
5.4.2.3.4 Dispatch Service	69
5.4.2.3.5 EventDispatchCreated	69
5.4.2.3.6 EventReceiveCreated	69
5.4.2.3.7 Ledger Service	69
5.4.2.3.8 Notification Service	69
5.4.2.4 Database.....	70
5.5 Vendor Management	70
5.5.1 User Interface	70
5.5.1.1 Create 3 rd Party Instruction Form.....	70
5.5.1.2 Detail 3 rd Party Instruction	70
5.5.2 Security.....	70
5.5.3 Application Services.....	71

5.5.3.1 User Service	71
5.5.3.2 Customer Service	71
5.5.3.3 LI Service	71
5.5.3.4 Contractor Service	71
5.5.3.5 Transaction Info Helper	71
5.5.3.6 LI Currency Service.....	71
5.5.3.7 LISI Service.....	71
5.5.3.8 3 rd Party Created	71
5.5.3.9 Vendor Invoice Added	71
5.5.3.10 Receive All.....	71
5.5.4 Database	71
5.6 Global Mill Advisor.....	72
5.6.1 Mill Audit Request.....	72
5.6.1.1 User Interface.....	72
5.6.1.1.1 Mill Audit Request Form.....	72
5.6.1.1.2 Detail Mill Audit Request	73
5.6.1.1.3 Detail Mill NCR.....	73
5.6.1.2 Security.....	73
5.6.1.3 Application Services	73
5.6.1.3.1 Initial Data and Form Options	73
5.6.1.3.2 Mill Audit Request Creation.....	73
5.6.1.3.3 Mill Audit Request Lifecycle	73
5.6.1.3.4 NCR Handling Post-Audit.....	74
5.6.1.4 Database.....	74
5.6.2 Mill Listing.....	74
5.6.2.1 User Interface.....	74
5.6.2.1.1 Create Mill Form.....	74
5.6.2.1.2 Product Range Form	75
5.6.2.1.3 Review Mill Listing.....	75
5.6.2.2 Security	75
5.6.2.3 Application Services	75
5.6.2.3.1 Display Form Option Services	75
5.6.2.3.2 Submit Product Service.....	75
5.6.2.3.3 Save Mill Listing Service	75
5.6.2.3.4 Detail Service.....	75
5.6.2.4 Domain Services.....	75
5.6.2.4.1 Comparison Service	75
5.6.2.4.2 Mill Listing Service.....	75
5.6.2.5 Database	75
5.6.3 Mill NCR	76
5.6.3.1 User Interface.....	76
5.6.3.2 Security	76
5.6.3.3 Application Services	76
5.6.3.3.1 Mill Audit Service.....	76
5.6.3.3.2 Mill NCR Service	76
5.6.3.3.3 Mill NCR Response Submitted	76
5.6.3.4 Database.....	77

5.6.4 Mill MoC.....	77
5.6.4.1 User Interface.....	77
5.6.4.2 Security	77
5.6.4.3 Application Services	77
5.6.4.3.1 Mill Audit Service.....	77
5.6.4.3.2 Mill Listing Service.....	77
5.6.4.3.3 Mill MoC Submitted.....	77
5.6.4.3.4 Mill MoC Service	78
5.6.4.3.5 Evaluation of Risk Control Submitted	78
5.6.4.3.6 Approver Assigned.....	78
5.6.4.3.7 Approver Accepted	78
5.7 Setup	78
5.7.1 User	78
5.7.1.1 User Interface.....	78
5.7.1.2 Security	78
5.7.1.3 Controller.....	78
5.7.1.3.1 Admin Controller	78
5.7.1.3.2 Client Controller	79
5.7.1.4 Database.....	79
5.7.2 Privilege	79
5.7.2.1 User Interface	79
5.7.2.2 Security	79
5.7.2.3 Controller.....	79
5.7.2.4 Database	80
5.7.3 Tablet Access	80
5.7.3.1 User Interface	80
5.7.3.2 Security	80
5.7.3.3 Controller.....	80
5.7.3.3.1 Tablet Access Submitted	80
5.7.3.3.2 Ino Yard Model.....	80
5.7.3.3.3 Ino IMEI Model	80
5.7.3.4 Database	80
5.7.4 Contractor	81
5.7.4.1 User Interface	81
5.7.4.2 Security	81
5.7.4.3 Controller.....	81
5.7.4.4 Database	81
5.7.5 Port.....	82
5.7.5.1 User Interface	82
5.7.5.2 Security	82
5.7.5.3 Controller.....	82
5.7.5.4 Database	82
5.7.6 Yard	82
5.7.6.1 User Interface	82
5.7.6.2 Security	83
5.7.6.3 Controller.....	83
5.7.6.4 Database	83

5.8 Support.....	83
5.8.1 User Interface	83
5.8.1.1 Support.....	83
5.8.1.2 Support Detail	83
5.8.2 Security.....	84
5.8.3 Application Services.....	84
5.8.3.1 User Service	84
5.8.3.2 Jira Service.....	84
5.8.3.3 JWT Service.....	84
5.8.4 API Component	84
5.8.5 Database	84
6. API Endpoints.....	84
6.1 Quality & HSE	85
6.1.1 SPR / NCR.....	85
6.1.1.1 List/Index Page	85
6.1.1.2 Create & Edit NCR	85
6.1.1.3 Detail 3 rd Party Instruction.....	85
6.1.2 MoC	88
6.1.3 Inspection Report.....	88
6.1.3.1 Ageing Inspection	88
6.1.3.2 Create and Edit Inspection	88
6.1.3.3 View Inspection Data	91
6.1.3.4 Close Short, Terminate, and Reopen Inspection	92
6.1.3.5 Scope of Work.....	92
6.1.3.6 Screening Process	93
6.1.4 Quarantine Lot.....	98
6.1.4.1 Create & Edit Quarantine	98
6.1.4.2 Detail & List Quarantine	99
6.1.4.3 Assign New Quarantine.....	99
6.1.4.4 Assign Release & Release Quarantine.....	100
6.1.4.5 Close Short & Terminate Quarantine	100
6.2 Commercial.....	101
6.2.1 Account Balance.....	101
6.2.1.1 Account Balance Header	101
6.2.1.2 Account Balance Level Tabs & Items.....	101
6.2.1.3 Account Balance Level Subrow Item & Detail.....	101
6.2.2 Commitment.....	102
6.2.2.1 Public Commitment.....	102
6.2.2.2 Excel Generation	103
6.2.2.3 General Routes	103
6.2.2.4 Approval.....	104
6.2.2.5 Commitment Management Routes	105
6.2.2.6 Read-Only.....	106
6.2.2.7 Shared	107
6.2.2.8 User Authentication.....	108
6.2.3 Forecast.....	108
6.3 Procurement	109

6.3.1 Lot Info.....	109
6.3.1.1 Lot Info List	109
6.3.1.2 Create Lot Info	109
6.3.1.3 Lot Info Detail.....	109
6.3.1.4 MDR	110
6.3.1.5 Tally List	110
6.3.1.6 Inspection.....	111
6.3.2 Mill Set Up	111
6.4 Inventory	111
6.4.1 Stock on Hand	111
6.4.2 Transfer	112
6.4.2.1 Authentication.....	112
6.4.2.2 Transfer Management	112
6.4.2.3 Document Generation.....	113
6.4.2.4 Delivery Note	114
6.4.2.5 Utility.....	114
6.4.2.6 ERP Integration.....	115
6.4.2.7 Tablet Operation	115
6.4.2.8 Internal Notes.....	116
6.4.2.9 Attachments	116
6.4.2.10 Others	117
6.5 Vendor Management	117
6.5.1 List/Index Page	117
6.5.2 Create & Edit 3 rd Party Instruction	118
6.5.3 Detail 3rd Party Instruction	119
6.6 Global Mill Advisor.....	120
6.6.1 Mill Audit Request.....	120
6.6.1.1 List/Index Page	120
6.6.1.2 Create & Edit Mill Audit Request	121
6.6.1.3 Detail Mill Audit Request.....	121
6.6.2 Mill Listing.....	123
6.6.2.1 Mill Listing APIs	123
6.6.2.2 Audit Related APIs	123
6.6.2.3 MoC APIs.....	124
6.6.3 Mill NCR.....	124
6.6.3.1 List/Index Page.....	124
6.6.3.2 Edit Mill NCR	124
6.6.3.3 Detail Mill NCR.....	124
6.6.4 Mill MoC.....	125
6.6.4.1 List/Index Page.....	125
6.6.4.2 Create & Edit MoC.....	125
6.6.4.3 Detail Mill MoC	125
6.7 Setup.....	126
6.7.1 User.....	126
6.7.2 Privilege	127
6.7.3 Tablet Access	127
6.7.3.1 List/Index Page.....	127

6.7.3.2 Create & Edit Tablet Setup.....	127
6.7.3.3 Detail Tablet Setup	128
6.7.4 Contractor.....	128
6.7.5 Port.....	129
6.7.6 Item	130
6.7.7 OD	130
6.7.8 Grade.....	131
6.7.9 Connection.....	131
6.7.10 Special Condition	132
6.7.11 Yard.....	132
6.8 Support.....	133
6.8.1 Jira API	133
6.8.2 Support API.....	134
7. Data Flow Diagram	135
7.1 Market Index.....	135
7.1.1 Market Index - Data Flow Diagram Level 0.....	135
7.2 Quality & HSE.....	136
7.2.1 SPR / NCR	136
7.2.1.1 NCR - Data Flow Diagram Level 0.....	136
7.2.1.2 NCR - Data Flow Diagram Level 1.....	137
7.2.2 MoC.....	139
7.2.2.1 MoC - Data Flow Diagram Level 0	139
7.2.2.2 MoC - Data Flow Diagram Level 1	140
7.2.3 HSE.....	142
7.2.3.1 HSE - Data Flow Diagram Level 0	142
7.2.4 IVMS	143
7.2.4.1 IVMS - Data Flow Diagram Level 0	143
7.2.5 Lesson Learned	144
7.2.5.1 Lesson Learned - Data Flow Diagram Level 0	144
7.2.6 SDS	144
7.2.6.1 SDS - Data Flow Diagram Level 0	144
7.2.7 Audit > Schedule.....	145
7.2.7.1 Audit > Schedule - Data Flow Diagram Level 0	145
7.2.8 Audit > Audit Report	145
7.2.8.1 Audit > Audit Report - Data Flow Diagram Level 0	145
7.2.9 Audit > TPI Qualification.....	146
7.2.9.1 TPI Qualification - Data Flow Diagram Level 0	146
7.2.10 Inspection > Inspection Report	146
7.2.10.1 Inspection > Inspection Report - Data Flow Diagram Level 0.....	146
7.2.10.2 Inspection > Inspection Report - Data Flow Diagram Level 1.....	147
7.2.11 Inspection > Quarantine Lot	149
7.2.11.1 Inspection > Quarantine Lot - Data Flow Diagram Level 0	149
7.2.11.2 Inspection > Quarantine Lot - Data Flow Diagram Level 1	149
7.2.12 Inspection > Template SoW	151
7.2.12.1 Inspection > Template SoW - Data Flow Diagram Level 0	151
7.3 Commercial.....	151
7.3.1 Contract	151

7.3.1.1 Contract - Data Flow Diagram Level 0.....	151
7.3.2 Account Balance	152
7.3.2.1 Account Balance - Data Flow Diagram Level 0.....	152
7.3.3 Commitment.....	153
7.3.3.1 Commitment - Data Flow Diagram Level 0	153
7.3.4 Forecast.....	153
7.3.4.1 Forecast - Data Flow Diagram Level 0	153
7.3.5 Material Requisition.....	154
7.3.5.1 Material Requisition - Data Flow Diagram Level 0	154
7.3.5.2 Material Requisition - Data Flow Diagram Level 1	154
7.3.6 Return	156
7.3.6.1 Return - Data Flow Diagram Level 0.....	156
7.3.6.2 Return - Data Flow Diagram Level 1.....	156
7.3.7 Ownership Transfer	158
7.3.7.1 Ownership Transfer - Data Flow Diagram Level 0	158
7.3.7.2 Ownership Transfer - Data Flow Diagram Level 1.....	159
7.4 Procurement.....	160
7.4.1 Lot Info.....	160
7.4.1.1 Lot Info - Data Flow Diagram Level 0	160
7.4.1.2 Lot Info - Data Flow Diagram Level 1	160
7.4.2 Open Order	162
7.4.2.1 Open Order - Data Flow Diagram Level 0.....	162
7.4.2.2 Open Order - Data Flow Diagram Level 1.....	162
7.4.3 Mill Set Up.....	164
7.4.3.1 Mill Set Up - Data Flow Diagram Level 0	164
7.4.4 GIMS Inquiry > Search	164
7.4.4.1 GIMS Inquiry > Search - Data Flow Diagram Level 0.....	164
7.4.5 GIMS Inquiry > History	165
7.4.5.1 GIMS Inquiry > History - Data Flow Diagram Level 0.....	165
7.5 Inventory.....	165
7.5.1 Stock on Hand.....	165
7.5.1.1 Stock on Hand - Data Flow Diagram Level 0	165
7.5.2 Adjustment.....	166
7.5.2.1 Adjustment - Data Flow Diagram Level 0	166
7.5.3 Transfer > Port / Yard	166
7.5.3.1 Transfer > Port / Yard - Data Flow Diagram Level 0.....	166
7.5.3.2 Transfer > Port / Yard - Data Flow Diagram Level 1.....	167
7.5.4 Transfer > Allocation.....	168
7.5.4.1 Transfer > Allocation - Data Flow Diagram Level 0	168
7.5.5 Transfer > Rack.....	168
7.5.5.1 Transfer > Rack - Data Flow Diagram Level 0	168
7.6 Vendor Management	169
7.6.1 3 rd Party Instruction	169
7.6.1.1 3 rd Party Instruction - Data Flow Diagram Level 0.....	169
7.6.1.2 3 rd Party Instruction - Data Flow Diagram Level 1.....	169
7.7 Production.....	170
7.7.1 Production.....	170

7.7.1.1 Production - Data Flow Diagram Level 0	170
7.7.1.2 Production - Data Flow Diagram Level 1	170
7.8 KPI & Report.....	171
7.8.1 KPI	171
7.8.1.1 Account Summary - Data Flow Diagram Level 0	171
7.8.1.2 Yard Info - Data Flow Diagram Level 0	172
7.8.1.3 Commitment Statistic - Data Flow Diagram Level 0.....	172
7.8.1.4 Commitments Ageing - Data Flow Diagram Level 0	173
7.8.1.5 Invoice - Data Flow Diagram Level 0	173
7.8.1.6 MR Order Notification - Data Flow Diagram Level 0	173
7.8.1.7 Material Requisition Compliance to issue rig / jetty - Data Flow Diagram Level 0.....	174
7.8.1.8 Stock View - Data Flow Diagram Level 0.....	174
7.8.1.9 Inventory Profile - Data Flow Diagram Level 0.....	175
7.8.1.10 Material Requisition Statistic - Data Flow Diagram Level 0	175
7.8.1.11 Ageing - Data Flow Diagram Level 0	176
7.8.1.12 Material Requisition Compliance - Data Flow Diagram Level 0	176
7.8.1.13 Stock Management – Open End - Data Flow Diagram Level 0	177
7.8.1.14 Stock Management – Closed End - Data Flow Diagram Level 0	177
7.8.2 Report	178
7.8.2.1 Adnoc Requirement.....	178
7.8.2.1.1 ADNOC ERP Log - Data Flow Diagram Level 0	178
7.8.2.1.2 Material Requisition - Data Flow Diagram Level 0.....	178
7.8.2.1.3 Rig Return - Data Flow Diagram Level 0.....	178
7.8.2.1.4 Stock Aging - Data Flow Diagram Level 0	179
7.8.2.1.5 Stock View - Data Flow Diagram Level 0	179
7.8.2.1.6 Transaction Report - Data Flow Diagram Level 0	180
7.8.2.2 Commercial	180
7.8.2.2.1 ACCS Commitment Reports - Data Flow Diagram Level 0	180
7.8.2.2.2 Ageing Detail Report - Data Flow Diagram Level 0	181
7.8.2.2.3 Ageing Recap - Data Flow Diagram Level 0.....	181
7.8.2.2.4 Commitment Ageing Recap - Data Flow Diagram Level 0.....	182
7.8.2.2.5 Dispatch Reports - Data Flow Diagram Level 0.....	182
7.8.2.2.6 Material Requisition - Data Flow Diagram Level 0.....	183
7.8.2.2.7 Material Requisition Invoice - Data Flow Diagram Level 0	183
7.8.2.2.8 OCTG Commitment Reports - Data Flow Diagram Level 0	183
7.8.2.2.9 Return Creation - Data Flow Diagram Level 0	184
7.8.2.3 Inventory	184
7.8.2.3.1 Inventory Report - Data Flow Diagram Level 0	184
7.8.2.3.2 Inventory Review - Data Flow Diagram Level 0	185
7.8.2.3.3 Stock Summary - Data Flow Diagram Level 0	185
7.8.2.4 Operational Report	185
7.8.2.4.1 Movement Detail Report - Data Flow Diagram Level 0	185
7.8.2.4.2 Movement Report - Data Flow Diagram Level 0	186
7.8.2.4.3 Ownership Transfer & Return Ledger Report - Data Flow Diagram Level 0	186
7.8.2.5 Quality & HSE	187

7.8.2.5.1 Damaged Inspection Result - Data Flow Diagram Level 0	187
7.8.2.5.2 Inspection Creation - Data Flow Diagram Level 0.....	187
7.8.2.6 System Report.....	188
7.8.2.6.1 Inactive User - Data Flow Diagram Level 0	188
7.8.2.6.2 List Super Admin - Data Flow Diagram Level 0	188
7.8.2.6.3 Login Recap - Data Flow Diagram Level 0.....	189
7.8.2.7 Traceability Report.....	189
7.8.2.7.1 Batch Movement Report - Data Flow Diagram Level 0.....	189
7.8.2.7.2 Traceability Report - Data Flow Diagram Level 0.....	189
7.9 Resources.....	190
7.9.1 Resources	190
7.9.1.1 Resources - Data Flow Diagram Level 0	190
7.10 Global Mill Advisor	191
7.10.1 Mill Audit Request	191
7.10.1.1 Mill Audit Request - Data Flow Diagram Level 0	191
7.10.1.2 Mill Audit Request - Data Flow Diagram Level 1.....	191
7.10.2 Mill Listing	193
7.10.2.1 Mill Listing - Data Flow Diagram Level 0.....	193
7.10.3 Mill NCR.....	194
7.10.3.1 Mill NCR - Data Flow Diagram Level 0	194
7.10.4 Mill MoC	194
7.10.4.1 Mill MoC - Data Flow Diagram Level 0	194
7.10.4.2 Mill MoC - Data Flow Diagram Level 1.....	195
7.11 Set Up.....	196
7.11.1 User	196
7.11.1.1 User - Data Flow Diagram Level 0.....	196
7.11.1.2 User - Data Flow Diagram Level 1.....	197
7.11.2 Privilege.....	197
7.11.2.1 Privilege - Data Flow Diagram Level 0	197
7.11.3 Tablet Access.....	198
7.11.3.1 Tablet Access - Data Flow Diagram Level 0	198
7.11.3.2 Tablet Access - Data Flow Diagram Level 1.....	198
7.11.4 Customer > Customer Info	199
7.11.4.1 Customer > Customer Info - Data Flow Diagram Level 0	199
7.11.4.2 Customer > Customer Info - Data Flow Diagram Level 1.....	200
7.11.5 Customer > Lot Price	201
7.11.5.1 Customer > Lot Price - Data Flow Diagram Level 0	201
7.11.6 Customer > Fees.....	201
7.11.6.1 Customer > Fees - Data Flow Diagram Level 0	201
7.11.7 Customer > Customer Item No.	202
7.11.7.1 Customer > Customer Item No. - Data Flow Diagram Level 0.....	202
7.11.8 Customer > Collection Point.....	202
7.11.8.1 Customer > Collection Point - Data Flow Diagram Level 0	202
7.11.9 Procurement > Contractor	203
7.11.9.1 Procurement > Contractor - Data Flow Diagram Level 0.....	203
7.11.10 Procurement > Port.....	203
7.11.10.1 Procurement > Port - Data Flow Diagram Level 0	203

7.11.11 Inventory > Item	204
7.11.11.1 Inventory > Item - Data Flow Diagram Level 0.....	204
7.11.12 Inventory > OD	204
7.11.12.1 Inventory > OD - Data Flow Diagram Level 0.....	204
7.11.13 Inventory > Grade	205
7.11.13.1 Inventory > Grade - Data Flow Diagram Level 0.....	205
7.11.14 Inventory > Connection	205
7.11.14.1 Inventory > Connection - Data Flow Diagram Level 0.....	205
7.11.15 Inventory > Special Condition	206
7.11.15.1 Inventory > Special Condition - Data Flow Diagram Level 0.....	206
7.11.16 Inventory > Yard.....	206
7.11.16.1 Inventory > Yard - Data Flow Diagram Level 0.....	206
7.11.16.2 Inventory > Yard - Data Flow Diagram Level 1.....	207
7.11.17 Inventory > Package	208
7.11.17.1 Inventory > Package - Data Flow Diagram Level 0.....	208
7.11.18 Quality & HSE > Category Customization.....	208
7.11.18.1 Quality & HSE > Category Customization - Data Flow Diagram Level 0	208
7.12 Support.....	209
7.12.1 Support - Data Flow Diagram Level 0	209
7.12.2 Support - Data Flow Diagram Level 1.....	209
8. Database Collection.....	210
8.1 abbreviation.....	211
8.2 actionlist.....	211
8.3 assign_evaluators	212
8.4 auditschedule	213
8.5 ax_soh	214
8.6 axlog	215
8.7 azure_log	216
8.8 batch_trackability_itemcode_matrix	216
8.9 batch_trackability_matrix	217
8.10 calloff_counter	217
8.11 Client	217
8.12 commitment.....	220
8.13 config.....	222
8.14 configuration	222
8.15 connection	223
8.16 contract	223
8.17 contract_doc	224
8.18 contractor	225
8.19 counter	226
8.20 country	227
8.21 d365_inv_arch	227
8.22 d365_inv_log	229
8.23 d365_obs_arch	229
8.24 d365_obs_log.....	232
8.25 ddd_logger	232

8.26 delivery	233
8.27 deliveryterms.....	233
8.28 department.....	234
8.29 doc_format	234
8.30 docs.....	234
8.31 document.....	236
8.32 ehs.....	238
8.33 ehs_target	239
8.34 email_log.....	240
8.35 erp_findim.....	241
8.36 erp_log.....	241
8.37 fc_acc_weighted	242
8.38 fc_d365.....	242
8.39 fc_d365_hist.....	243
8.40 fc_d365_recap.....	243
8.41 fees	244
8.42 forecast	244
8.43 forecast_acc	245
8.44 forecast_acc_hist.....	245
8.45 forecast_d365_arch.....	246
8.46 forecast_mb_hist.....	247
8.47 grade	248
8.48 helps	248
8.49 imei	249
8.50 index	249
8.51 indexval	250
8.52 inquiry	251
8.53 inspection_counter.....	252
8.54 inspection_image.....	252
8.55 inspection_temp	253
8.56 internal_note	254
8.57 inventory_report	254
8.58 invoice.....	256
8.59 invoice_mts.....	257
8.60 invoice_pdf	258
8.61 invreport	258
8.62 item.....	260
8.63 item_counter	263
8.64 item_type	264
8.65 item_type_attribute	264
8.66 itemcard	265
8.67 itemcard_balance	267
8.68 itemcard_rack	270
8.69 itemcpo	273
8.70 iteminspection	275
8.71 itemtally	275
8.72 itemtrans.....	277

8.73 itemvims.....	288
8.74 ivms.....	289
8.75 jira_issue_tickets	291
8.76 jira_log	291
8.77 lessonlearned	292
8.78 li_currency	293
8.79 li_delivery	293
8.80 li_recipients	293
8.81 li_uom.....	294
8.82 loadout_tally.....	294
8.83 locode	295
8.84 log_api	295
8.85 logs.....	296
8.86 mdr.....	297
8.87 mdr_folder	298
8.88 merchandise_balance.....	298
8.89 merchandise_balance_ceiling	299
8.90 merchandise_balance_comm.....	300
8.91 merchandise_balance_comm_raw.....	300
8.92 merchandise_balance_log.....	301
8.93 merchandise_balance_raw	301
8.94 merchandise_balance_raw_hist	302
8.95 merchandise_balance_report.....	303
8.96 merchandise_balance_sc.....	305
8.97 mill	305
8.98 milladvisor	307
8.99 moc	308
8.100 moc_dropdown.....	311
8.101 notification.....	311
8.102 od	312
8.103 openorder_vessel_lotid.....	312
8.104 ordervims	313
8.105 pipesales_BU_setup	315
8.106 pipesales_global_setup.....	316
8.107 pipesales_term_condition	318
8.108 pl_reject_reason.....	318
8.109 po_mts	319
8.110 port.....	319
8.111 privcode.....	320
8.112 privilege	320
8.113 product_level.....	321
8.114 protectormat	321
8.115 quality	321
8.116 quarantine	322
8.117 range.....	323
8.118 recap_listing_ledger	323
8.119 recap_metadata	325

8.120 region.....	325
8.121 revision_document	326
8.122 role.....	326
8.123 salesdata.....	327
8.124 scope_work	329
8.125 scope_work_template.....	329
8.126 signoff	330
8.127 specialcond	332
8.128 spr	332
8.129 spr_causal_factor	335
8.130 spr_causal_factor_old	336
8.131 spr_hse_option.....	336
8.132 spr_notify.....	337
8.133 spr_reason.....	337
8.134 spr_reason_old	337
8.135 spr_user_role	337
8.136 sprv2.....	338
8.137 sr_material	340
8.138 sr_od	340
8.139 sr_type	341
8.140 storage_files	341
8.141 tally_history	342
8.142 tenderdoc	343
8.143 todo	343
8.144 tpicomp.....	344
8.145 transaction_logs.....	344
8.146 transfer_activity	345
8.147 transfer_counter	345
8.148 transfer_receipt	346
8.149 turnrate_hist.....	346
8.150 turnrate_price.....	347
8.151 user_activity.....	348
8.152 user_config.....	348
8.153 user_log.....	349
8.154 userlog.....	349
8.155 users	350
8.156 vendor_order.....	353
8.157 vessel.....	354
8.158 vessel_tracking.....	355
8.159 vims_avgprice	356
8.160 vims_batchprice.....	357
8.161 vims_breakdown	357
8.162 vims_forecast	358
8.163 vims_forecast_hist.....	360
8.164 vims_pdf	361
8.165 vims_recomorder	361
8.166 well_design	362

8.167 yard	362
8.168 yard_balance_report.....	364
8.169 yard_opening_balance_report	365
8.170 yard_supervisor	366

TABLE OF FIGURES

Figure 1. Basic Coding Standard	36
Figure 2. Namespace and Use Declaration.....	36
Figure 3. Classes, Properties, and Methods	36
Figure 4. Control Structure	37
Figure 5. Coding Example	37
Figure 6. Security Consideration	37
Figure 7. Code Readability and Maintainability.....	38
Figure 8. Common Mistakes.....	38
Figure 9. PHP_CODESNIFFER	39
Figure 10. PHP-CS-FIXER	39
Figure 11. Dependency Injection.....	40
Figure 12. SOLID Principle.....	41
Figure 13. Unit Test Code.....	41
Figure 14. Warning and Error.....	42
Figure 15. Handle Exception	42
Figure 16. Error Logging Implementation.....	43
Figure 17. NCR Component Design.....	45
Figure 18. Inspection Report Component Design	48
Figure 19. Quarantine Lot Component Design	51
Figure 20. Account Balance Component Design	53
Figure 21. Commitment Component Design.....	55
Figure 22. Forecast Component Design.....	58
Figure 23. Ownership Transfer Component Design	60
Figure 24. Lot Info Component Design	63
Figure 25. Mill Set Up Component Design.....	65
Figure 26. Stock on Hand Component Design	66
Figure 27. Transfer Component Design.....	68
Figure 28. Vendor Management Component Design	70
Figure 29. Mill Audit Request Component Design	72
Figure 30. Mill Listing Component Design	74
Figure 31. Mill NCR Component Design	76
Figure 32. Mill MoC Component Design	77
Figure 33. Setup User Component Design.....	78
Figure 34. Setup Privilege Component Design	79
Figure 35. Setup Tablet Component Design	80
Figure 36. Setup – Contractor Component Design.....	81
Figure 37. Setup – Port Component Design.....	82
Figure 38. Setup – Yard Component Design	82
Figure 39. Support Component Design	83
Figure 40. Market Index - Data Flow Diagram Level 0.....	135
Figure 41. NCR - Data Flow Diagram Level 0	136
Figure 42. NCR - Data Flow Diagram Level 1	138
Figure 43. MoC - Data Flow Diagram Level 0	140
Figure 44. MoC - Data Flow Diagram Level 1.....	141
Figure 45. HSE - Data Flow Diagram Level 0.....	143

Figure 46. IVMS - Data Flow Diagram Level 0	143
Figure 47. Lesson Learned - Data Flow Diagram Level 0.....	144
Figure 48. SDS - Data Flow Diagram Level 0.....	144
Figure 49. Audit > Schedule - Data Flow Diagram Level 0	145
Figure 50. Audit > Audit Report - Data Flow Diagram Level 0	145
Figure 51. TPI Qualification - Data Flow Diagram Level 0	146
Figure 52. Inspection > Inspection Report - Data Flow Diagram Level 0.....	147
Figure 53. Inspection > Inspection Report - Data Flow Diagram Level 1	148
Figure 54. Inspection > Quarantine Lot - Data Flow Diagram Level 0.....	149
Figure 55. Inspection > Quarantine Lot - Data Flow Diagram Level 1.....	150
Figure 56. Inspection > Template SoW - Data Flow Diagram Level 0	151
Figure 57. Contract - Data Flow Diagram Level 0.....	152
Figure 58. Account Balance - Data Flow Diagram Level 0.....	152
Figure 59. Commitment - Data Flow Diagram Level 0	153
Figure 60. Forecast - Data Flow Diagram Level 0	153
Figure 61. Material Requisition - Data Flow Diagram Level 0	154
Figure 62. Material Requisition - Data Flow Diagram Level 1.....	155
Figure 63. Return - Data Flow Diagram Level 0	156
Figure 64. Return - Data Flow Diagram Level 1.....	157
Figure 65. Ownership Transfer - Data Flow Diagram Level 0.....	158
Figure 66. Ownership Transfer - Data Flow Diagram Level 1	159
Figure 67. Lot Info - Data Flow Diagram Level 0	160
Figure 68. Lot Info - Data Flow Diagram Level 1.....	161
Figure 69. Open Order - Data Flow Diagram Level 0	162
Figure 70. Open Order - Data Flow Diagram Level 1	163
Figure 71. Mill Set Up - Data Flow Diagram Level 0	164
Figure 72. GIMS Inquiry > Search - Data Flow Diagram Level 0	164
Figure 73. GIMS Inquiry > History - Data Flow Diagram Level 0	165
Figure 74. Stock on Hand - Data Flow Diagram Level 0.....	165
Figure 75. Adjustment - Data Flow Diagram Level 0	166
Figure 76. Transfer > Port / Yard - Data Flow Diagram Level 0.....	167
Figure 77. Transfer > Port / Yard - Data Flow Diagram Level 1.....	167
Figure 78. Transfer > Allocation - Data Flow Diagram Level 0	168
Figure 79. Transfer > Rack - Data Flow Diagram Level 0.....	168
Figure 80. 3 rd Party Instruction - Data Flow Diagram Level 0	169
Figure 81. 3 rd Party Instruction - Data Flow Diagram Level 1.....	169
Figure 82. Production - Data Flow Diagram Level 0.....	170
Figure 83. Production - Data Flow Diagram Level 1	171
Figure 84. Account Summary - Data Flow Diagram Level 0.....	172
Figure 85. Yard Info - Data Flow Diagram Level 0.....	172
Figure 86. Commitment Statistic - Data Flow Diagram Level 0	172
Figure 87. Commitments Ageing - Data Flow Diagram Level 0.....	173
Figure 88. Invoice - Data Flow Diagram Level 0	173
Figure 89. MR Order Notification - Data Flow Diagram Level 0.....	174
Figure 90. Material Requisition Compliance to issue rig / jetty - Data Flow Diagram Level 0	174
Figure 91. Stock View - Data Flow Diagram Level 0.....	175

Figure 92. Inventory Profile - Data Flow Diagram Level 0	175
Figure 93. Material Requisition Statistic - Data Flow Diagram Level 0	176
Figure 94. Ageing - Data Flow Diagram Level 0	176
Figure 95. Material Requisition Compliance - Data Flow Diagram Level 0	176
Figure 96. Stock Management – Open End - Data Flow Diagram Level 0	177
Figure 97. Stock Management – Closed End - Data Flow Diagram Level 0	177
Figure 98. ADNOC ERP Log - Data Flow Diagram Level 0	178
Figure 99. Material Requisition - Data Flow Diagram Level 0	178
Figure 100. Rig Return - Data Flow Diagram Level 0	179
Figure 101. Stock Aging - Data Flow Diagram Level 0	179
Figure 102. Stock View - Data Flow Diagram Level 0	179
Figure 103. Transaction Report - Data Flow Diagram Level 0	180
Figure 104. ACCS Commitment Reports - Data Flow Diagram Level 0	180
Figure 105. Ageing Detail Report - Data Flow Diagram Level 0	181
Figure 106. Ageing Recap - Data Flow Diagram Level 0	182
Figure 107. Commitment Ageing Recap - Data Flow Diagram Level 0	182
Figure 108. Dispatch Reports - Data Flow Diagram Level 0	182
Figure 109. Material Requisition - Data Flow Diagram Level 0	183
Figure 110. Material Requisition Invoice - Data Flow Diagram Level 0	183
Figure 111. OCTG Commitment Reports - Data Flow Diagram Level 0	184
Figure 112. Return Creation - Data Flow Diagram Level 0	184
Figure 113. Inventory Report - Data Flow Diagram Level 0	184
Figure 114. Inventory Review - Data Flow Diagram Level 0	185
Figure 115. Stock Summary - Data Flow Diagram Level 0	185
Figure 116. Movement Detail Report - Data Flow Diagram Level 0	186
Figure 117. Movement Report - Data Flow Diagram Level 0	186
Figure 118. Ownership Transfer & Return Ledger Report - Data Flow Diagram Level 0	187
Figure 119. Damaged Inspection Result - Data Flow Diagram Level 0	187
Figure 120. Inspection Creation - Data Flow Diagram Level 0	188
Figure 121. Inactive User - Data Flow Diagram Level 0	188
Figure 122. List Super Admin - Data Flow Diagram Level 0	188
Figure 123. Login Recap - Data Flow Diagram Level 0	189
Figure 124. Batch Movement Report - Data Flow Diagram Level 0	189
Figure 125. Report Preview - Data Flow Diagram Level 0	190
Figure 126. Resources - Data Flow Diagram Level 0	190
Figure 127. Mill Audit Request - Data Flow Diagram Level 0	191
Figure 128. Mill Audit Request - Data Flow Diagram Level 1	192
Figure 129. Mill Listing - Data Flow Diagram Level 0	193
Figure 130. Mill NCR - Data Flow Diagram Level 0	194
Figure 131. Mill MoC - Data Flow Diagram Level 0	195
Figure 132. Mill MoC - Data Flow Diagram Level 1	195
Figure 133. User - Data Flow Diagram Level 0	196
Figure 134. User - Data Flow Diagram Level 1	197
Figure 135. Privilege - Data Flow Diagram Level 0	198
Figure 136. Tablet Access - Data Flow Diagram Level 0	198
Figure 137. Tablet Access - Data Flow Diagram Level 1	199

Figure 138. Customer > Customer Info - Data Flow Diagram Level 0.....	199
Figure 139. Customer > Customer Info - Data Flow Diagram Level 1	200
Figure 140. Customer > Lot Price - Data Flow Diagram Level 0	201
Figure 141. Customer > Fees - Data Flow Diagram Level 0.....	201
Figure 142. Customer > Customer Item No. - Data Flow Diagram Level 0	202
Figure 143. Customer > Collection Point - Data Flow Diagram Level 0.....	202
Figure 144. Procurement > Contractor - Data Flow Diagram Level 0	203
Figure 145. Procurement > Port - Data Flow Diagram Level 0	203
Figure 146. Inventory > Item - Data Flow Diagram Level 0.....	204
Figure 147. Inventory > OD - Data Flow Diagram Level 0.....	204
Figure 148. Inventory > Grade - Data Flow Diagram Level 0	205
Figure 149. Inventory > Connection - Data Flow Diagram Level 0.....	205
Figure 150. Inventory > Special Condition - Data Flow Diagram Level 0.....	206
Figure 151. Inventory > Yard - Data Flow Diagram Level 0	207
Figure 152. Inventory > Yard - Data Flow Diagram Level 1.....	207
Figure 153. Inventory > Package - Data Flow Diagram Level 0.....	208
Figure 154. Quality & HSE > Category Customization - Data Flow Diagram Level 0 ...	208
Figure 155. Support - Data Flow Diagram Level 0	209
Figure 156. Support - Data Flow Diagram Level 1.....	210

LIST OF TABLES

Table 1. Hosting Specification.....	33
Table 2. Server Configuration	34
Table 3. Database Configuration	34
Table 4. Authentication and Security	34
Table 5. Logging and Monitoring.....	35
Table 6. API Configuration	35
Table 7. NCR – List/Index Page	85
Table 8. NCR – Create & Edit 3 rd Party Instruction.....	85
Table 9. Detail NCR API Endpoints	85
Table 10. Inspection – Ageing Inspection API Endpoints.....	88
Table 11. Create and Edit Inspection API Endpoints	88
Table 12. Inspection – View Inspection Data API Endpoints.....	91
Table 13. Inspection – Close Short, Terminate, and Reopen Inspection API Endpoints	92
Table 14. Inspection – Scope of Work API Endpoints	92
Table 15. Inspection – Screening Process API Endpoints	93
Table 16. Quarantine – Create & Edit API Endpoints.....	98
Table 17. Quarantine – Detail & List API Endpoints	99
Table 18. Quarantine – Assign New Quarantine API Endpoints	99
Table 19. Quarantine – Assign Release & Release API Endpoints	100
Table 20. Quarantine – Close Short & Terminate API Endpoints.....	100
Table 21. Account Balance Header API Endpoints	101
Table 22. Account Balance Level Tabs & Items API Endpoints	101
Table 23. Account Balance Level Subrow Item & Detail.....	101
Table 24. Commitment – Public Commitment.....	102
Table 25. Commitment – Excel Generation API Endpoints.....	103
Table 26. Commitment – General Routes API Endpoints.....	103
Table 27. Commitment – Approval API Endpoints	104
Table 28. Commitment Management API Endpoints.....	105
Table 29. Commitment – Read Only API Endpoints.....	106
Table 30. Commitment – Shared API Endpoints	107
Table 31. Commitment – User Authentication API Endpoints	108
Table 32. Forecast API Endpoints	108
Table 33. Lot Info – List API Endpoints	109
Table 34. Lot Info – Create Lot Info.....	109
Table 35. Lot Info – Detail API Endpoints	109
Table 36. Lot Info – MDR API Endpoints	110
Table 37. Lot Info – Tally List API Endpoints	110
Table 38. Lot Info – Inspection API Endpoints.....	111
Table 39. Mill Set Up API Endpoints.....	111
Table 40. Stock on Hand API Endpoints	111
Table 41. Transfer – Authentication API Endpoints	112
Table 42. Transfer – Transfer Management API Endpoints	112
Table 43. Transfer – Document Generation API Endpoints	113
Table 44. Transfer – Delivery Note API Endpoints.....	114
Table 45. Transfer – Utility API Endpoints	114

Table 46. Transfer – ERP Integration API Endpoints	115
Table 47. Transfer – Tablet Operation API Endpoints	115
Table 48. Transfer – Internal Notes API Endpoints	116
Table 49. Transfer – Attachment API Endpoints	116
Table 50. Transfer – Others API Endpoints.....	117
Table 51. List/Index Page API Endpoints.....	117
Table 52. Create & Edit 3 rd Party Instruction API Endpoints	118
Table 53. Detail 3 rd Party Instruction.....	119
Table 54. Mill Audit Request – List/Index API Endpoints.....	120
Table 55. Create & Edit Mill Audit Request API Endpoints.....	121
Table 56. Mill Listing API Endpoints	123
Table 57. Mill Listing – Audit Related API Endpoints	123
Table 58. Mill Listing – MoC API Endpoints	124
Table 59. Mill NCR – List/Index Page API Endpoints	124
Table 60. Edit Mill NCR API Endpoints.....	124
Table 61. Detail Mill NCR API Endpoints	124
Table 62. Mill MoC – List/Index API Endpoints	125
Table 63. Create & Edit Mill MoC API Endpoints	125
Table 64. Detail Mill MoC API Endpoints	125
Table 65. Setup User API Endpoints.....	126
Table 66. Setup Privilege API Endpoints.....	127
Table 67. Setup Tablet – List/Index Page API Endpoints	127
Table 68. Create & Edit Table Setup	127
Table 69. Detail Tablet Setup	128
Table 70. Setup Contractor API Endpoints.....	128
Table 71. Setup Port API Endpoints	129
Table 72. Setup Item API Endpoints.....	130
Table 73. Setup OD API Endpoints.....	130
Table 74. Setup Grade API Endpoints	131
Table 75. Setup Connection API Endpoints.....	131
Table 76. Setup Special Condition API Endpoints	132
Table 77. Setup Yard API Endpoints	132
Table 78. Support – Jira API Endpoints.....	133
Table 79. Support API Endpoints	134
Table 80. NCR - Data Flow Diagram level 1 key components	139
Table 81. MoC - Data Flow Diagram level 1 key components	142
Table 82. Inspection > Inspection Report Data Flow Diagram level 1 key components	148
Table 83. Inspection > Quarantine Lot - Data Flow Diagram level 1 key components	150
Table 84. Material Requisition - Data Flow Diagram level 1 key components	155
Table 85. Return - Data Flow Diagram level 1 key components	157
Table 86. Ownership Transfer - Data Flow Diagram level 1 key components.....	159
Table 87. Lot Info - Data Flow Diagram level 1 key components	161
Table 88. Open Order - Data Flow Diagram level 1 key components	163
Table 89. Transfer > Port / Yard - Data Flow Diagram level 1 key components.....	167
Table 90. 3 rd Party Instruction - Data Flow Diagram level 1 key components	170
Table 91. Production - Data Flow Diagram level 1 key components	171

Table 92. 3 rd Mill Audit Request - Data Flow Diagram level 1 key components.....	193
Table 93. 3 rd Mill MoC - Data Flow Diagram level 1 key components	196
Table 94. User - Data Flow Diagram level 1 key components	197
Table 95. Tablet Access - Data Flow Diagram level 1 key components	199
Table 96. Customer > Customer Info - Data Flow Diagram level 1 key components.	200
Table 97. Inventory > Yard - Data Flow Diagram level 1 key components.....	207
Table 98. Support - Data Flow Diagram level 1 key components.....	210
Table 99. abbreviations table	211
Table 100. actionlist table	211
Table 101. assign_evaluators table	212
Table 102. auditschedule table	213
Table 103. ax_soh table.....	214
Table 104. axlog table.....	215
Table 105. azure_log table	216
Table 106. batch_trackability_itemcode_matrix table	216
Table 107. batch_trackability_matrix table	217
Table 108. calloff_counter table.....	217
Table 109. Client table.....	217
Table 110. commitment table.....	220
Table 111. config table.....	222
Table 112. configuration table	222
Table 113. connection table	223
Table 114. contract table.....	223
Table 115. contract_doc table	224
Table 116. contractor table	225
Table 117. counter table	226
Table 118. country table	227
Table 119. d365_inv_arch table	227
Table 120. d365_inv_log table	229
Table 121. d365_obs_arch table	229
Table 122. d365_obs_log table.....	232
Table 123. ddd_logger table.....	232
Table 124. delivery table	233
Table 125. department table	233
Table 126. deliveryterms table.....	234
Table 127. doc_format table.....	234
Table 128. docs table.....	234
Table 129. document table	236
Table 130. ehs table	238
Table 131. ehs_target table	239
Table 132. email_log table	240
Table 133. erp_findim table	241
Table 134. erp_log table	241
Table 135. fc_acc_weighted table	242
Table 136. fc_d365 table	242
Table 137. fc_d365_hist table	243
Table 138. fc_d365_recap table	243

Table 139. fees table.....	244
Table 140. forecast table	244
Table 141. forecast_acc table.....	245
Table 142. forecast_acc_hist table	245
Table 143. forecast_d365_arch table.....	246
Table 144. forecast_mb_hist table	247
Table 145. grade table	248
Table 146. helps table	248
Table 147. imei table.....	249
Table 148. index table	249
Table 149. indexval table	250
Table 150. inquiry table	251
Table 151. inspection_counter table	252
Table 152. inspection_image table	252
Table 153. inspection_temp table	253
Table 154. internal_note table	254
Table 155. inventory_report table.....	254
Table 156. invoice table.....	256
Table 157. invoice_mts table	257
Table 158. invoice_pdf table.....	258
Table 159. invreport table	258
Table 160. item table	260
Table 161. item_counter table	263
Table 162. item_type table	264
Table 163. item_type_attribute table	264
Table 164. itemcard table	265
Table 165. itemcard_balance table	267
Table 166. itemcard_rack table	270
Table 167. itemcpo table.....	273
Table 168. iteminspection table	275
Table 169. itemtally table	275
Table 170. Itemtrans table.....	277
Table 171. itemvims table	288
Table 172. ivms table	289
Table 173. jira_issue_tickets.....	291
Table 174. jira_log	291
Table 175. lessonlearned table	292
Table 176. li_currency table.....	293
Table 177. li_delivery table	293
Table 178. li_recipients table	293
Table 179. li_uom table	294
Table 180. loadout_tally table.....	294
Table 181. locode table	295
Table 182. log_api table.....	295
Table 183. logs table	296
Table 184. mdr table.....	297
Table 185. mdr_folder table	298

Table 186. merchandise_balance table	298
Table 187. merchandise_balance_ceiling table.....	299
Table 188. merchandise_balance_comm table.....	300
Table 189. merchandise_balance_comm_raw table	300
Table 190. merchandise_balance_log table.....	301
Table 191. merchandise_balance_raw table	301
Table 192. merchandise_balance_raw_hist table.....	302
Table 193. chandise_balance_report table	303
Table 194. merchandise_balance_sc table	305
Table 195. mill table	305
Table 196. miladvisorl table	307
Table 197. moc table.....	308
Table 198. moc_dropdown table	311
Table 199. notification table.....	311
Table 200. od table.....	312
Table 201. od table.....	312
Table 202. ordervims table.....	313
Table 203. pipesales_bu_setup table	315
Table 204. pipesales_global_setup table.....	316
Table 205. pipesales_term_condition table	318
Table 206. pl_reject_reason table.....	318
Table 207. po_mts table	319
Table 208. port table.....	319
Table 209. privcode table	320
Table 210. privilege table	320
Table 211. product_level table.....	321
Table 212. protectormat table	321
Table 213. quality table	321
Table 214. quarantine table.....	322
Table 215. range table	323
Table 216. recap_listing_ledger table	323
Table 217. recap_metadata table	325
Table 218. region table	325
Table 219. revision_document table	326
Table 220. role table.....	326
Table 221. salesdata table	327
Table 222. scope_work table	329
Table 223. scope_work_template table	329
Table 224. signoff table	330
Table 225. specialcond table	332
Table 226. spr table.....	332
Table 227. spr_causal_factor table	335
Table 228. spr_causal_factor_old table.....	336
Table 229. spr_hse_option table	336
Table 230. spr_hse_option table	337
Table 231. spr_reason table.....	337
Table 232. spr_reason_old table	337

Table 233. spr_user_role table	337
Table 234. sprv2 table.....	338
Table 235. sr_material table	340
Table 236. sr_od table.....	340
Table 237. sr_type table	341
Table 238. storage_files table.....	341
Table 239. tally_history table	342
Table 240. tenderdoc table.....	343
Table 241. todo table.....	343
Table 242. tpicomp table.....	344
Table 243. transaction_logs table.....	344
Table 244. transfer_activity table.....	345
Table 245. transfer_counter table	345
Table 246. transfer_receipt table	346
Table 247. turnrate_hist table	346
Table 248. turnrate_price table.....	347
Table 249. user_activity table.....	348
Table 250. user_config table.....	348
Table 251. user_log table	349
Table 252. userlog table.....	349
Table 253. users table.....	350
Table 254. vendor_order table.....	353
Table 255. vessel table	354
Table 256. vessel_tracking table.....	355
Table 257. vims_avgprice table.....	356
Table 258. vims_batchprice table	357
Table 259. vims_breakdown table	357
Table 260. vims_forecast table	358
Table 261. vims_forecast_hist table	360
Table 262. vims_pdf table	361
Table 263. vims_recomorder table	361
Table 264. well_design table	362
Table 265. yard table.....	362
Table 266. yard_balance_report table.....	364
Table 267. well_yard_opening_balance_report table	365
Table 268. yard_supervisor table	366

LIST OF ABBREVIATIONS

Term	Description
API	Application Programming Interface
AWS	Amazon Web Services
CI/CD	Continuous Integration/Continuous Deployment (or Delivery)
CPU	Central Processing Unit
DB	Database
DFD	Data Flow Diagram
DI	Dependency Integration
DIP	Dependency Inversion Principle
DNS	Domain Name System
ERP	Enterprise Resource Planning
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ID	Identification Number
IDE	Integrated Development Environment
IMEI	International Mobile Equipment Identity
ISO	International Organization of Standardization
ISP	Interface Segregation Principle
JWT	JSON Web Token
LSP	Liskov Substitution Principle
MDR	Manufacturing Data Record
MITME	Marubeni Itochu Tubular Middle East
NoSQL	Non-Relational Standard Query Language
OCP	Open/Closed Principle
OOP	Object Oriented Programming
PHP-CS	PHP Code Sniffer
PSR	PHP Standard Recommendation
SI	Shipment Instruction
SQL	Standard Query Language
SRP	Single Responsibility Principle

1. Introduction

The Technical Design Document serves as a comprehensive guide outlining the system architecture and implementation details. It provides a structured framework for developers, architects, and stakeholders to understand the technical aspects of a project.

This document includes server configuration, which details the specifications and setup of the server, including cloud, database, authentication, logging, and API configurations. It also defines coding standards, outlining best practices and conventions for writing clean, maintainable, and efficient code, such as naming conventions, indentation rules, error handling, and documentation standards.

Another key aspect is interaction protocols, which describe the communication methods between different system components. The component design diagram provides a visual representation of system modules and their interactions, illustrating how different components work together to achieve the desired functionality.

Additionally, the document includes a list of API endpoints, specifying available APIs and HTTP methods. The data flow diagram (DFD) illustrates how data moves within the system, from user inputs to processing and data storage, ensuring clarity in data processing and transformation. Lastly, the database collections section defines database collections and tables used for storing and retrieving data efficiently.

2. Specific Configuration

2.1 Server Configuration

Table 1. Hosting Specification

Environment	Hosting Provider	Region	Processor	Server Name	RAM	Disk Size
Production	AWS	ap-southeast-1	8 vCPU	mitme-server	32 GB	622 GB
Demo	AWS	me-central-1	2 vCPU	mitme-demo	8 GB	156 GB
Staging	AWS	ap-southeast-1	4 vCPU	staging-dev	16 GB	156 GB
Test	AWS	ap-southeast-1	8 vCPU	test-server	32 GB	622 GB
Devel	AWS	ap-southeast-1	4 vCPU	devel	8 GB	156 GB

Table 2. Server Configuration

Environment	Server Type	Backend Framework	Port Configuration	SSL Certificate	Environment Variable
Production, Demo, Staging, Test, Devel	Nginx 1.18.0 (Ubuntu)	Laravel 10.13.5	80 (HTTP) 443 (HTTPS)	Enable	APP_ENV: Production PORT: - DEBUG_MODE: False

2.2 Database Configuration

Table 3. Database Configuration

Environment	Database Type	Host	Port
Production	MongoDB 7.0.14	cluster0-nyyfc.mongodb.net	27017
Demo	MongoDB 7.0.14	localhost	27017
Staging	MongoDB 7.0.14	localhost	27017
Test	MongoDB 7.0.14	localhost	27017
Devel	MongoDB 7.0.14	localhost	27017

In production, Tubestream use Managed MongoDB Atlas, while for other environments, Self-Hosted MongoDB is used within the server itself.

2.3 Authentication and Security

Table 4. Authentication and Security

Authentication Method	Password Encryption	User Roles	Rate Limiting
JWT	Bcrypt	Yard, User Approval, User Admin, TPI Contractor, Superadmin, Quality Manager, Quality, Operation, MD, Inventory Manager, Inventory, Finance, Delivery Coordinator, Commercial Manager, Commercial, Admin Tech, Admin Inv	Default by Laravel

2.4 Logging and Monitoring

Table 5. Logging and Monitoring

Logging Framework	Log Level	Monitoring Tool	Error Tracking
Monolog by Laravel	Error	Tick stack	Sentry

2.5 API Configuration

Table 6. API Configuration

Base URL	Rate Limiting	Allowed Origin	API Key Management
https://mitme.tubestream.com/api	Default by Laravel	https://mitme.tubestream.com	Enforce via environment variables

3. Coding Standard

Coding standards are created to establish the best practices for writing clean, maintainable, and scalable code. It includes guidelines for naming conventions, file structure, documentation, and version control to maintain consistency across the development team.

Tubestream follows the PSR-12 coding standard to ensure a consistent and maintainable codebase. This standard improves code readability, simplifies debugging, and facilitates collaboration among developers. PSR-12 is an extension of PSR-2, modernizing PHP coding standards to improve readability, consistency, and maintainability.

3.1 Basic Coding Standard

- Use 4 spaces for indentation.
- Maximum line length of 120 characters.
- Code must use UTF-8 encoding.
- No closing PHP tag at the end of the file.

Example:

```
// Incorrect (Using tabs instead of spaces)
function exampleFunction() {
    return true;
}

// Correct (Using 4 spaces for indentation)
function exampleFunction() {
    return true;
}
```

Figure 1. Basic Coding Standard

3.2 Namespace and Use Declarations

- Declare namespaces at the top of the file.
- Use statements must be grouped and organized alphabetically.

Example:

```
<?php

namespace App\Controllers;

use App\Models\User;
use App\Services\AuthService;
```

Figure 2. Namespace and Use Declaration

3.3 Classes, Properties, and Methods

- Class names must be in PascalCase.
- Properties should be camelCase.
- Methods should be camelCase.

Example:

```
class UserController
{
    private string $userName;

    public function getUserName(): string
    {
        return $this->userName;
    }
}
```

Figure 3. Classes, Properties, and Methods

3.4 Control Structure

- Use one space after control structure keywords.
- Use curly braces always, even for single-line statements.

Example:

```
// Incorrect
if($user->isActive()) echo "User is active";

// Correct
if ($user->isActive()) {
    echo "User is active";
}
```

Figure 4. Control Structure

3.5 Coding Example

Advanced coding practices incorporating PSR-12:

```
class AuthService
{
    public function authenticate(string $username, string $password): bool
    {
        if ($this->verifyCredentials($username, $password)) {
            return true;
        }
        return false;
    }
}
```

Figure 5. Coding Example

3.6 Security Consideration

- Always validate and sanitize user inputs.
- Use prepared statements to prevent SQL injection.
- Implement proper error handling and logging.

Example:

```
$stmt = $pdo->prepare('SELECT * FROM users WHERE email = :email');
$stmt->execute(['email' => $email]);
```

Figure 6. Security Consideration

3.7 Code Readability and Maintainability

- Use meaningful variable and method names.
- Avoid deeply nested code blocks.
- Break long functions into smaller ones.

Example:

```
// Breaking a long function into smaller functions
class OrderProcessor
{
    public function processOrder(Order $order)
    {
        $this->validateOrder($order);
        $this->chargePayment($order);
        $this->shipOrder($order);
    }
}
```

Figure 7. Code Readability and Maintainability

3.8 Common Mistakes

- Incorrect indentation.
- Inconsistent naming conventions.
- Omitting type declarations.

Example:

```
// Incorrect
function getData($data) { return strtoupper($data); }

// Correct
function getData(string $data): string
{
    return strtoupper($data);
}
```

Figure 8. Common Mistakes

3.9 Migration from PSR-2 to PSR-12

- Convert tabs to spaces.
- Adjust line lengths.
- Refactor namespace declarations.

3.10 Tools for Enforcing PSR-12

To ensure compliance with PSR-12, developers can use the following tools:

- PHP_CodeSniffer

PHP_CodeSniffer (PHPCS) is a widely used tool that detects violations of coding standards, including PSR-12. It provides automated checks and can be integrated into CI/CD pipelines.

Example:

```
#Install PHP_CodeSniffer globally
composer global require "squizlabs/php_codesniffer"

#Check a PHP file for PSR-12 violations
phpcs --standard=PSR12 path/to/your/file.php
```

Figure 9. PHP_CODESNIFFER

- PHP CS Fixer

PHP CS Fixer is another tool that automatically formats PHP code to adhere to PSR-12 standards.

Example:

```
# Install PHP CS Fixer
composer global require friendsofphp/php-cs-fixer

# Automatically fix code to comply with PSR-12
php-cs-fixer fix path/to/your/project --rules=@PSR12
```

Figure 10. PHP-CS-FIXER

- IDE Integrations (PHPStorm, VS Code, etc.)

Many modern IDEs offer built-in support for PSR-12 compliance through extensions or configuration settings.

Example (VS Code - PHP Intelephense Extension):

- Install the PHP Intelephense extension.
- Open VS Code settings and search for "format".
- Enable Editor: Format on Save.
- Use phpcs or php-cs-fixer as the default formatter.

Example (PHPStorm Configuration):

- Go to Preferences > Editor > Code Style > PHP.
- Select Set from... > Predefined Style > PSR-12.
- Click Apply and OK.
- Use Ctrl + Alt + L to format code automatically.

By using these tools, developers can ensure that their PHP code remains clean, consistent, and compliant with PSR-12.

3.11 Best Practice

To ensure a scalable and maintainable codebase, the following best practices should be followed

- Use Dependency Injection

Dependency Injection (DI) helps reduce coupling between classes by injecting dependencies rather than creating them internally. This improves testability and maintainability.

Example:

```
class UserService { private DatabaseConnection $db;
public function __construct(DatabaseConnection $db)
{
    $this->db = $db;
}
public function getUser(int $id): User
{
    return $this->db->fetchUserById($id);
}
```

Figure 11. Dependency Injection

By injecting DatabaseConnection into UserService, we allow for greater flexibility and easier testing.

- Follow SOLID Principle

SOLID principles help create maintainable and scalable applications:

- Single Responsibility Principle (SRP) - A class should have only one reason to change.
- Open/Closed Principle (OCP) - Classes should be open for extension but closed for modification.
- Liskov Substitution Principle (LSP) - Subtypes must be substitutable for their base types.
- Interface Segregation Principle (ISP) - Interfaces should be specific to the implementing class.
- Dependency Inversion Principle (DIP) - High-level modules should not depend on low-level modules; both should depend on abstractions.

Example:

```
// Open/Closed Principle - Instead of modifying PaymentProcessor, we extend it
interface PaymentMethod
{
    public function processPayment(float $amount): bool;
}

class CreditCardPayment implements PaymentMethod
{
    public function processPayment(float $amount): bool
    {
        // Process credit card payment
        return true;
    }
}

class PayPalPayment implements PaymentMethod
{
    public function processPayment(float $amount): bool
    {
        // Process PayPal payment
        return true;
    }
}
```

Figure 12. SOLID Principle

Using interfaces and extending functionality instead of modifying core classes makes code more maintainable.

- Write Unit Test for Critical Code

Testing is crucial for ensuring reliability. PHPUnit is a widely used testing framework for PHP.

Example:

```
use PHPUnit\Framework\TestCase;

class MathTest extends TestCase
{
    public function testAddition()
    {
        $this→assertEquals(4, 2 + 2);
    }
}
```

Figure 13. Unit Test Code

Running unit tests ensure that changes do not break existing functionality. By following these best practices, developers can write cleaner, more maintainable, and scalable PHP applications.

3.12 Error Handling and Logging

- Differentiate Between Warnings and Critical Errors

Handling different severity levels properly ensures that minor issues don't crash the application while critical errors are immediately addressed.

```
try {

    if (!$user) {
        throw new Exception("User not found");
    }

} catch (Exception $e) {
    $log->error('Critical: ' . $e->getMessage());
}
```

Figure 14. Warning and Error

- Use Try-Catch Blocks to Handle Exceptions

Proper exception handling prevents runtime errors from crashing the application.

```
try {

    $db = new DatabaseConnection();
    $db->connect();

} catch (DatabaseConnectionException $e) {

    $log->error("Database connection failed: " . $e->getMessage());

    echo "Service temporarily unavailable.";

}
```

Figure 15. Handle Exception

- Error Logging Implementation

In enterprise applications, a structured error logging system should be implemented to categorize and properly handle different exceptions.

```

public static function formatLogWithResponse(Throwable $e, bool $saveToDatabase = true, bool
$withoutSnapshot = false): \Illuminate\Http\JsonResponse

{
    $msg = self::formatLog($e, $saveToDatabase, $withoutSnapshot);

    if ($e instanceof TubestreamDefinedWarningException) {

        return response()->json(['message' => $msg], 'confirmation_endpoint' => $e->
getConfirmationEndpoint(), 422);

    } elseif ($e instanceof TubestreamDefinedForbiddenException) {

        return response()->json(['error' => $msg], 403);

    } elseif ($e instanceof TubestreamDefinedNotFoundException) {

        return response()->json(['error' => $msg], 404);

    } elseif ($e instanceof TubestreamDefinedException) {

        return response()->json(['error' => $msg], 400);

    } else {

        return response()->json(['error' => $msg], 500);
    }
}
  
```

Figure 16. Error Logging Implementation

This structured approach ensures that different error types are handled appropriately and logged for future debugging. Implementing structured error handling and logging improves application stability and debugging efficiency.

3.13 Performance Optimization

- Use opcode caching.
- Optimize database queries.
- Profile code using tools like Xdebug and Blackfire.

3.14 Advanced OOP Concept

- Use traits for code reuse.
- Implement interfaces and abstract classes appropriately.

3.15 Design Pattern in PHP

- Utilize patterns such as Singleton, Factory, and Observer.
- Improve maintainability with well-structured patterns.

3.16 Dependency Injection and Service Containers

- Use DI to manage dependencies.
- Implement Laravel or Symfony service containers.

3.17 Testing and Debugging Best Practices

- Write unit tests using PHPUnit.
- Leverage debugging tools like Xdebug.

3.18 Continuous Integration and Deployment (CI/CD)

- Automate testing and deployment using GitHub Actions or GitLab CI/CD.
- Implement Docker for consistent environments.

3.19 Documentation and Code Comment

- Use PHPDoc to document functions and classes.
- Maintain well-structured README files.

Implementing PSR-12 enhances code readability, maintainability, and collaboration by ensuring a standardized coding style across PHP projects. Adhering to these guidelines help developers write structured and professional code, reducing inconsistencies and improving overall project quality.

4. Interaction Protocols

4.1 Client-Server Communication

- The frontend communicates with the backend through RESTful API calls over HTTPS.
- API requests include authentication tokens to ensure secure access.
- Responses follow a standardized JSON format for consistency.

4.2 API Request and Response Flow

- Clients send requests to the backend API using HTTP methods (GET, POST, PUT, DELETE).
- The server processes the request, validates data, and returns a response.
- Responses include HTTP status codes to indicate success or failure.

4.3 Authentication Flow

- Users log in using their credentials, and the system issues a JWT token.
- The token is included in API requests to authenticate users.
- Token expiration and refresh mechanisms are implemented for security.

5. Component Design

Component design section outlines the structure and interaction of various system components, ensuring a modular, scalable, and maintainable architecture. It defines the key components, their responsibilities, and how they communicate with each other through APIs, data flow, or messaging. This section provides a clear blueprint for developers, guiding implementation while maintaining consistency with system requirements and design principles.

5.1 Quality & HSE

5.1.1 NCR

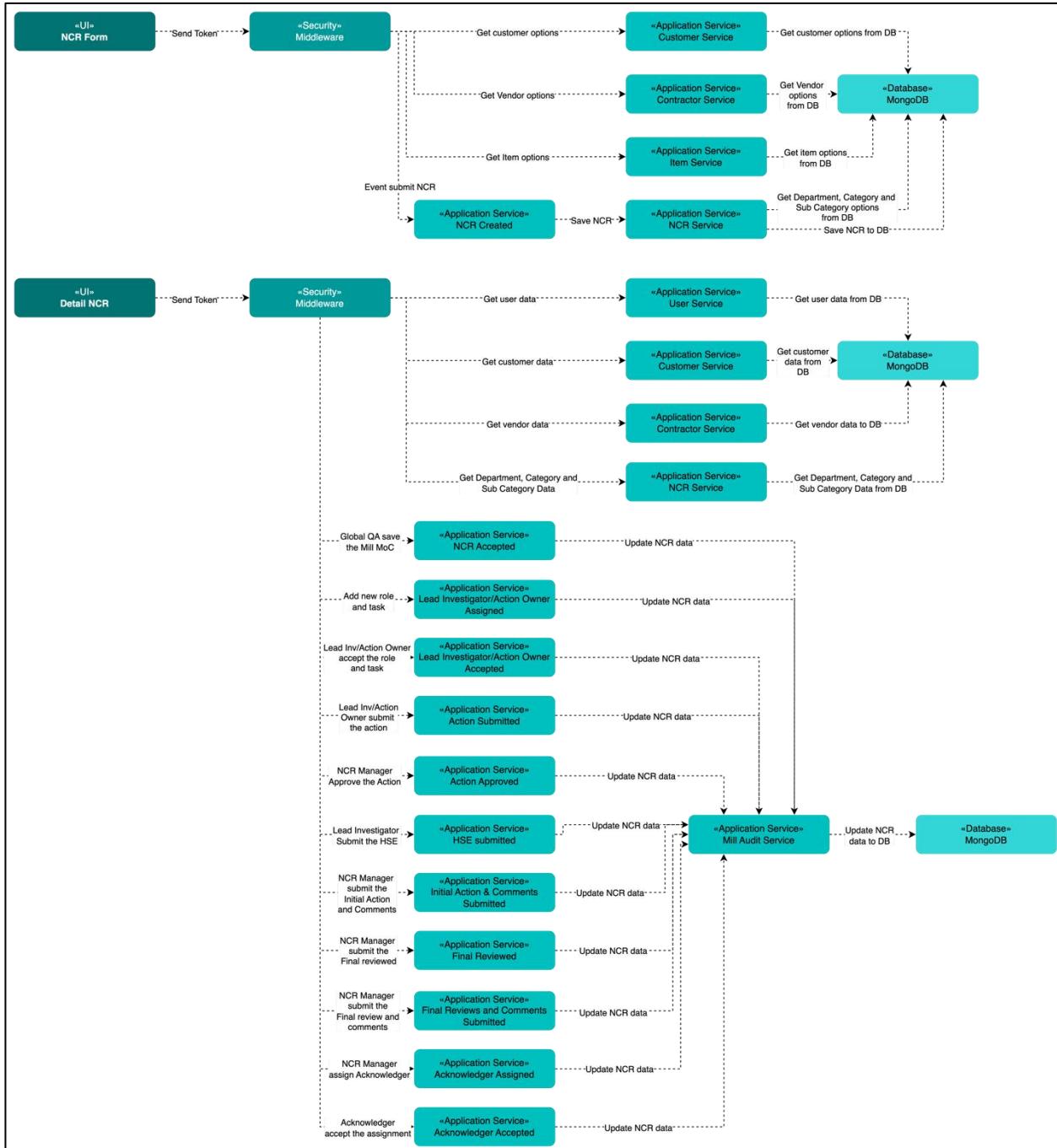


Figure 17. NCR Component Design

5.1.1.1 User Interface

5.1.1.1 NCR Form

This is the entry point for submitting a new NCR. Users can select from customer, vendor, and item options. Upon submission, it sends a token for authentication and triggers the submit NCR event.

5.1.1.2 Detail NCR

This UI is used to view and manage the full lifecycle of a submitted NCR. It supports actions like assigning owners, submitting actions, and reviewing/commenting. Each action passes through security and calls specific application services to update the NCR record.

5.1.1.2 Security

Middleware validates the authentication token sent from both the NCR Form and Detail NCR UIs. Only authenticated and authorized users can proceed to perform or view NCR actions.

5.1.1.3 Application Services

5.1.1.3.1 Initial Data Retrieval

- **Customer Service:** Fetches customer options from the database.
- **Contractor Service:** Retrieves vendor data.
- **Item Service:** Provides item options needed for the NCR.
- **NCR Service:** Retrieves department, category, and subcategory options for classifying the NCR.
- **User Service:** Loads user details tied to submitted or reviewed NCRs.

5.1.1.3.2 NCR Created

Handles the initial NCR creation and saves it to MongoDB. This includes selecting customers, vendors, items, and categories.

5.1.1.3.3 NCR Lifecycle Events

Each of the following services handles a specific stage in the NCR flow. All of them update NCR data in MongoDB through the NCR Service:

- **NCR Accepted:** Marks the NCR as accepted.
- **Lead Investigation/Action Owner Assigned:** Assigns users to handle further investigation and action.
- **Lead Investigation/Action Owner Accepted:** Confirms the assignment.
- **Action Submitted:** Submits a proposed corrective action.
- **Action Approved:** NCR Manager approves the submitted action.
- **HSE Submitted:** Lead Investigator submits HSE assessment.
- **Initial Action & Comments Submitted:** NCR Manager submits initial plan.
- **Final Reviewed:** Manager reviews the final plan.

- **Final Reviews and Comments Submitted:** Additional comments are logged.
- **Acknowledger Assigned:** NCR Manager assigns a user to acknowledge.
- **Acknowledger Accepted:** Assigned person accepts the acknowledgment task.

5.1.1.3.4 Mill Audit Service

This service is called when the HSE (Health, Safety, Environment) assessment is submitted. It updates audit results related to the NCR in the database.

5.1.1.4 Database

MongoDB is the central repository for:

- NCR records
- Dropdown/master data like vendors, customers, and categories
- HSE audit results

All create, update, and fetch operations on NCRs are handled here via the relevant services.

5.1.2 Inspection Report

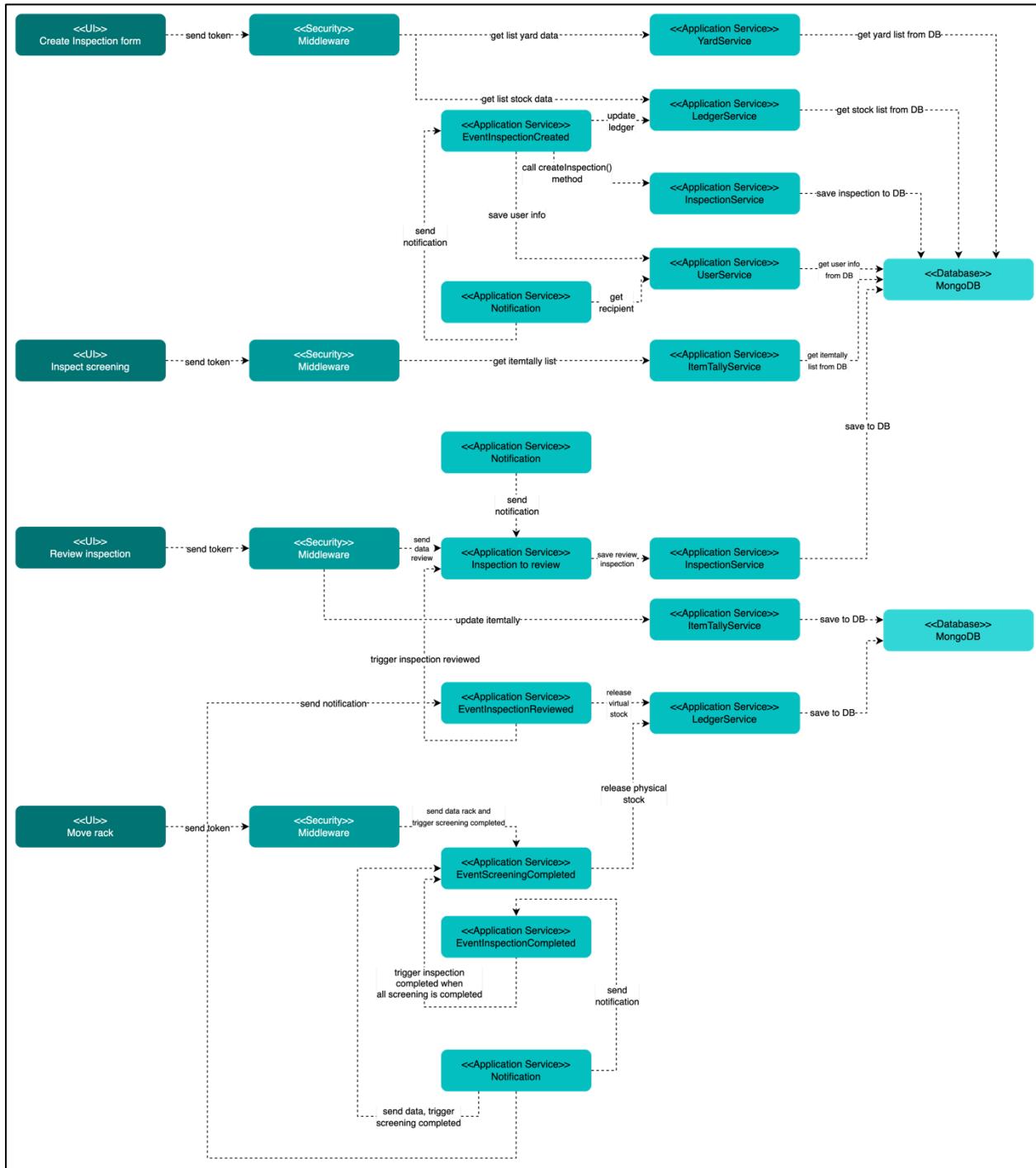


Figure 18. Inspection Report Component Design

5.1.2.1 User Interface

UI Component provides the user interface for creating, reviewing, and managing inspections. It allows users to input data, view inspection details, and track status updates.

5.1.2.1.1 Create Inspection

The Form Create Inspection UI is the entry point for users to initiate an inspection. Users input inspection details, such as location, stock information, and other relevant data. Once submitted, the request is validated and forwarded to the middleware for authentication. The system fetches the necessary yard and stock data before creating the inspection.

5.1.2.1.2 Inspect Screening

The inspect screening displays the list of items awaiting inspection. Users can review stock data and verify the inspection details. The UI retrieves item tally lists and other required information from the system. After verification, the inspection details are processed, and notifications are sent.

5.1.2.1.3 Review Inspection

Review inspection allows inspectors or authorized personnel to review inspection results. Users can add comments, confirm, or request changes to the inspection records. Sends reviewed data back to the InspectionService to update the inspection status in the database. The system triggers item tally updates and notification alerts.

5.1.2.1.4 Move Rack

Move rack UI handles the movement of inspected stock to the correct storage location. It displays options for updating virtual and physical stock locations. After rack movement is completed, it will trigger screening completion and inspection completion events once all steps are validated. The ledger and releases stock will accordingly be updated.

5.1.2.2 Security

The Middleware handles authentication and authorization by validating the security token. It ensures only authorized users can access the inspection process.

5.1.2.3 Application Service

Application Service Component handles business logic and coordinates interactions between UI, Web Services, and the database. It ensures data validation, processes inspection workflows, and triggers necessary updates across the system.

5.1.2.3.1 EventInspectionCreated

Processes the incoming inspection request by retrieving data from YardService and LedgerService, then calls the InspectionService to create a new inspection entry and sends notifications about the new inspection.

5.1.2.3.2 Notification

Sends alerts regarding the inspection progress.

5.1.2.3.3 EventInspectionReviewed

This service is triggered when an inspection review is completed, updating the inspection status and triggering item tally updates in the system. It interacts with InspectionService and ItemTallyService to store updated data in the database and sends notifications to relevant users about the inspection review results.

5.1.2.3.4 EventScreeningCompleted

This service processes the screening results of the inspection, ensuring all necessary verification steps are completed before marking an inspection as successful. Once screening is done, it triggers the EventInspectionCompleted service and interacts with LedgerService to update stock movements and record virtual stock adjustments.

5.1.2.3.5 EventInspectionCompleted

This is the final step in the inspection process, confirming that all required inspections and screenings have been successfully completed. It updates the database with the final inspection results, releases physical stock, and ensures the ledger is updated accordingly. Additionally, it sends notifications to users, informing them that the inspection process has concluded.

5.1.2.3.6 YardService

Manage yard inventory by retrieving stock, storage locations, and item statuses. It ensures inspections have accurate, real-time data by fetching details from the database.

5.1.2.3.7 LedgerService

Manages stock movements, updates inventory records, and ensures accurate tracking of virtual and physical stock.

5.1.2.3.8 InspectionService

Handles inspection data, stores results, and tracks the inspection process from creation to completion.

5.1.2.3.9 UserService

Manages user details, retrieves recipient information, and ensures correct notification delivery.

5.1.2.3.10 ItemTallyService

Updates item tally records, processes inspection-related item counts, and ensures accurate stock adjustments.

5.1.2.4 Database

Database component stores inspection details, user information, and stock data updates, where InspectionService saves inspection records into MongoDB, and LedgerService updates stock-related data based on inspections.

5.1.3 Quarantine Lot

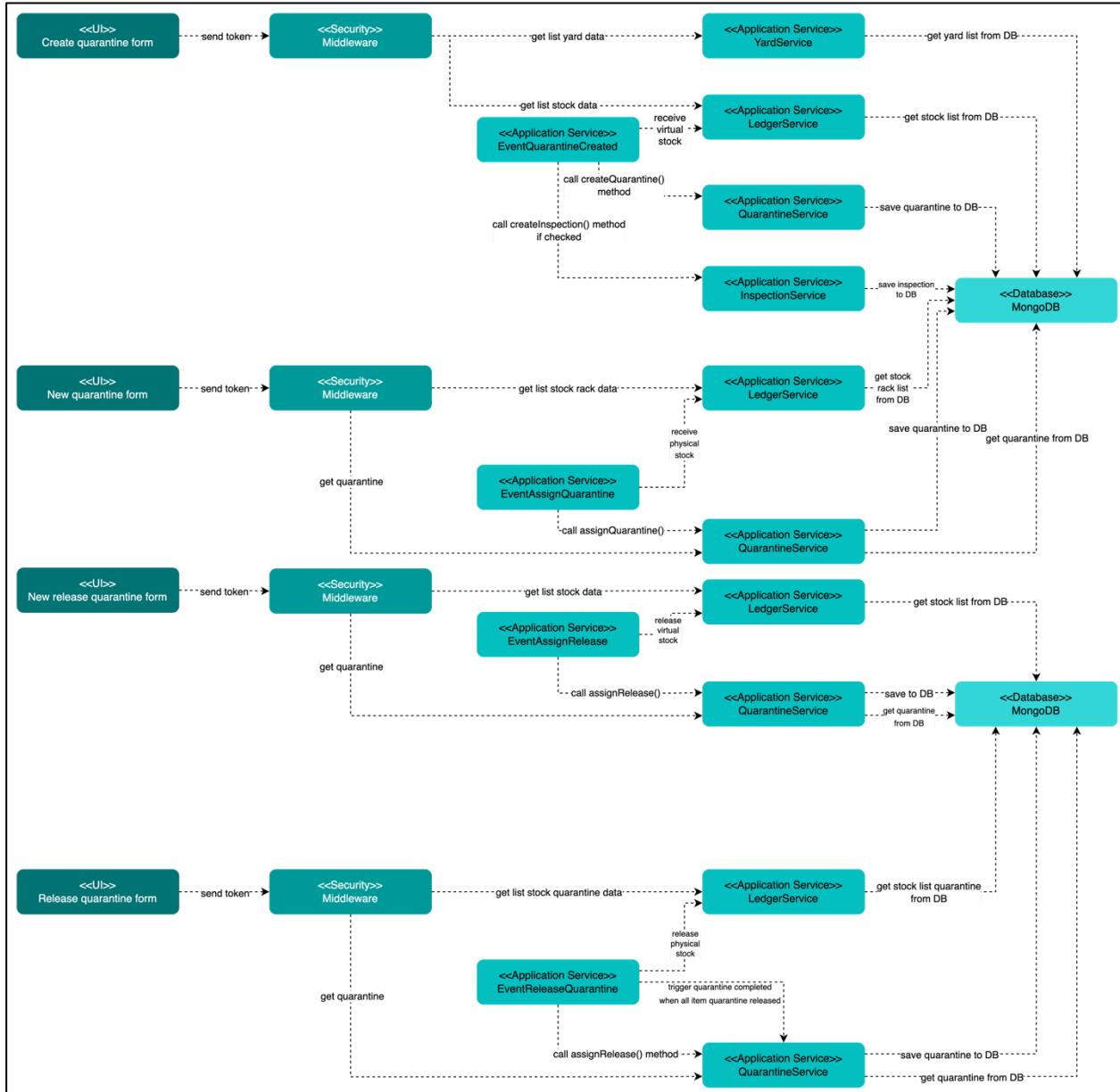


Figure 19. Quarantine Lot Component Design

5.1.3.1 User Interface

5.1.3.1.1 Create Quarantine Form

This form is used by users to initiate the creation of a new quarantine. It sends an authentication token to the middleware, which verifies the request before passing it along. The user's intent here is to mark certain stocks as quarantined, potentially triggering inspection requirements and other stock validations.

5.1.3.1.2 New Quarantine Form

The New Quarantine Form allows users to assign newly created quarantines to specific items or stock. After the user submits the form, it interacts with the backend to retrieve

current stock rack data and update the quarantine assignment accordingly. It serves as the second step in the quarantine process after creation.

5.1.3.1.3 New Release Quarantine Form

This form initiates a request to prepare a quarantined stock for release. The form collects necessary details and forwards them with a valid token. It does not directly release the stock but rather begins the validation and assignment of the release process.

5.1.3.1.4 Release Quarantine Form

The Release Quarantine Form is used to execute the final release of stock from quarantine. It confirms previous assignment decisions and communicates with the backend to complete the un-quarantining of the stock items.

5.1.3.2 Security

Middleware acts as a centralized authentication and authorization layer. It checks the token submitted by each UI form to ensure the request is valid and made by an authenticated user. Only upon passing this check does the middleware allow the request to proceed with the application services.

5.1.3.3 Application Services

5.1.3.3.1 EventQuarantineCreated

This service handles the full process of creating a quarantine. It retrieves yard data from YardService, fetches stock data from LedgerService and processes any applicable virtual stock. After validating the data, it calls the createQuarantine() method in QuarantineService. If inspection is required (as determined by the form), it also triggers the createInspection() method in InspectionService. Once complete, it saves the results to MongoDB.

5.1.3.3.2 EventAssignQuarantine

This service is responsible for assigning the newly created quarantine to specific stock or racks. It retrieves the relevant stock rack data via LedgerService, verifies eligibility, and then calls the assignQuarantine() method in QuarantineService. This ensures the quarantined items are tracked and stored appropriately in the system.

5.1.3.3.3 EventAssignRelease

The EventAssignRelease service manages the preliminary steps of releasing quarantined stock. It first retrieves stock quarantine data, including virtual stock details, and verifies the conditions for release. It then calls the assignRelease() method in QuarantineService to prepare the quarantine for final release.

5.1.3.3.4 EventReleaseQuarantine

This service completes the final step of the release process. After fetching all required data and confirming the stock is ready, it executes the release() method in QuarantineService, updating the quarantine status and releasing the items from the quarantine list. This action is also saved to the MongoDB database for consistency and traceability.

5.1.3.3.5 YardService

YardService is a utility application service that provides data about the yard (storage location). It is used by EventQuarantineCreated to get the list of available yard areas where stock might be quarantined.

5.1.3.3.6 LedgerService

LedgerService is used to retrieve various types of stock data, such as stock racks or quarantine-specific stock lists. It supports the quarantine and release flows by providing current inventory and location details.

5.1.3.3.7 QuarantineService

This is the central business logic service for saving, updating, and managing quarantine records. It is called by all major event services (EventQuarantineCreated, EventAssignQuarantine, EventAssignRelease, EventReleaseQuarantine) to persist changes into the database.

5.1.3.3.8 InspectionService

InspectionService is triggered during quarantine creation if the user has flagged the need for inspection. It creates inspection records and stores them in the database. It works in tandem with QuarantineService to ensure that both quarantine and inspection processes are handled together when applicable.

5.1.3.4 Database

MongoDB is the primary data store used in this architecture. It persists all quarantine records, inspection logs, and stock data changes. It is accessed by QuarantineService and InspectionService for writing and retrieving data throughout all stages of the quarantine process.

5.2 Commercial

5.2.1 Account Balance

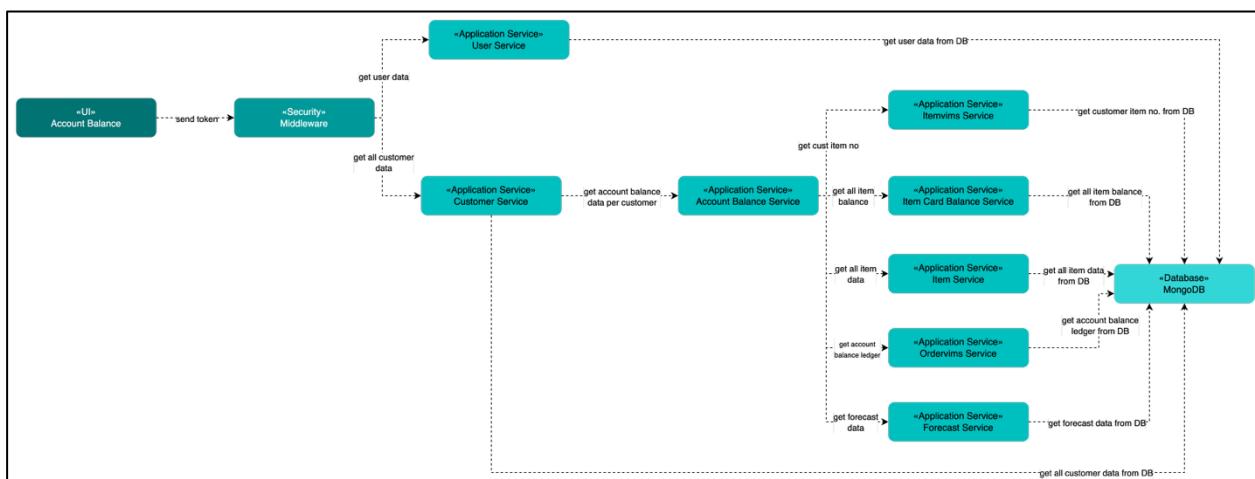


Figure 20. Account Balance Component Design

5.2.1.1 User Interface

When a user opens this page, the UI sends a token (usually a JWT) to the backend. This token represents the user's identity and is used for authentication. No business logic happens here; it simply initiates the request and presents the result to the user.

5.2.1.2 Security

The middleware acts as a security layer, handling both authentication and authorization. After receiving the token from the UI, it verifies the token's validity by calling the User Service, which retrieves user data from the database. Once the user is verified, the middleware then requests all related customer data through Customer Service, allowing the system to proceed with fetching account balances for those customers.

5.2.1.3 Application Service

5.2.1.3.1 User Service

Retrieves user data based on the token provided. This is used to verify that the user is authenticated and authorized to access further data.

5.2.1.3.2 Customer Service

Returns all customer records related to the authenticated user. This list of customers will later be used to fetch account balance data individually.

5.2.1.3.3 Account Balance Service

This is the core service that gathers and processes the account balance data for each customer. It integrates several other services to collect the required information.

5.2.1.3.4 Itemvims Service

Fetches customer-specific item numbers from the database. These item numbers are used to look up balances and other related data.

5.2.1.3.5 Item Card Balance Service

Retrieves the balance details of each item—such as available quantities—from the database.

5.2.1.3.6 Item Service

Returns full metadata about each item (e.g., name, description, unit). This is used to enrich the account balance details.

5.2.1.3.7 Ordervims Service

Provides account balance ledger entries. These are the transactional records (debits, credits, etc.) related to the account, used to calculate balances.

5.2.1.3.8 Forecast Service

Supplies forecast data, giving insights into future trends or projections of the account balance. This helps with planning or reporting.

5.2.1.4 Database

MongoDB serves as the central data store for the entire application. It holds all persistent data, including user records, customer profiles, item data, transaction ledgers, and forecast models. All the application services communicate directly with

MongoDB to fetch or process the data they are responsible for. This structure allows for flexible, scalable, and efficient data handling.

5.2.2 Commitment

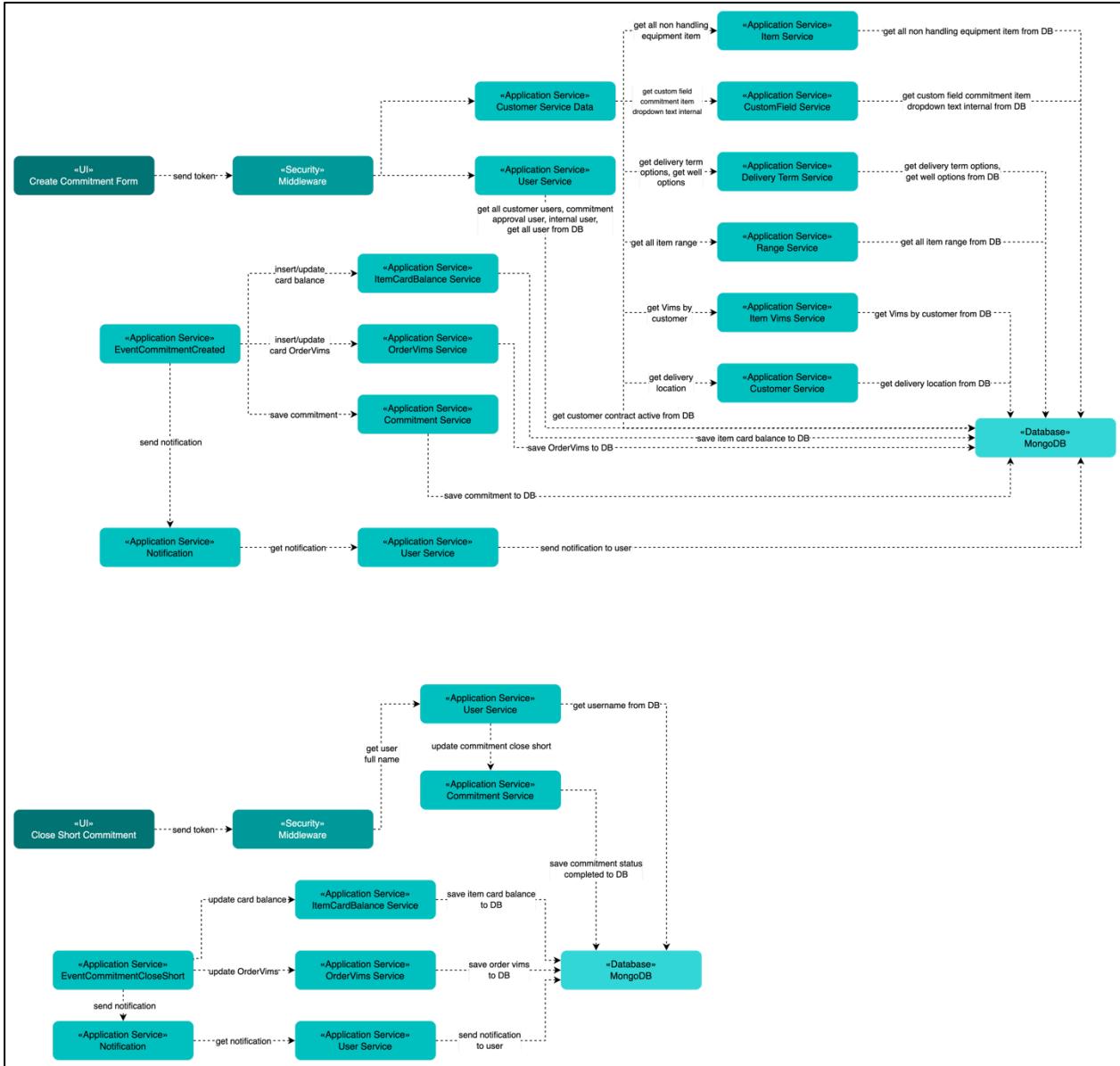


Figure 21. Commitment Component Design

5.2.2.1 User Interface

5.2.2.1.1 Create Commitment Form

This is the user-facing interface where users can input and submit new commitment data. It sends a token for authentication and initiates the process of creating a new commitment.

5.2.2.1.2 Close Short Commitment

This UI component is used to update or close a commitment earlier than its full term (e.g., due to cancellation or completion). It also sends a token to initiate the secure process.

5.2.2.2 Security

The middleware validates the incoming token from both UI components to authenticate the user. It ensures only authorized users can proceed. After verification, it calls the User Service to fetch user data, such as full name or username, used in both commit creation and closure.

5.2.2.3 Application Service

5.2.2.3.1 Create Commitment

5.2.2.3.1.1 User Service

Retrieves all user data from the database including internal users, customer approvers, etc. It's also used later for notification purposes and fetching user info.

5.2.2.3.1.2 Customer Service Data

Provides customer-related information like customer contact, contract details, and active statuses.

5.2.2.3.1.3 CustomerField Service

Supplies dropdown options such as commitment field input for custom UI fields.

5.2.2.3.1.4 Delivery Term Service

Fetches delivery term options, including general terms and special ones like VMI (Vendor Managed Inventory).

5.2.2.3.1.5 Item Service

Retrieves a list of items that are not handling equipment, i.e., eligible for commitment.

5.2.2.3.1.6 Range Service

Gets item range data (e.g., valid quantity or period ranges) for validation or selection.

5.2.2.3.1.7 Item VMI Service

Retrieves and validates VMI (Vendor Managed Inventory) by customer.

5.2.2.3.1.8 ItemCardBalance Service

Responsible for inserting or updating the item card balance related to the commitment.

5.2.2.3.1.9 OrderVMI Service

Manages the VMI orders. Inserts or updates order data linked to the commitment.

5.2.2.3.1.10 Commitment Service

Main service for saving the commitment itself into the system. It interacts with multiple services to ensure everything is correctly stored—commitment data, item card balance, order VMI, and customer contract data.

5.2.2.3.1.11 Notification Service

Sends a notification (usually after saving) to users involved in the commitment (e.g., confirmation or status update).

5.2.2.3.2 Close Short Commitment

5.2.2.3.2.1 User Service

Retrieves user information like full name or username for logging or tracking who closed the commitment.

5.2.2.3.2.2 Commitment Service

Updates the status of the commitment to completed short and saves this update to the database.

5.2.2.3.2.3 ItemCardBalance Service

Updates the item card balance, reflecting the short closing of the commitment.

5.2.2.3.2.4 OrderVMI Service

Updates any related VMI orders to reflect the closure.

5.2.2.3.2.5 Notification Service

Triggers and sends out notification to inform relevant users about the closure.

5.2.2.4 Database

MongoDB is the central database where all data is stored or retrieved, including:

- Customer and user information
- Item data and ranges
- Commitment records
- VMI orders and balances
- Notifications

All application services interact directly with MongoDB to perform reads and writes based on their responsibilities.

5.2.3 Forecast

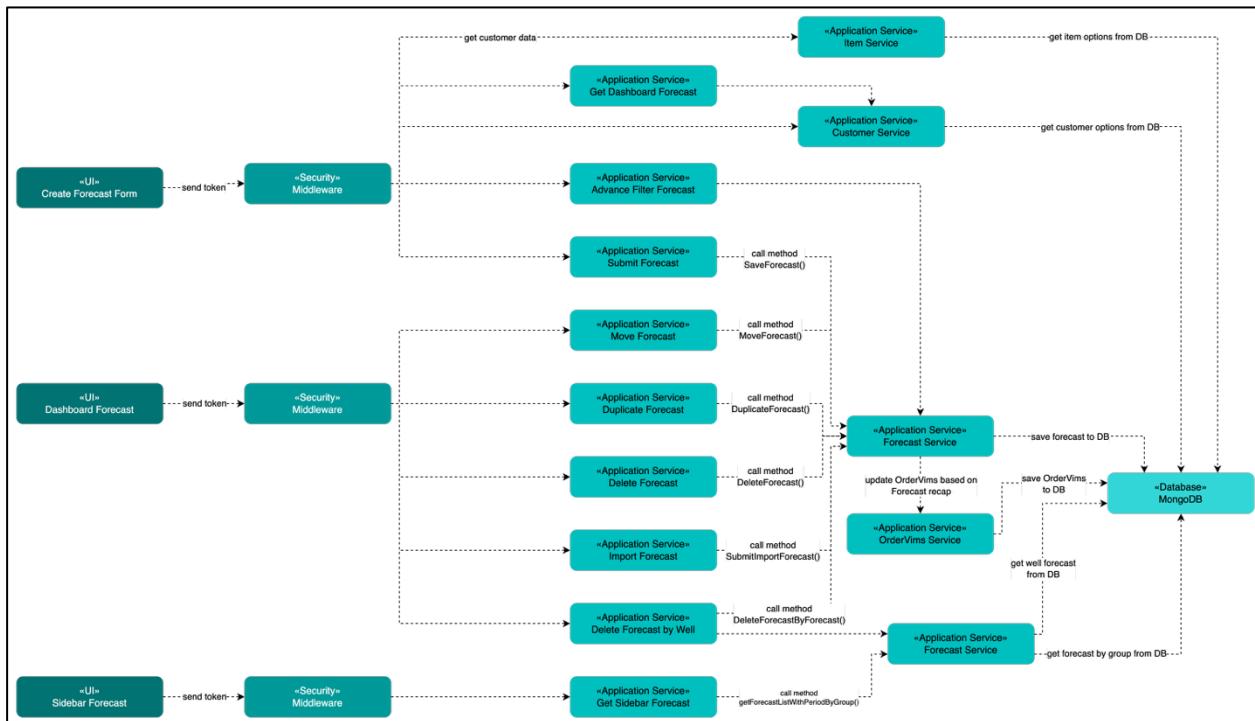


Figure 22. Forecast Component Design

5.2.3.1 User Interface

5.2.3.1.1 Create Forecast Form

This is the user interface for inputting new forecast data. It sends a token for user authentication and triggers the loading of item/customer options. After completing the form, it calls backend services to save the forecast.

5.2.3.1.2 Dashboard Forecast

This UI displays all forecast records in a centralized view. Users can perform operations like filtering, moving, duplicating, deleting, or importing forecast data. It sends a token to access protected data.

5.2.3.1.3 Sidebar Forecast

This UI presents a summary of a grouped view of forecasts, typically for quick access, filtering, or navigation. It also sends a token and interacts with the backend to get grouped data, such as forecast per period or group.

5.2.3.2 Security

Each UI component communicates with the Middleware, which:

- Verifies the user token
- Validates access rights
- Passes control to the relevant Application Services

This ensures that only authenticated and authorized users can perform forecast operations.

5.2.3.3 Application Services

5.2.3.3.1 Form Data and Filtering

5.2.3.3.1.1 Customer Service

Retrieves customer data used in dropdowns or filters.

5.2.3.3.1.2 Item Service

Retrieves item options needed to define forecast items.

5.2.3.3.1.3 Get Dashboard Forecast

Loads forecast data to be shown in the dashboard.

5.2.3.3.1.4 Advance Filter Forecast

Applies custom filters on forecast data (e.g., customer, item, well, or date range).

5.2.3.3.2 Forecast Operation

5.2.3.3.2.1 Submit Forecast → SaveForecast()

Saves new forecast entries to the database.

5.2.3.3.2.2 Move Forecast → MoveForecast()

Moves a forecast from one period or group to another.

5.2.3.3.2.3 Duplicate Forecast → DuplicateForecast()

Copies an existing forecast record for reuse.

5.2.3.3.2.4 Delete Forecast → DeleteForecast()

Removes a specific forecast from the database.

5.2.3.3.2.5 Import Forecast → SubmitImportForecast()

Imports multiple forecasts from an external source.

5.2.3.3.2.6 Delete Forecast by Well → DeleteForecastByForecast()

Deletes forecasts based on well ID or group.

5.2.3.3.2.7 Get Sidebar Forecast → getForecastListWithPeriodByGroup()

Retrieves grouped forecasts by period to be shown in the sidebar interface.

5.2.3.3.3 Supporting Logic

5.2.3.3.3.1 OrderVims Service

Updates OrderVims data based on forecast recaps (e.g., quantity forecasts turn into order plans) and saves or updates forecast-based OrderVims records in MongoDB.

5.2.3.4 Database

MongoDB is the single source of truth for all forecasting data. It stores:

- Forecast records
- Customer and item master data
- Grouped forecast views
- Imported forecast batches

- Updated OrderVims entries

MongoDB is accessed by:

- Forecast Service (read/write all forecast-related records)
- Customer & Item Service (read dropdown/filter data)
- OrderVims Service (read/write forecast-linked VMI data)

5.2.4 Ownership Transfer

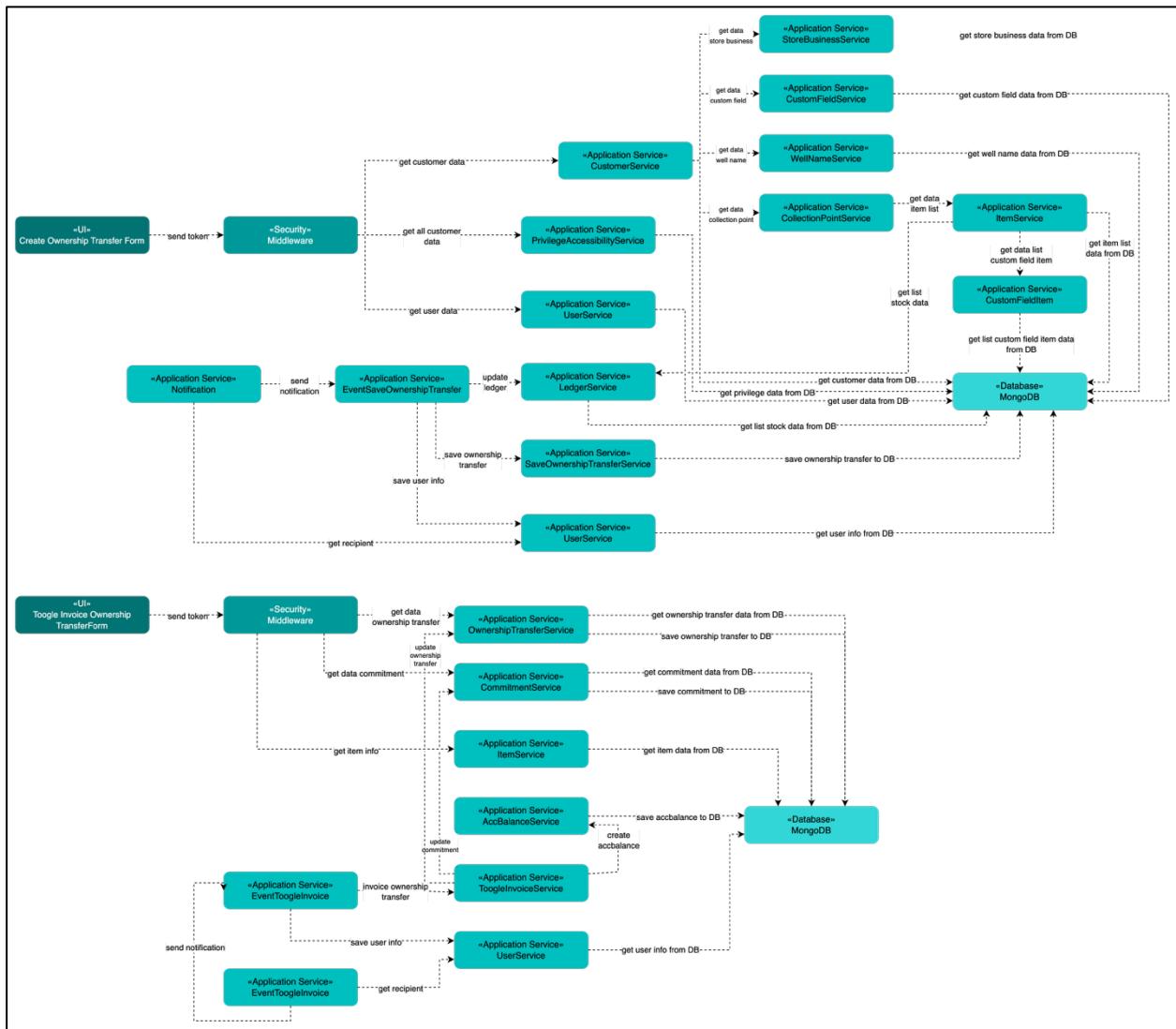


Figure 23. Ownership Transfer Component Design

5.2.4.1 User Interface

5.2.4.1.1 Create Ownership Transfer Form

This is the user interface where users input and submit data to initiate an ownership transfer. The form sends a token for user authentication and triggers the data gathering and submission process.

5.2.4.1.2 Toggle Invoice Ownership Transfer Form

This UI is used to toggle or update the ownership transfer status for invoices. It also sends a token to the backend to initiate secure data processing.

5.2.4.2 Security

In both flows, the middleware acts as a gatekeeper. It authenticates the token received from the UI to verify the user's identity. Once validated, it calls the User Service to fetch user information and allows the request to proceed to application services.

5.2.4.3 Application Services

5.2.4.3.1 Create Ownership Transfer

5.2.4.3.1.1 Customer Service

Retrieves customer data needed for the form or processing the ownership change.

5.2.4.3.1.2 PrivilegeAccessibility Service

Retrieves user privilege data to determine whether the user has the rights to perform ownership transfer.

5.2.4.3.1.3 User Service

Provides user information (ID, name, role) for ownership logging, tracking, and permissions.

5.2.4.3.1.4 StoreBusiness Service

Supplies store or business-related data from the database for contextual validation.

5.2.4.3.1.5 CustomField Service

Provides dynamic custom field configurations for the form (e.g., dropdowns).

5.2.4.3.1.6 WellName Service

Retrieves available well names associated with the ownership context.

5.2.4.3.1.7 CollectionPlan Service

Gets data used in relation to collection planning, likely involved with ownership responsibilities.

5.2.4.3.1.8 Item Service

Fetches item data, including stock lists, associated with the transfer. It also handles item selection in the form.

5.2.4.3.1.9 CustomFieldItem Service

Supplies item-specific custom fields (e.g., classification, tags) used for filtering or validation.

5.2.4.3.1.10 Ledger Service

Updates the ledger to reflect the ownership change, ensuring historical and financial accuracy.

5.2.4.3.1.11 SaveOwnershipTransfer Service

The core service that saves the actual ownership transfer record to the database.

5.2.4.3.1.12 EventSaveOwnershipTransfer

Sends a signal or record that an ownership transfer event occurred, possibly for audit trail or logging.

5.2.4.3.1.13 Notification Service

Sends a notification to users involved in the ownership transfer (e.g., sender/receiver).

5.2.4.3.2 Toggle Invoice Ownership Transfer Form**5.2.4.3.2.1 OwnershipTransfer Service**

Loads the existing ownership transfer data from the database.

5.2.4.3.2.2 Commitment Service

Retrieves and saves commitment data linked to the transferred item(s).

5.2.4.3.2.3 Item Service

Retrieves the item data involved in the invoice ownership transfer process.

5.2.4.3.2.4 AccBalance Service

Handles creation and saving of new account balances related to the invoice (e.g., moving ownership value from one party to another).

5.2.4.3.2.5 CreateAccbalance Service

Creates new account balance entries based on the ownership transfer.

5.2.4.3.2.6 ToggleInvoice Service

The main logic for updating or toggling the invoice ownership status.

5.2.4.3.2.7 EventToggleInvoice

Logs or notifies that an invoice ownership transfer was toggled.

5.2.4.3.2.8 User Service

Retrieves user info to determine the actor and recipient in the transfer.

5.2.4.4 Database

MongoDB is the central database that stores all relevant data such as customer and user information, business/store data, custom field definitions, item data and item-specific fields, ownership transfer records, ledger entries, account balances, notifications. MongoDB serves as the central data storage that supports all application services by providing persistent and structured data for the Ownership Transfer and Invoice Toggle Transfer processes.

5.3 Procurement

5.3.1 Lot Info

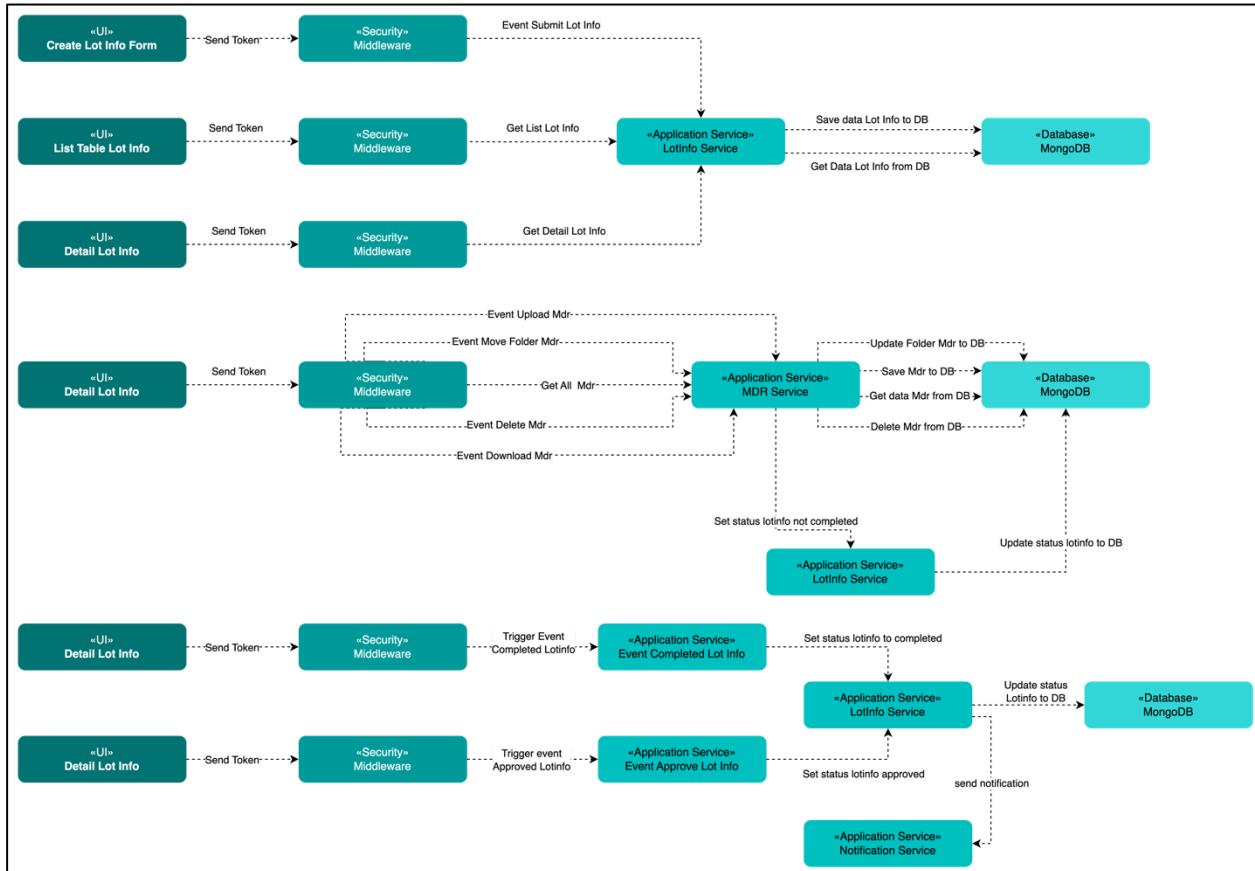


Figure 24. Lot Info Component Design

5.3.1.1 User Interface

5.3.1.1.1 Create Lot Info Form

This interface allows users to create and submit new Lot Info entries. Upon submission, it sends a token for authentication and triggers the Submit Lot Info event.

5.3.1.1.2 List Table Lot Info

Displays a table listing all Lot Info entries. It sends a token and requests a list of entries from the backend.

5.3.1.1.3 Detail Lot Info

This component appears multiple times in the flow — for displaying, uploading MDRs, updating statuses, and triggering events. It authenticates the user with a token before allowing any action.

5.3.1.2 Security

Every UI interaction sends a token to the middleware. The middleware:

- Validates the user's session
- Authorizes access to backend services

- Passes the request to the appropriate application service (e.g., LotInfo, MDR, Event, or Notification)

5.3.1.3 Application Service

5.3.1.3.1 LotInfo Service

This is the main service handling Lot Info records:

- Save data to MongoDB on lot creation
- Retrieve list/detail of Lot Info from MongoDB
- Update status of Lot Info (e.g., to completed, not completed, approved)

5.3.1.3.2 MDR Service

Handles MDR (Most Documented Records) file operations related to Lot Info:

- Upload, delete, download, or move MDR files
- Update folders in DB
- Save or retrieve MDRs from MongoDB

5.3.1.3.3 Event Service

- Event Completed Lot Info: Triggers completion of a Lot Info. Updates status to completed and saves to MongoDB.
- Event Approve Lot Info: Triggers approval of a Lot Info. Sets the status to approved, then passes it to LotInfo Service and Notification Service.

5.3.1.3.4 Notification Service

Sends notification once a Lot Info is approved (e.g., to notify relevant users or systems)

5.3.1.4 Database

MongoDB is the central data store for:

- Lot Info: Creation, updates, and status transitions
- MDR documents: Storage and retrieval of uploaded files and metadata
- Event logs: Related to approval and completion
- All services (LotInfo Service, MDR Service, Event Services, Notification Service) interact with MongoDB for persistence and retrieval.

5.3.2 Mill Set Up

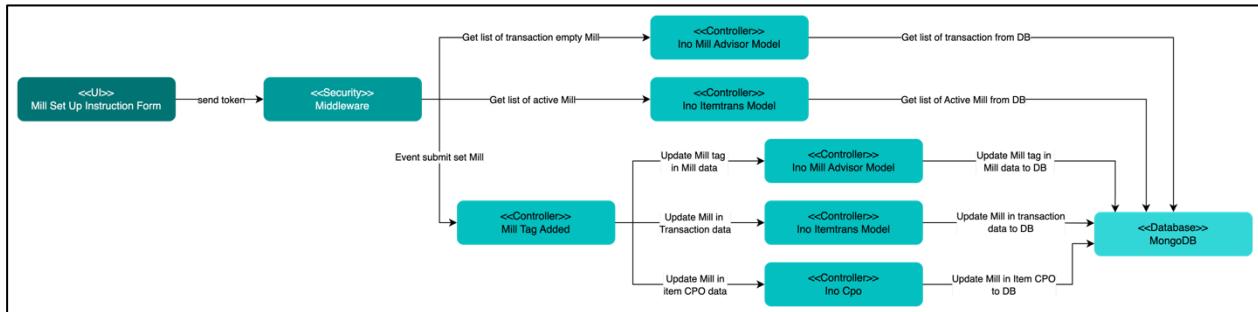


Figure 25. Mill Set Up Component Design

5.3.2.1 User Interface

The Mill Set Up Instruction form serves as the user entry point for assigning or tagging a mill to various records in the system. When a user submits the form, it sends an authentication token to the backend. This action initiates the mill setup workflow, updating mill assignments in multiple data models, such as transaction records and CPO (Certificate of Product Origin) items.

5.3.2.2 Security

The Middleware layer authenticates the token submitted with the request. Once verified, it allows the request to proceed to the appropriate controllers. This layer enforces security by ensuring only authorized users can assign or update mill data in the system.

5.3.2.3 Controller

5.3.2.3.1 Mill Tag Added

This controller is triggered when the form is submitted. It orchestrates the full mill setup process by delegating updates to three downstream models: Ino Mill Advisor Model, Ino Itemtrans Model, and Ino Cpo. Each model is responsible for updating a specific set of data related to the mill assignment.

5.3.2.3.2 Ino Mill Advisor Model

This component fetches the list of transactions where the mill tag is empty from the database. It then updates the appropriate mill tag fields in the mill-related records and saves the changes back to the database. This ensures that the advisor-related mill data is correctly tagged.

5.3.2.3.3 Ino Itemtrans Model

This component is responsible for fetching and updating mill information in transaction-related data. It reads the list of active mills, updates the corresponding records with the new mill tag, and persists the updated data to MongoDB.

5.3.2.3.4 Ino Cpo

This component handles updates to mill information in CPO item data. It ensures that product origin records are updated with the correct mill tag. Like the others, it writes the updated data to the database after validation.

5.3.2.4 Database

MongoDB is used as the backend data store for all mill-related operations. It contains collections for transaction data, mill advisor records, item transaction logs, and CPO data. All three service components (Ino Mill Advisor Model, Ino Itemtrans Model, and Ino Cpo) perform read and write operations against this database, ensuring the updated mill tag is saved across the system consistently.

5.4 Inventory

5.4.1 Stock on Hand

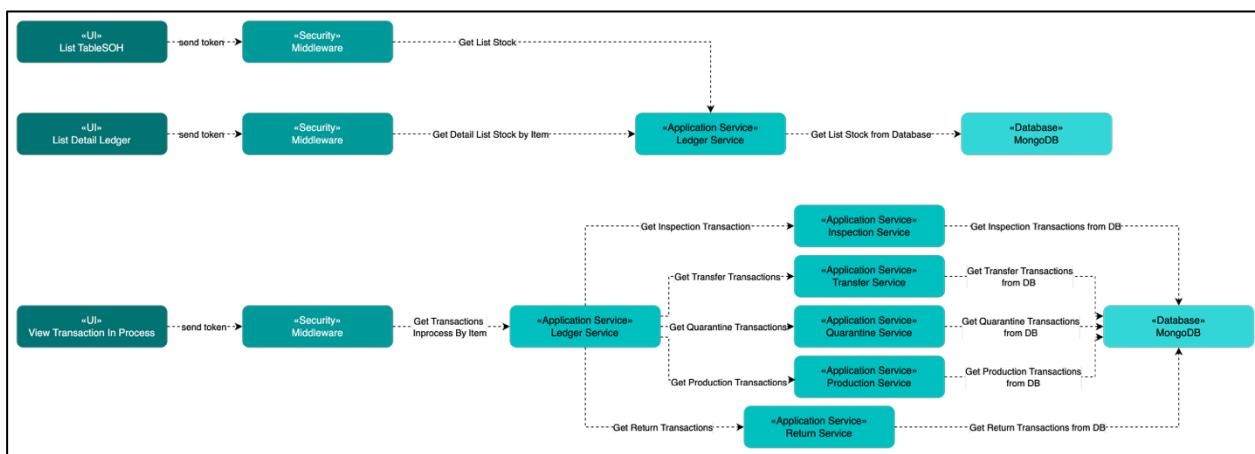


Figure 26. Stock on Hand Component Design

5.4.1.1 User Interface

5.4.1.1.1 List TableSOH

This form allows users to retrieve an overview of the stock-on-hand (SOH). When the form is submitted, a token is sent for authentication. The request then proceeds to fetch summarized stock data across all items from the backend.

5.4.1.1.2 List Detail Ledger

This interface lets users view detailed stock information item by item. It provides deeper insight than the TableSOH, helping users analyze specific stock movements. A token is sent for authentication before processing the request.

5.4.1.1.3 View Transaction in Process

This UI component displays all ongoing or pending transactions. It helps users monitor real-time activity, such as stock transfers, inspections, quarantines, production usage, and returns. As with other forms, the user submits a token to authenticate the request.

5.4.1.2 Security

For all three UI components, the middleware validates the user's token to ensure that only authenticated users can access stock and transaction data. It functions as the central security checkpoint and passes verified requests to the appropriate application service.

5.4.1.3 Application Services

5.4.1.3.1 LedgerService

The Ledger Service handles both the TableSOH and Detail Ledger views. It fetches stock data from MongoDB—either summarized (for the SOH table) or item-specific (for the detail view). It is also responsible for retrieving in-process transactions and aggregating data from several other services.

5.4.1.3.2 InspectionService

This service retrieves all inspection-related transactions from the database. When users view in-process data, Ledger Service calls this service to fetch relevant inspection entries and combine them into unified response.

5.4.1.3.3 TransferService

The Transfer Service handles stock movement or relocation transactions. It pulls transfer transaction records from the database for the current item in process view.

5.4.1.3.4 QuarantineService

The Quarantine Service is responsible for pulling quarantine transaction data. It works behind the scenes during transaction monitoring to provide insight into what items are currently held under quarantine.

5.4.1.3.5 ProductionService

This service retrieves production usage transactions, which refer to how stock is consumed or allocated in the production process. These transactions are also shown in the in-process transaction view.

5.4.1.3.6 ReturnService

The Return Service handles records of returned items, such as rejected stock or returned inventory. It contributes its data to the in-process overview and completes the picture of ongoing transactions.

5.4.1.4 Database

MongoDB is the persistent store for all stock and transaction-related data, including SOH summaries, inspection records, transfer logs, quarantines, production usage, and return entries. All application services access MongoDB to retrieve their respective data segments and serve it through the Ledger Service to the UI.

5.4.2 Transfer

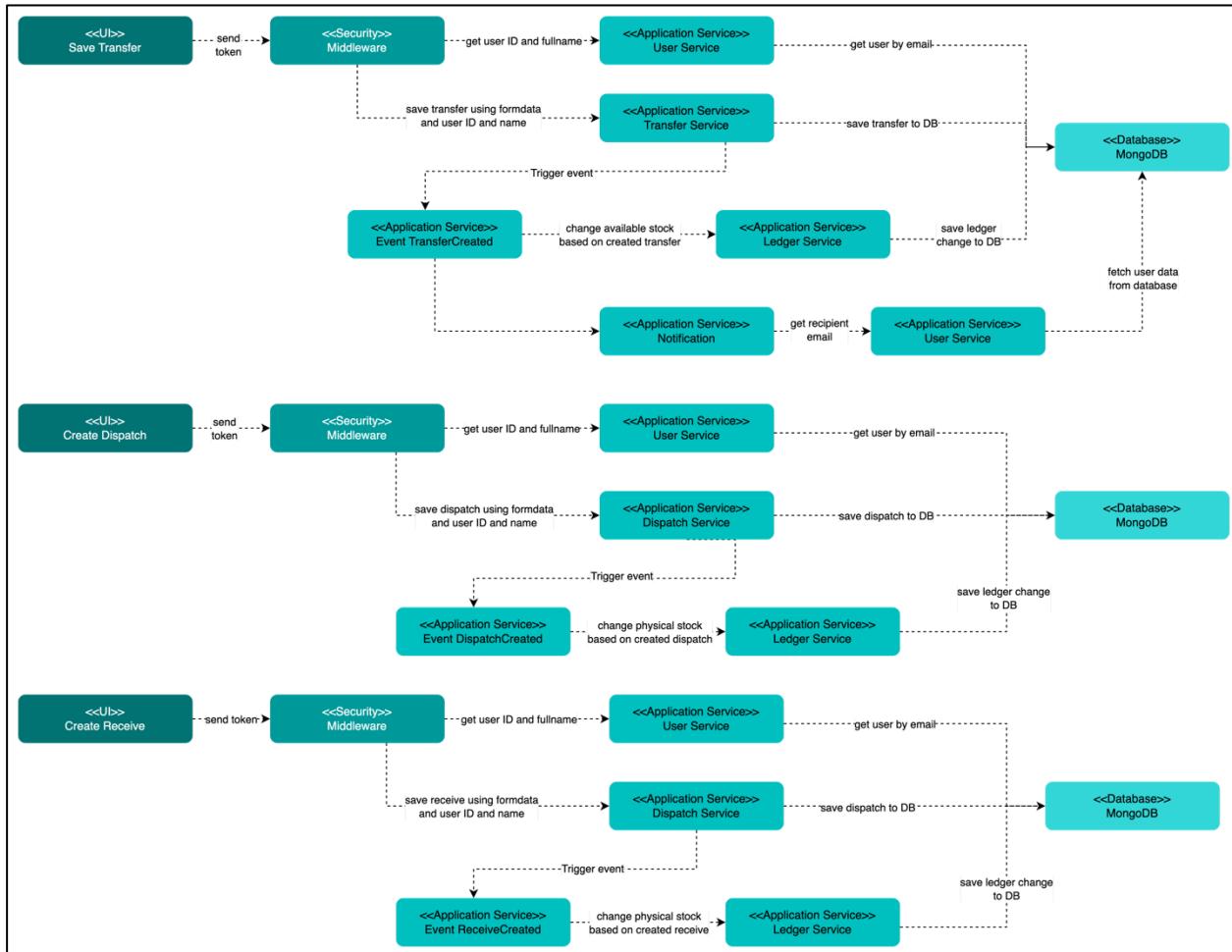


Figure 27. Transfer Component Design

5.4.2.1 User Interface

5.4.2.1.1 Save Transfer

This form allows users to initiate a stock transfer between locations or records. When the form is submitted, a token is included for authentication. The form captures transfer data such as source, destination, quantity, and triggers the transfer-saving process in the backend.

5.4.2.1.2 Create Dispatch

This UI component is used to log a dispatch operation, typically representing the physical movement of goods out of a source location. Upon submission, the form sends the dispatch data and user token, which triggers the dispatch flow.

5.4.2.1.3 Create Receive

This form is used to record the receipt of dispatched goods. Once the receiving data is submitted and authenticated, it triggers the receive flow, which includes ledger and stock updates reflecting the arrival of goods.

5.4.2.2 Security

For each of the three UI forms (Transfer, Dispatch, Receive), the middleware intercepts the request and validates the token to ensure the user is authenticated. After validation, the middleware fetches the user ID and full name to log the operation properly and then passes the request to the appropriate service layer.

5.4.2.3 Application Services

5.4.2.3.1 User Service

The User Service is called by all three workflows to fetch user information based on the authentication token. It retrieves the user ID, name and resolves the recipient's email if needed for notifications.

5.4.2.3.2 Transfer Service

This service processes the Save Transfer operation. It accepts formatted transfer data along with the user info, then saves the record into the database. It also triggers the EventTransferCreated service to update stock and ledger information accordingly.

5.4.2.3.3 EventTransferCreated

This event handler updates the available stock based on the submitted transfer. It invokes the Ledger Service adjusting stock levels and then persists with these changes to the ledger in the database.

5.4.2.3.4 Dispatch Service

The Dispatch Service handles the logic behind Create Dispatch and Create Receive requests. It saves the dispatch or receives records to the database and triggers the appropriate event (EventDispatchCreated or EventReceiveCreated) to apply stock and ledger changes.

5.4.2.3.5 EventDispatchCreated

Once a dispatch record is created, this event triggered to apply physical stock reduction. It calculates the stock to be deducted and updates the corresponding ledger records via Ledger Service.

5.4.2.3.6 EventReceiveCreated

Similarly, this event is triggered after a receive record is created. It updates the physical stock by increasing the count at the destination and adjusts the ledger to reflect the new quantity.

5.4.2.3.7 Ledger Service

This service is responsible for updating stock levels and maintaining accurate ledger records. It is called by event handlers (EventTransferCreated, EventDispatchCreated, and EventReceiveCreated) to persist all stock adjustments to the database.

5.4.2.3.8 Notification Service

Triggered after a transfer is created, the Notification Service sends alerts to the recipient via email. It coordinates with User Service to get the recipient's address, ensuring stakeholders are informed when a transfer is initiated.

5.4.2.4 Database

MongoDB stores all records related to transfers, dispatches, receives, and stock ledgers. It acts as the system of record for the backend, ensuring data consistency across operations. All services write to and read from MongoDB to persist transactions and maintain up-to-date inventory data.

5.5 Vendor Management

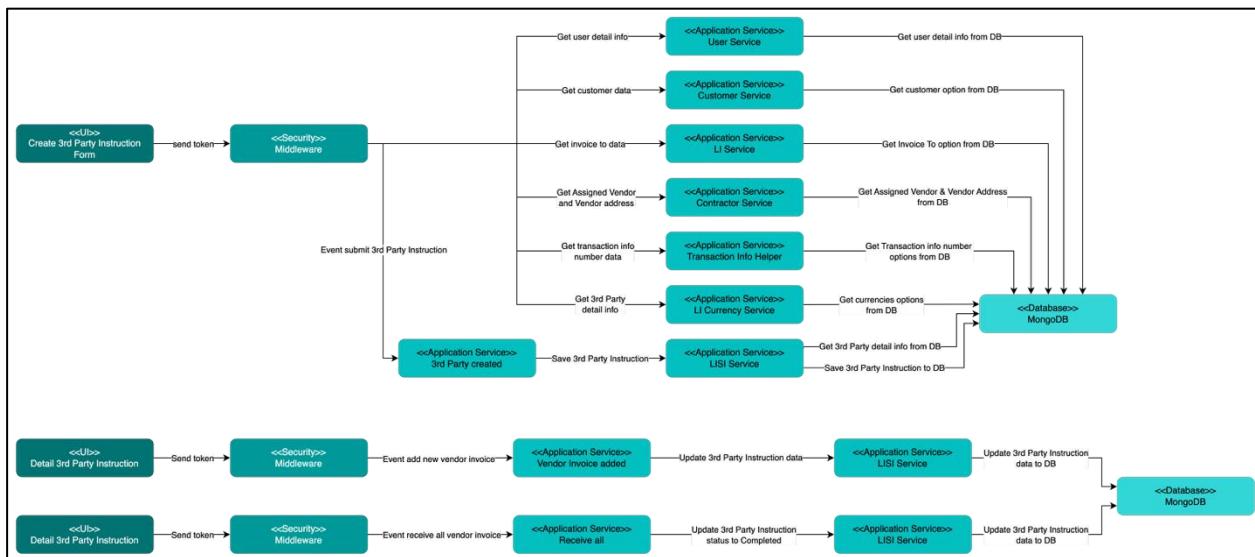


Figure 28. Vendor Management Component Design

5.5.1 User Interface

5.5.1.1 Create 3rd Party Instruction Form

This form allows users to initiate the creation of a 3rd Party Instruction, which is a structured set of tasks assigned to external vendors or partners. When submitted, it sends a security token to authenticate the user and then collects various supporting data (user info, customer, vendor, invoice, transaction) needed to create the instruction.

5.5.1.2 Detail 3rd Party Instruction

This UI is used to display and manage the details of an already-created 3rd Party Instruction. It is the interface where vendor invoices can be reviewed and updated. Like the others, the token is sent to ensure that the action is performed by an authorized user.

5.5.2 Security

For both forms, the middleware is responsible for token validation. It ensures that the user is authenticated before granting access to view, create, or update 3rd Party Instructions. After authentication, it forwards the request to the correct application services.

5.5.3 Application Services

5.5.3.1 User Service

Fetches user-specific information (ID, full name, etc.) from the database to populate the form and to track the user making the instruction.

5.5.3.2 Customer Service

Retrieves customer-related data required in the 3rd Party Instruction form, including the list of available customer options.

5.5.3.3 LI Service

Fetches invoice-related information labeled as Invoice To that must be linked to the 3rd Party Instruction.

5.5.3.4 Contractor Service

Gets assigned vendor details and their addresses that are associated with the 3rd Party Instruction. These vendors will execute the instructions.

5.5.3.5 Transaction Info Helper

Returns transaction reference numbers that are relevant to this 3rd party transaction, useful for financial or operational tracking.

5.5.3.6 LI Currency Service

Provides available currency options from MongoDB to be used in the instruction form.

5.5.3.7 LISI Service

This is the core service responsible for saving and updating 3rd Party Instructions. It handles initial saving operations when a new instruction is created, as well as updates during later stages, such as vendor invoice submission and final status completion.

5.5.3.8 3rd Party Created

This service is triggered when a user submits the instruction form. It orchestrates the saving process and may call multiple supporting services to gather and validate required data. Once done, it passes the information to the LISI Service for storage.

5.5.3.9 Vendor Invoice Added

This service is called when a new vendor invoice is submitted for instruction. It updates the 3rd Party Instruction data with the new invoice entry and hands it off to LISI Service for persistence.

5.5.3.10 Receive All

Triggered when all expected vendor invoices are received. This service updates the instruction's status to Completed to indicate that all vendor actions have been fulfilled.

5.5.4 Database

MongoDB stores all data related to 3rd Party Instructions, including user details, customer info, vendor assignments, invoices, currencies, and transaction references. It is accessed by all supporting services, especially LISI Service, for both read and write operations. Data is written during creation and invoice updates, and status is modified upon completion.

5.6 Global Mill Advisor

5.6.1 Mill Audit Request

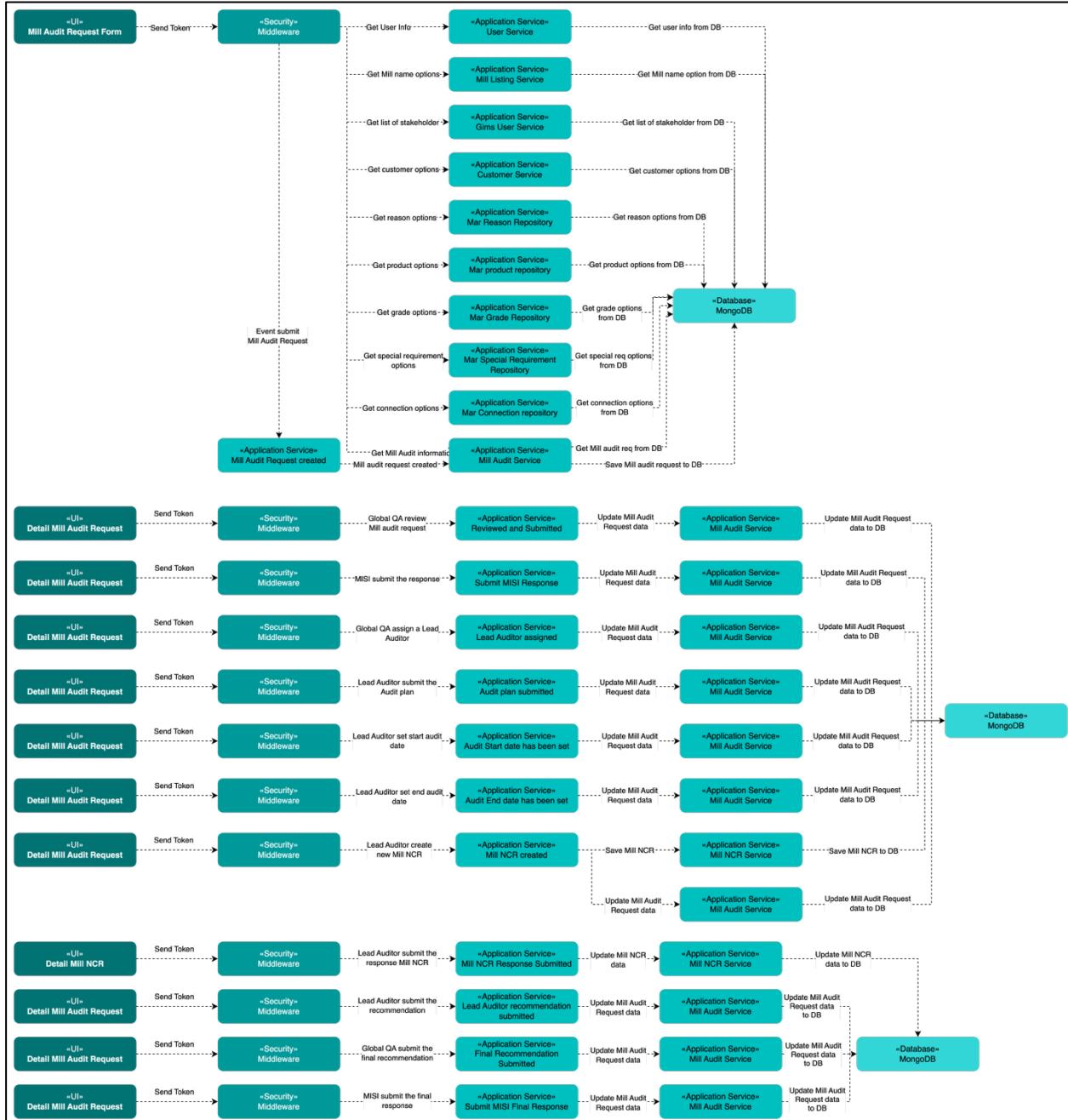


Figure 29. Mill Audit Request Component Design

5.6.1.1 User Interface

5.6.1.1.1 Mill Audit Request Form

This interface is used to create and submit a new mill audit request. It sends a token for authentication and requests several dropdown options (like customer, product, grade, connection types) before submitting the request.

5.6.1.1.2 Detail Mill Audit Request

Used to handle the entire lifecycle of the mill audit request, including assigning auditors, submitting audit plans, responding to audits, and converting issues into NCRs.

5.6.1.1.3 Detail Mill NCR

Specifically handles the flow after an NCR is created from a mill audit, including NCR responses and final follow-ups.

5.6.1.2 Security

All UI components send a token to the Middleware for:

- Authentication of user identity
- Authorization of access to mill audit data and workflows

Once verified, the middleware allows interaction with the backend services.

5.6.1.3 Application Services

5.6.1.3.1 Initial Data and Form Options

Several services are responsible for retrieving dropdown or option data used in the form:

User Service: Retrieves user info

Mill Listing Service: Fetches mill name options

Gims User Service: Gets stakeholder list

Customer Service: Retrieves customer options

Reason Repository, Product Repository, Grade Repository, Special Reason Repository, Connection Repository: Get option lists for reasons, products, grades, special cases, and connection types respectively

These services all pull their respective data from MongoDB.

5.6.1.3.2 Mill Audit Request Creation

- **Mill Audit Request Created:** Orchestrates creation of the mill audit request and gathers needed reference data.
- **Mill Audit Service:** Saves the request into MongoDB.

5.6.1.3.3 Mill Audit Request Lifecycle

Handled mostly in the **Detail Mill Audit Request** UI:

- **Submit Mill Response:** Saves audit response
- **Reviewed and Submitted:** Submits final audit review
- **Lead Auditor Assigned:** Assigns a lead auditor
- **Audit Plan Submitted:** Saves the audit plan
- **Audit Date Seen by Lead Auditor / Global QA:** Confirms audit scheduling
- **Mill NCR Created:** Converts audit issue into a new NCR

Each of these stages updates the mill audit request using the **Mill Audit Service** and persists changes to **MongoDB**.

5.6.1.3.4 NCR Handling Post-Audit

- **Mill NCR Service:** Creates the NCR record
- **Mill NCR Response Submitted:** Saves NCR response
- **Lead Auditor Review:** Updates NCR status with audit review
- **Submit NCR Final Response:** Completes and saves final resolution

All these services interact with MongoDB to store or update the data as part of the audit resolution.

5.6.1.4 Database

MongoDB stores:

- Audit request entries
- User and reference dropdown values (grade, product, reason)
- Responses and submitted forms
- NCR records and their resolution status

Almost all application services (Audit, NCR, Repositories) read from or write to MongoDB.

5.6.2 Mill Listing

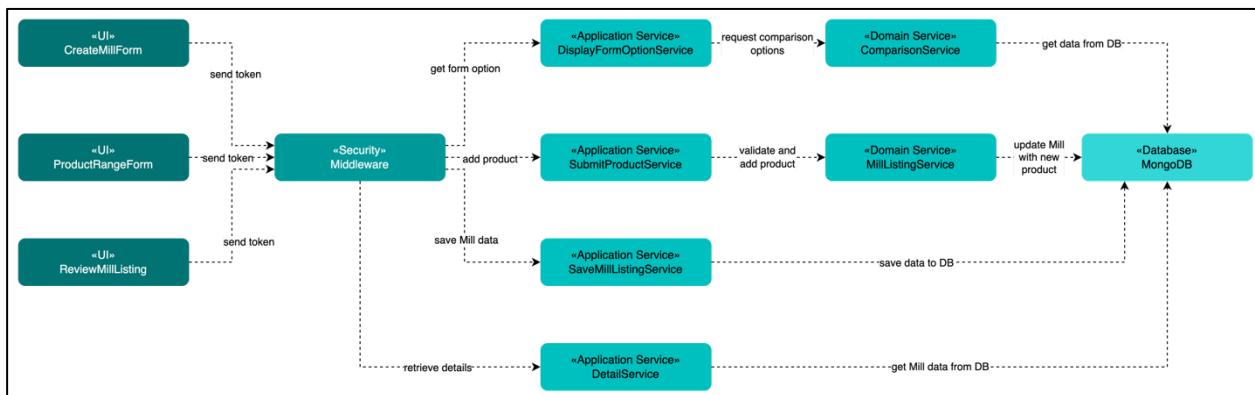


Figure 30. Mill Listing Component Design

5.6.2.1 User Interface

5.6.2.1.1 Create Mill Form

This form initiates the mill listing process. It allows users to begin registering or listing a mill, including selecting options related to comparison, location, and certification. Upon submission, a token is sent for authentication, and the form interacts with backend services to retrieve form options.

5.6.2.1.2 Product Range Form

This form is used to specify the product range for a mill listing. Users can add multiple products that the mill produces. Once a product is added, it is validated and processed through backend services to update the mill listing accordingly.

5.6.2.1.3 Review Mill Listing

This interface lets users review, verify, or finalize the details of a mill listing. It may be used before final submission or publication. The form sends a token for authentication and retrieves full details from the database via backend services.

5.6.2.2 Security

All UI forms go through the middleware, which validates the user token before allowing further actions. It ensures that only authorized users can submit, update, or retrieve mill listing information.

5.6.2.3 Application Services

5.6.2.3.1 Display Form Option Services

This service provides form selection options for mill creation. It fetches options such as comparison types (e.g., certifications or performance levels) from the domain layer to be shown in the form.

5.6.2.3.2 Submit Product Service

When a product is submitted through the ProductRangeForm, this service handles the validation and coordination of adding the product to the mill listing. It delegates the update operation to the domain service responsible for mill listing logic.

5.6.2.3.3 Save Mill Listing Service

This service saves the entire Mill listing after all required data has been provided. It handles persistence logic, saving the data to the database via domain logic.

5.6.2.3.4 Detail Service

This service is used in the review step. It retrieves full Mill listing data (including product range and comparison settings) from the database and returns it to the UI so users can view and confirm all entries before finalizing.

5.6.2.4 Domain Services

5.6.2.4.1 Comparison Service

This domain service provides business logic related to comparison options—such as matching mill features to standard benchmarks or certifications. It provides this data to the DisplayFormOptionService.

5.6.2.4.2 Mill Listing Service

This domain service is responsible for handling core logic of Mill listing, including:

- Updating mill data with newly added products
- Validating listing structure
- Preparing the data to be saved to MongoDB.

5.6.2.5 Database

MongoDB stores all Mill listing-related data including:

- General Mill info
- Product range
- Comparison settings
- Detailed listing records.

The database is accessed by domain services during both write (submit, update) and read (review) operations.

5.6.3 Mill NCR

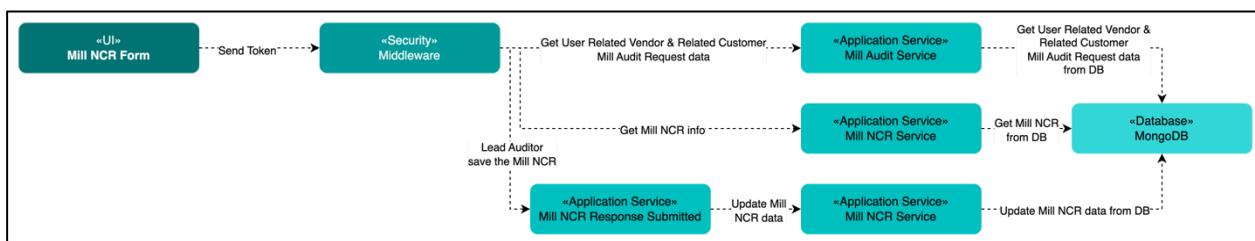


Figure 31. Mill NCR Component Design

5.6.3.1 User Interface

Mill NCR Form component is the user interface used by the Lead Auditor to submit a Non-Conformance Report (NCR) that is related to a previous Mill Audit Request. When the form is used, it sends a token for user authentication.

5.6.3.2 Security

The Middleware layer verifies the token and handles authentication. After successful verification, it facilitates communication with backend services. It ensures that only authenticated users can fetch or submit NCR-related data.

5.6.3.3 Application Services

5.6.3.3.1 Mill Audit Service

Responsible for retrieving user-related vendor and customer data tied to previous Mill Audit Requests. It queries about the database to pull contextual data relevant to the Mill NCR being created.

5.6.3.3.2 Mill NCR Service

This service performs two main functions:

1. **Get Mill NCR info** from the database for display or review.
2. **Update Mill NCR data** when a response is submitted or revised. It is the central component for saving and updating NCR records tied to Mill Audit Requests.

5.6.3.3.3 Mill NCR Response Submitted

This is a specific application service triggered when the Lead Auditor submits the NCR response. It calls the Mill NCR Service to persist the updated data into the database.

5.6.3.4 Database

MongoDB is used to store:

- Mill Audit Request data including vendor and customer context
- Mill NCR records
- Updates from NCR response submissions

It is accessed by both Mill Audit Service (to get audit request data) and Mill NCR Service (to get and update NCR details).

5.6.4 Mill MoC

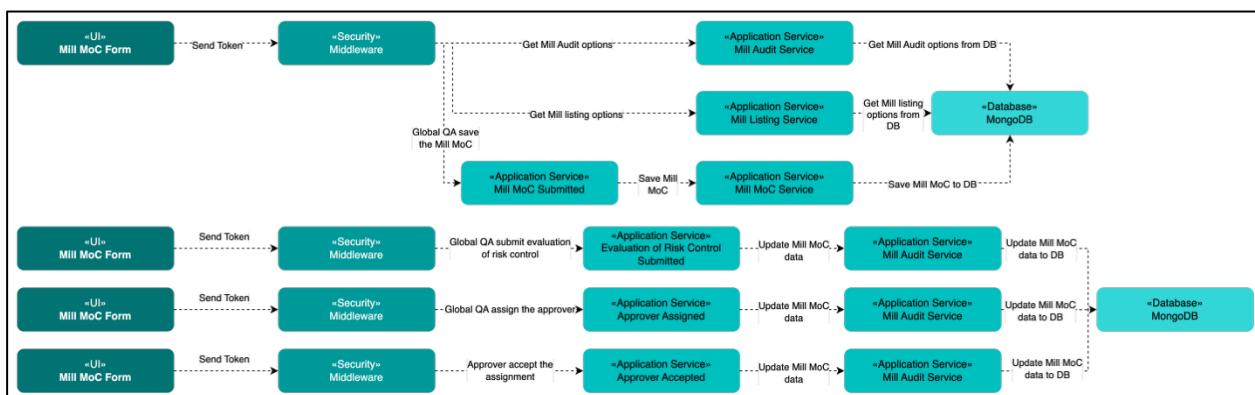


Figure 32. Mill MoC Component Design

5.6.4.1 User Interface

Mill MoC form allows the Global QA team and Approvers to create, evaluate, assign, and approve Mill MoC records. Each interaction from this UI sends a token for authentication via middleware.

5.6.4.2 Security

Middleware is responsible for validating the authentication taken before any data request or submission occurs. It ensures that only authorized users can access and interact with the Mill MoC services.

5.6.4.3 Application Services

5.6.4.3.1 Mill Audit Service

Retrieves a list of Mill Audit options from MongoDB to populate the selection fields in the Mill MoC form. It is also used later to update Mill MoC data into the database during evaluation, assignment, and approval phases.

5.6.4.3.2 Mill Listing Service

Provides a list of available mill listings used to associate the MoC with a particular Mill. It fetches this data from MongoDB.

5.6.4.3.3 Mill MoC Submitted

Triggered when the Global QA saves the Mill MoC form. This service processes and routes the form data to the appropriate MoC service for persistence.

5.6.4.3.4 Mill MoC Service

Responsible for saving the Mill MoC record to the database. It handles the creation of new MoC records and is reused in the subsequent update steps.

5.6.4.3.5 Evaluation of Risk Control Submitted

Invoked when the Global QA submits the risk control evaluation. It passes updated data to the Mill Audit Service, which then updates the Mill MoC record in the database.

5.6.4.3.6 Approver Assigned

Triggered when a Global QA assigns an approver to review and approve the MoC. It results in an update to the Mill MoC record with assigned user info.

5.6.4.3.7 Approver Accepted

Called when the Approver accepts their assignment. This updates the Mill MoC record's status to reflect the acceptance.

5.7 Setup

5.7.1 User



Figure 33. Setup User Component Design

5.7.1.1 User Interface

The user fills out the registration form. Upon submission, a token is sent for authentication and user data is forwarded to the backend for processing. No complex logic is handled in the UI other than displaying the form and managing server responses.

5.7.1.2 Security

The middleware layer is responsible for ensuring that any request reaching the backend is authenticated. It checks the token sent from the UI against the system's authentication mechanism. If the token is valid, the request proceeds to the relevant controller for further handling. If the token is invalid or missing, the middleware blocks the request, ensuring that only authenticated users or processes can access secure backend operations.

5.7.1.3 Controller

5.7.1.3.1 Admin Controller

The Admin Controller handles both user registration and account activation. During registration, it begins by validating user input such as email and name. It checks the ino_user table to verify whether the email has already been used. If the user is not associated with MITME, the controller retrieves client data from the database where the is_lowest flag is set to 'n', and assigns that data to the customer field. It then sets the appropriate user_type and initializes the account status as Pending. This data is submitted to the Single Sign-On (SSO) system with a pending status. The controller also logs this action for audit purposes and triggers an email that includes an account activation link.

When the user clicks the activation link, the Admin Controller processes the request to activate the account. It verifies the link and, if valid, updates the user status to Activ" in both the local system and the SSO. After this, it forwards the user data to the Client Controller for database persistence.

5.7.1.3.2 Client Controller

The Client Controller is responsible for persisting in user data after successful account activation. It receives the activated user data from the Admin Controller, formats it as needed, and sends it to the database for permanent storage. This ensures that only users who have completed validation and activation are saved into the backend system, maintaining data integrity and supporting secure user management across the application.

5.7.1.4 Database

The database component uses MongoDB to store finalized user records. Only users who have been validated and activated are stored here. The database holds critical user information including email, name, user type, account status, and customer association. MongoDB serves as the central repository for all active users, enabling query access, reporting, integration, and secure long-term storage.

5.7.2 Privilege

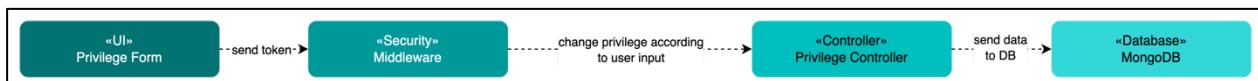


Figure 34. Setup Privilege Component Design

5.7.2.1 User Interface

The user interface begins with the Privilege Form, which allows an authorized user (such as an admin) to assign or modify privileges for a specific user account. Once the form is submitted, the request includes the updated privilege configuration along with a token for authentication. The UI component is primarily responsible for capturing user input and sending it to the backend for processing.

5.7.2.2 Security

Middleware acts as a security layer that validates the authentication token received from the UI. This ensures that only authenticated users with the correct permissions are allowed to make changes to user privileges. If the token is valid, the request proceeds to the controller. Otherwise, the request is rejected, and the privilege change is not executed.

5.7.2.3 Controller

The Privilege Controller handles all backend logic related to updating user privileges. Upon receiving a valid and authenticated request, it processes the user input, verifies the data, and updates the user's access rights or roles accordingly. This may involve granting or revoking access to specific features, modules, or data scopes. Once the privilege changes are processed, the controller formats the data and sends it to the database for permanent storage.

5.7.2.4 Database

The MongoDB database stores the updated privilege information for each user. This includes any roles, access levels, or permissions that have been assigned or revoked. By saving this data, the system ensures that the user's access configuration is consistently enforced across the application and persists even after system restarts or deployments.

5.7.3 Tablet Access

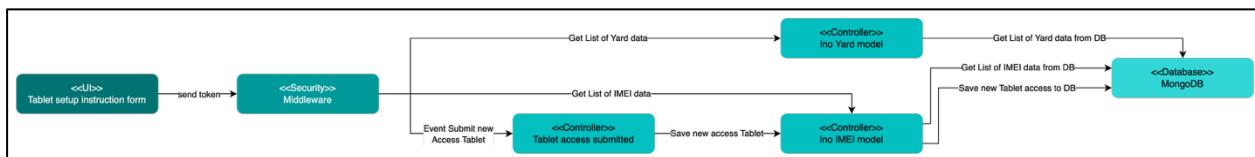


Figure 35. Setup Tablet Component Design

5.7.3.1 User Interface

The Tablet Setup Instruction form is used to configure or register a tablet device (likely used for yard operations) into the system. The user selects from the available yard locations and IMEI numbers, then submits the form with a valid token. This action initiates the creation of new tablet access records tied to a specific device and location.

5.7.3.2 Security

Once the user submits the tablet setup form, the middleware first checks the provided token to authenticate the request. Only authenticated users can proceed to register new tablet access. After validation, the middleware forwards the request to the corresponding controller to process the data.

5.7.3.3 Controller

5.7.3.3.1 Tablet Access Submitted

This controller handles the core logic of the tablet access submission. After receiving the request, it orchestrates data fetching from two services: one for yard data and another for tablet (IMEI) data. Once these references are resolved, it passes the completed access data to be saved by the appropriate controller.

5.7.3.3.2 Ino Yard Model

This service fetches the list of available Yard locations from the database. These locations are associated with the tablet device to define its operating context or assignment.

5.7.3.3.3 Ino IMEI Model

This service retrieves the list of registered IMEI numbers (unique device identifiers) from the database. It ensures the tablet being set up is recognized and its identifier is stored for future reference or security validation.

5.7.3.4 Database

MongoDB is the central repository where all configuration data is stored. In this flow, MongoDB stores yard data, IMEI numbers, and the new tablet access information

submitted through the form. The access record links a tablet's IMEI to a yard and allows its participation in operational processes.

5.7.4 Contractor

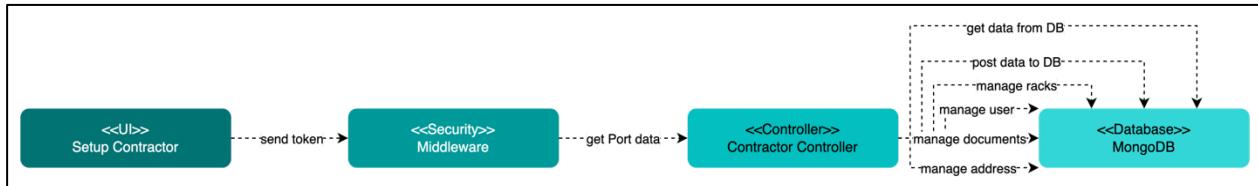


Figure 36. Setup – Contractor Component Design

5.7.4.1 User Interface

The Setup Contractor form is used by users to configure contractor-related data, such as creating, updating, or managing vendors/contractors involved in the operation. When a user accesses or submits this form, it sends an authentication token to the backend and may include contractor-specific inputs like name, port, address, or associated documents.

5.7.4.2 Security

The middleware ensures that all requests coming from the Setup Contractor form are authenticated. It verifies the token provided with the request and only allows valid, authorized users to proceed with data access or configuration. Once authenticated, it forwards the request to the Contractor Controller.

5.7.4.3 Controller

The Contractor Controller manages the business logic related to contractor setup. It retrieves port data, saves new contractor configurations, and handles multiple related components such as:

- User management (e.g., linked users under a contractor)
- Rack assignments (if applicable)
- Document management (e.g., licenses, contracts)
- Address registration and updates

This controller acts as the central point of orchestration for all contractor-related activities.

5.7.4.4 Database

MongoDB is the primary database storing all contractor-related information. It includes:

- Port data linked to contractors
- Contractor profiles
- Associated users and racks
- Documents and address information

The Contractor Controller interacts with MongoDB to fetch and persist all contractor data as needed for operational readiness.

5.7.5 Port

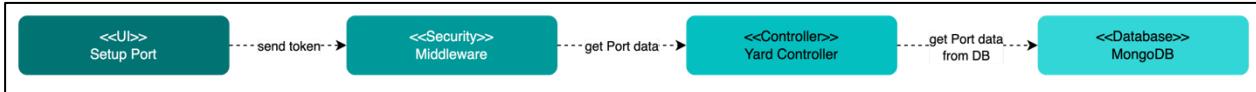


Figure 37. Setup – Port Component Design

5.7.5.1 User Interface

The Setup Port form allows users to configure or select port-related data for operational use—such as defining which port is connected to a Yard or device. When submitted, it sends a token for authentication and triggers a request to retrieve relevant port data for display or further processing.

5.7.5.2 Security

The middleware acts as a secure gatekeeper. Upon receiving the request from the Setup Port form, it validates the token to ensure that the user is authorized. Once validated, the request is passed to the controller responsible for fetching port data.

5.7.5.3 Controller

The Yard controller is responsible for handling logic related to yard and port configurations. When called, it queries MongoDB to fetch the required Port data, which may include port codes, names, locations, or connectivity information. The controller then prepares this data to be returned to the frontend for user interaction or selection.

5.7.5.4 Database

MongoDB stores all Port-related information, such as configurations, assignments, and metadata. When the Yard Controller requests data, it queries MongoDB to retrieve up-to-date information for use in the Setup Port process.

5.7.6 Yard

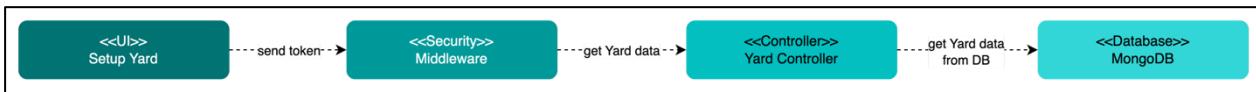


Figure 38. Setup – Yard Component Design

5.7.6.1 User Interface

The Setup Yard form allows users to configure or retrieve Yard-related information, such as setting up or managing yard areas used in operations (e.g., storage or quarantine areas). When the user accesses or submits this form, it sends a token to authenticate the request and initiate data retrieval.

5.7.6.2 Security

The middleware acts as the authentication layer. Upon receiving the token from the form, it verifies the user's identity. Once validated, it allows the request to continue and forwards it to the appropriate backend controller to fetch yard data.

5.7.6.3 Controller

The Yard controller handles the logic for retrieving Yard data from the database. It executes a query to MongoDB to fetch yard configurations or existing yard records and passes the data back to the front end for display or further setup processing.

5.7.6.4 Database

MongoDB serves as the database storing all Yard-related records, including yard names, types, statuses, and configurations. The Yard Controller accesses this data source to retrieve up-to-date information when users interact with the Setup Yard form.

5.8 Support

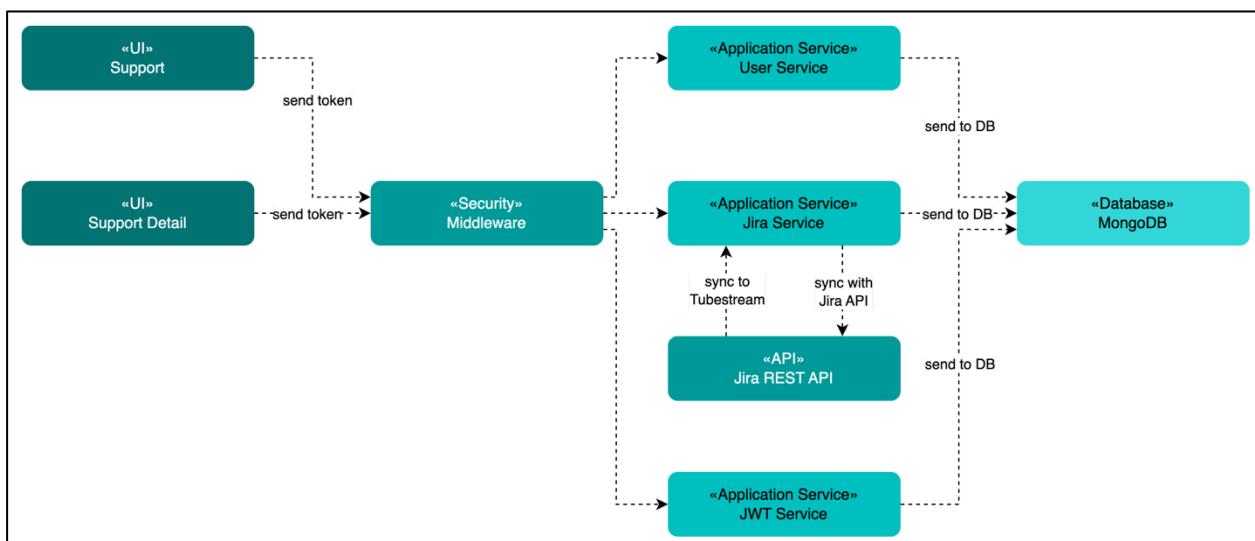


Figure 39. Support Component Design

5.8.1 User Interface

5.8.1.1 Support

This is the main user interface where users can access the support system (e.g. submitting issues or viewing ticket summaries). It sends a token to authenticate the user before accessing support data.

5.8.1.2 Support Detail

This UI displays a detailed view of a specific support case or ticket. Users can see more information, track progress, or make updates. Like the main Support UI, it also sends a token for security validation.

5.8.2 Security

The Middleware component authenticates the token sent by the UI components. It ensures that only valid, authenticated users can access the services. After validating, it allows access to other application services (e.g., User Service, Jira Service).

5.8.3 Application Services

5.8.3.1 User Service

Retrieves user data necessary for support tracking or reporting. Send user-related data (e.g., name, role, ID) to MongoDB for logging or linking tickets to specific users.

5.8.3.2 Jira Service

This is the core service for handling support tickets. It:

- Saves ticket-related data into MongoDB
- Synchronizes ticket info with:
 - Tubestream (an internal system)
 - Jira REST API (external service for ticket tracking and management)

5.8.3.3 JWT Service

Provides identity and authentication logic using JSON Web Tokens. Likely involved in securing API calls or verifying user sessions.

5.8.4 API Component

An external API interface that allows this system to create, update, or sync support tickets with the Jira platform. The Jira Service communicates with this API and pushes any updates back to MongoDB for internal tracking.

5.8.5 Database

MongoDB acts as the primary data store for all support-related information, including:

- Support tickets submitted through the UI
- User data associated with tickets
- Sync logs or status updates from Jira and Tubestream

All services (User Service, Jira Service, and the external Jira REST API) store and retrieve data here to ensure consistent access and reporting.

6. API Endpoints

API endpoints serve as the communication bridge between different parts of a web application, allowing the frontend and backend to interact seamlessly. They define specific routes for requesting, sending, and managing data, ensuring structured and

secure data exchanges. The following tables provide details on the available API endpoints, including their URLs and functionalities.

6.1 Quality & HSE

6.1.1 SPR / NCR

6.1.1.1 List/Index Page

Table 7. NCR – List/Index Page

API Name	API Route	HTTP Method
Get List of NCR (Tab = open, in_progress, rejected, completed)	/api/ncr_api/	GET
Get List of NCR Type for export excel NCR list	/ncr_api/ncr/ncr_list_export	GET
Get Link Excel NCR list for download it	/ncr_api/ncr/generate_ncr_list	POST

6.1.1.2 Create & Edit NCR

Table 8. NCR – Create & Edit 3rd Party Instruction

API Name	API Route	HTTP Method
Get several options (embed = causal_factors, customers, departments, vendors, items, levels, currency, mill)	/ncr_api/enquiry/options	GET
Get options of Subcategory field	/ncr_api/categories	GET
Add Attachment	/ncr_api/enquiry/add_attachment	POST
Save NCR (Draft/Submit)	/ncr_api/ncr/save	POST

6.1.1.3 Detail 3rd Party Instruction

Table 9. Detail NCR API Endpoints

API Name	API Route	HTTP Method
Get Detail of NCR	/ncr_api/detail/{id}	GET
Get options of HSE options (types = title,	/ncr_api/hse/options	GET

API Name	API Route	HTTP Method
(sex, employment_type, phase_workday, role, incident_type, incident_location, agent_object, timing, environment_condition, surface_condition, type_ppe_worn, injury_type, treatment_type)		
Reject NCR	/ncr_api/reject_ncr/{id}	POST
Accept NCR	/ncr_api/accept_ncr/{id}	POST
Display options of Lead Investigator	/ncr_api/ncr_users/{ncr_id}/lead_investigator	GET
Display options of Action Owner	/ncr_api/ncr_users/{ncr_id}/action_owner	GET
NCR Manager assign the new Role and Task	/ncr_api/role_and_task/assign_role_task/{ncr_id}	POST
NCR Manager modify the Role and Task	/ncr_api/role_and_task/modify_assign_role_task/	POST
NCR Manager delete the Role and Task	/ncr_api/role_and_task/delete_role_task/	POST
Lead Investigator/Action Owner reject the Role and Task	/ncr_api/role_and_task/reject_task/{ncr_id}	POST
Lead Investigator/Action Owner accept the Role and Task	/ncr_api/role_and_task/accept_task/	POST

API Name	API Route	HTTP Method
Lead Investigator/Action Owner submit the Action Task (Task Requested)	/ncr_api/role_and_task/save_action_task/	POST
NCR Manager reject the Action Task (Task Requested)	/ncr_api/role_and_task/reject_action/	POST
NCR Manager approve the Action Task (Task Requested)	i/ncr_api/role_and_task/accept_action/	POST
Lead Investigator save the HSE Incident	/ncr_api/ncr/{ncr_id}/hse/incident	POST
Lead Investigator save the HSE Injury	/ncr_api/ncr/{ncr_id}/hse/injury	POST
Lead Investigator save the HSE Equipment	/ncr_api/ncr/{ncr_id}/hse/equipment	POST
Lead Investigator save the HSE Environment	/ncr_api/ncr/{ncr_id}/hse/environment	POST
Save Initial Actions & Comments	/ncr_api/initial_action_comment/save	POST
Set Final Review	/ncr_api/{ncr_id}/final_review	POST
Upload Attachment	/ncr_api/upload_attachment/	POST
Save Final Reviews & Comments	/ncr_api/save_final_action_comment	POST
NCR Manager assign the Acknowledger	/ncr_api/acknowledger/assign	POST

API Name	API Route	HTTP Method
NCR Manager modify the Acknowledger	/ncr_api/acknowledger/modify	POST
Acknowledger reject the Assignment	/ncr_api/acknowledger/reject	POST
Acknowledger accept the Assignment	/ncr_api/acknowledger/accept	POST

6.1.2 MoC

6.1.3 Inspection Report

6.1.3.1 Ageing Inspection

Table 10. Inspection – Ageing Inspection API Endpoints

API Name	API Route	HTTP Method
Get Aging Inspection Data	/api/inspection_api/aging_inspection/get_data/{type}/{mode?}	GET
Export Aging Inspection Data	/api/inspection_api/aging_inspection/export_data/{type}/{mode?}	GET
Save Skip Insurance Inspection	/api/inspection_api/aging_inspection/save_skip_insurance_inspection	POST
Get Advance Option Form for Aging Inspection	/api/inspection_api/aging_inspection/get_advance_option_form	GET

6.1.3.2 Create and Edit Inspection

Table 11. Create and Edit Inspection API Endpoints

API Name	API Route	HTTP Method
Get Inspection Privileges	/api/inspection_api/get_privilege	GET

API Name	API Route	HTTP Method
Get Inspection Privileges	/api/inspection_api/get_privilege	GET
Download List in Excel Format	/api/inspection_api/list/download	GET
Get User Information	/api/inspection_api/get_user	GET
Get Form for Editing Customer	/api/inspection_api/get_form_edit_customer/{id}	GET
Submit Edited Customer Information	/api/inspection_api/submit_edit_customer	POST
Get Form for Editing Customer Reference	/api/inspection_api/get_form_edit_customer_ref/{id}	GET
Submit Edited Customer Reference	/api/inspection_api/submit_edit_customer_ref	POST
Get Form for Editing Quantity	/api/inspection_api/get_form_edit_qty/{id}	GET
Get Scope of Work for Inspection	/api/inspection_api/scope_of_work/{id}	GET
Submit Edited Inspector Information	/api/inspection_api/submit_edit_inspector	POST
Get List of Inspectors	/api/inspection_api/get_form_edit_inspector/{id}	GET

API Name	API Route	HTTP Method
Get Data for Scope of Work Template	/api/inspection_api/data_template_sow/{type}	GET
Get Scope of Work Template	/api/inspection_api/get_sow_template/{id}	GET
Delete Scope of Work Template	/api/inspection_api/delete_template_sow	POST
Submit Edited Unit Length	/api/inspection_api/submit_edit_unit_length	POST
Get Scope of Work Form	/api/inspection_api/form_scope_of_work/{id}	GET
Get Scope of Work Template Options	/api/inspection_api/get_sow_template_options	GET
Get Scope of Work Level Template Options	/api/inspection_api/get_sow_level_template_options/{id}	GET
Submit Edited Scope of Work	/api/inspection_api/submit_edit_sow	POST
Generate New Item for Inspection	/api/inspection_api/get_new_item/{id}	GET
Submit Inspection Data	/api/inspection_api/submit_data_inspection	POST
Save Auto-populate Data for Inspection	/api/inspection_api/save_autopopulate_data	POST
Get Auto-populated Inspection Data	/api/inspection_api/get_autopopulate_data/{id}	GET

API Name	API Route	HTTP Method
Populate Inspection from SOH	/api/inspection_api/populateSoh	POST

6.1.3.3 View Inspection Data

Table 12. Inspection – View Inspection Data API Endpoints

API Name	API Route	HTTP Method
View Inspection Data	/api/inspection_api/data/{type}	GET
Get Inspection Information	/api/inspection_api/inspection_information/{id}	GET
Get Item List for Inspection	/api/inspection_api/inspection_item_list/{id}	GET
Export Inspection Photos	/api/inspection_api/export_documents	POST
Download Inspection Final Report	/api/inspection_api/report/{id}	GET
Download Inspection Report as ZIP	/api/inspection_api/download_zip/{id}	GET
Download Inspection Checklist Report	/api/inspection_api/download_checklist/{id}	GET
Generate Revision PDF for Inspection	/api/inspection_api/generateRevisionPdf/{id}	GET
Download Inspection Review Report	/api/inspection_api/report_review/{id}	GET
Get Item List Form	/api/inspection_api/get_item_list_form/{id}	GET

API Name	API Route	HTTP Method
Add Attachment for Inspection	/api/inspection_api/add_attachment_inspection	POST
Regenerate Inspection Report	/api/inspection_api/regenerate_report/{idScreening}/{reportType}	GET
Get Terminate Inspection Data	/api/inspection_api/get_terminate_inspection/{id}	GET

6.1.3.4 Close Short, Terminate, and Reopen Inspection

Table 13. Inspection – Close Short, Terminate, and Reopen Inspection API Endpoints

API Name	API Route	HTTP Method
Close Short Inspection Parent	/api/inspection_api/closeshort_inspection_parent/{id}	POST
Terminate Child Inspection	/api/inspection_api/terminate_inspection_child/{id}	POST
Terminate Parent Inspection	/api/inspection_api/terminate_inspection_parent/{id}	POST
Reopen Inspection	/api/inspection_api/reopen_inspection	POST
Reopen Screening	/api/inspection_api/reopen_screening	POST

6.1.3.5 Scope of Work

Table 14. Inspection – Scope of Work API Endpoints

API Name	API Route	HTTP Method
Index Scope of Work Template	/api/inspection_api/sow_template/	POST
Create Scope of Work Template	/api/inspection_api/sow_template_create/	POST
Edit Scope of Work Template	/api/inspection_api/sow_template_edit/	POST

API Name	API Route	HTTP Method
Post Scope of Work Template	/api/inspection_api/sow_template_post/	POST
Update Scope of Work Template	/api/inspection_api/sow_template_update/{id}	POST
Set Default Scope of Work Template	/api/inspection_api/sow_template_default/	POST
Delete Scope of Work Template	/api/inspection_api/sow_template_delete/	POST

6.1.3.6 Screening Process

Table 15. Inspection – Screening Process API Endpoints

API Name	API Route	HTTP Method
Submit Move Rack Data	/api/inspection_api/submit_move_rack/{id}	POST
Submit Edited Quantity	/api/inspection_api/submit_edit_qty	POST
Get Screening Rack Information	/api/inspection_api/get_from_rack/{id}	GET
Get Edit Screening Rack Information	/api/inspection_api/get_edit_from_rack/{id}	GET
Save Edited Screening Rack Data	/api/inspection_api/submit_edit_from_rack	POST
Get Drift Header Information	/api/inspection_api/drift_header/{id}	GET
Generate Screening ID	/api/inspection_api/generate_screening_id/{id}	GET

API Name	API Route	HTTP Method
Display Batch Action for Inspection	/api/inspection_api/batch_action/{id}	GET
Get Pipe Number Options	/api/inspection_api/get_pipe_no_options	GET
Get Heat Number Options	/api/inspection_api/get_heat_no_options	GET
Clear Tally Inspection List	/api/inspection_api/screening/{id}/clear_tally_inspection	POST
Submit Item for Screening	/api/inspection_api/submit_item	POST
Get Inspection Form Options	/api/inspection_api/get_inspection_form_options/{id}	GET
Get Item Options for Inspection	/api/inspection_api/get_item_options	GET
Get Serial Number Options	/api/inspection_api/get_serial_no_options/{id}	GET
Get Form for Editing Unit Length	/api/inspection_api/get_form_edit_unit_length/{id}	GET
Save Screening Draft	/api/inspection_api/save_screening/{id}	POST
Delete Screening Item	/api/inspection_api/delete_screening_item	POST
Submit Screening Photos	/api/inspection_api/submit_screening_photos	POST

API Name	API Route	HTTP Method
Get Screening Photos	/api/inspection_api/get_screening_photos/{id}	GET
Submit for Review	/api/inspection_api/submit_for_review/{id}	POST
Get Detailed Item List	/api/inspection_api/get_detail_item_list/{id}	GET
Save Review Information	/api/inspection_api/save_review/{id}	POST
Get Inspection Note	/api/inspection_api/inspection_note/{id}	GET
Submit Yard Follow-up Information	/api/inspection_api/submit_for_yard_follow_up/{id}	POST
Get Batch Action Form	/api/inspection_api/get_batch_action_form/{id}	GET
Get Move Rack Item List	/api/inspection_api/move_rack_item_list/{id}	GET
Get Move Rack Options	/api/inspection_api/get_move_rack_option/{id}	GET
Get Inspection Form	/api/inspection_api/get_inspection_form/{id?}	GET
Get Type Options for Inspection	/api/inspection_api/get_type_options	GET
Set Default Scope of Work Template	/api/inspection_api/set_default_template	POST

API Name	API Route	HTTP Method
Submit Tally Data	/api/inspection_api/submit_data_tally	POST
Import Tally Data	/api/inspection_api/submit_import_tally	POST
Get Batch Options	/api/inspection_api/get_batch_options	GET
Save Data as Draft	/api/inspection_api/save_data_inspection	POST
Submit Screening Attachment	/api/inspection_api/submit_screening_attachment	POST
Delete Screening Attachment	/api/inspection_api/delete_screening_attachment	POST
Get Screening Attachment	/api/inspection_api/get_screening_attachment/{id}	GET
Get Item Length	/api/inspection_api/get_item_length	POST
Fetch Populated Inspection from SOH	/api/inspection_api/retrievePopulatedPopulatedSoh/{id}	GET
Reinspect from Review	/api/inspection_api/reinspect_partial	POST
Get Form for Editing Screening Item	/api/inspection_api/get_form_edit_screening_item/{id}	GET
Submit Edited Screening Item	/api/inspection_api/submit_edit_screening_item	POST

API Name	API Route	HTTP Method
Get Detailed Screening Inspection	/api/inspection_api/screening_inspection/{id}	GET
Get Inspection Detail	/api/inspection_api/inspection/{id}	GET
Post Data for Inspection	/api/inspection_api/post/	POST
Post Data for Review	/api/inspection_api/post_review/	POST
Post Data for Yard Follow-up	/api/inspection_api/post_action/	POST
Download Screening Report	/api/inspection_api/report_screening/{id}	GET
Download Tally Report	/api/inspection_api/report_tally/{id}	GET
Submit Edited Inspection Note	/api/inspection_api/submit_edit_inspection_note	POST
Get Tablet Options for Assigned Tablet	/api/inspection_api/get_form_edit_assigned_tablet/{id}	GET
Submit Edited Tablet Screening	/api/inspection_api/submit_edit_tablet_screening	POST
Get Tablet Options for Assigned Tablet	/api/inspection_api/get_form_edit_assigned_tablet/{id}	GET
Submit Edited Tablet Screening	/api/inspection_api/submit_edit_tablet_screening	POST

API Name	API Route	HTTP Method
Save Journal Inspection	/api/inspection_api/submit_journal_inspection/{id}	POST
Regenerate Report	/api/inspection_api/regenerate_report/{id}	GET
Get Inspection History for Reject Items	/api/inspection_api/get_inspection_history	POST
Save Certificate Date	/api/inspection_api/save_certificate_date	POST
Approve Multiple Items	/api/inspection_api/submit_item_approve_mass	POST

6.1.4 Quarantine Lot

6.1.4.1 Create & Edit Quarantine

Table 16. Quarantine – Create & Edit API Endpoints

API Name	API Route	HTTP Method
Get Create Quarantine Form	/api/quarantine_api/get_quarantine_form/{id?}/{adlInspectionReq?}	GET
Submit Create/Edit Quarantine Form	/api/quarantine_api/submit_quarantine_form	POST
Submit Attachment Form	/api/quarantine_api/add_attachment	POST
Get Item Option	/api/quarantine_api/get_item_options	GET
Get Batch List	/api/quarantine_api/get_batch_options	GET
Get Yard List	/api/quarantine_api/get_yard_options	GET

6.1.4.2 Detail & List Quarantine

Table 17. Quarantine – Detail & List API Endpoints

API Name	API Route	HTTP Method
Get Detail Quarantine By Id	/api/quarantine_api/quarantine/{id}	GET
Get List Data Quarantine By Status	/api/quarantine_api/data/{type}	GET
Get Batch List with Filter	/api/quarantine_api/get_filter_batch_options	GET
Get Yard List with Filter	/api/quarantine_api/get_filter_yard_options	GET
Get Item List with Filter	/api/quarantine_api/get_filter_item_options	GET
Download Excel List Quarantine	/api/quarantine_api/list/download	GET
Get Quarantine Privileges	/api/quarantine_api/get_privilege	GET
Get User	/api/quarantine_api/get_user	GET

6.1.4.3 Assign New Quarantine

Table 18. Quarantine – Assign New Quarantine API Endpoints

API Name	API Route	HTTP Method
Get Quarantine Order Form	/api/quarantine_api/get_quarantine_order_form/{id}	GET
Submit Quarantine Order Form	/api/quarantine_api/submit_quarantine_order	POST

6.1.4.4 Assign Release & Release Quarantine

Table 19. Quarantine – Assign Release & Release API Endpoints

API Name	API Route	HTTP Method
Get Release Quarantine Information	/api/quarantine_api/get_release_request_info/{id}	GET
Get New Release Quarantine Form	/api/quarantine_api/get_new_release_form/{id}	GET
Submit New Release Quarantine Form	/api/quarantine_api/submit_new_release_form	POST
Get Release Rack Assignment Quarantine Form	/api/quarantine_api/get_release_rack_assignment_form/{id}	GET
Submit Release Rack Assignment Quarantine Form	/api/quarantine_api/submit_release_rack_assignment_form	POST

6.1.4.5 Close Short & Terminate Quarantine

Table 20. Quarantine – Close Short & Terminate API Endpoints

API Name	API Route	HTTP Method
Get Terminate Quarantine Form	/api/quarantine_api/get_terminate_quarantine/{id}	GET
Submit Terminate Quarantine	/api/quarantine_api/terminate_quarantine	POST
Submit Closeshort Quarantine	/api/quarantine_api/closeshort_quarantine	POST
Get Terminate Quarantine Qty Information	/api/quarantine_api/get_terminate_info/{id}	GET

6.2 Commercial

6.2.1 Account Balance

6.2.1.1 Account Balance Header

Table 21. Account Balance Header API Endpoints

API Name	API Route	HTTP Method
Get Customers	/api/accbalance_api/accbalance/get_customer	GET

6.2.1.2 Account Balance Level Tabs & Items

Table 22. Account Balance Level Tabs & Items API Endpoints

API Name	API Route	HTTP Method
Get Tabs	/api/accbalance_api/accbalance/get_tabs_item	POST
Get Items	/api/accbalance_api/accbalance/get_item	POST
Export Account Balance Summary	api/accbalance_api/export/acc_balance_summary/{date_range}/{customer_id}/{condition}/{unit}	GET

6.2.1.3 Account Balance Level Subrow Item & Detail

Table 23. Account Balance Level Subrow Item & Detail

API Name	API Route	HTTP Method
Get Subrow Item	/api/accbalance_api/accbalance/get_item_subrow	POST
Get Detail Pending Approval & Approved	/api/accbalance_api/getModalIncoming	POST
Get Detail Opening Balance Customer Owned	/api/accbalance_api/getModalCustomer	POST
Get Detail Forecast	/api/accbalance_api/getModalForecast	POST
Submit Forecast	/api/accbalance_api/submitForecast	POST
Delete Forecast	/api/accbalance_api/deleteForecast	POST
Get Detail Dispatch	/api/accbalance_api/get_dispatch_modal	GET

API Name	API Route	HTTP Method
Get Detail Return	/api/accbalance_api/get_return_modal	GET

6.2.2 Commitment

6.2.2.1 Public Commitment

Table 24. Commitment – Public Commitment

API Name	API Route	HTTP Method
Get Motivation Quote	/api/commitment_api/get_motivation	GET
Generate Prefix Numbering (Tablet)	/api/commitment_api/generatePrefixNumberingTablet	POST
Sync Invoicing	/api/commitment_api/syncInvoicing	POST
Check Balance	/api/commitment_api/checkBalance	POST
Get Status Sync	/api/commitment_api/getStatusSync/{all?}	POST
Cancel Invoicing	/api/commitment_api/cancelInvoicing	POST
Back To Queue	/api/commitment_api/backToQueue	POST
Load Invoice	/api/commitment_api/loadInvoice/{id}	GET
Create Calloff Reference	/api/commitment_api/createCalloffReference	POST
Create Calloff Reference Option	/api/commitment_api/createCalloffReferenceOption	POST
Get Calloff From DO No	/api/commitment_api/getCalloffFromDONo/{id}	GET
Get Customer Options	/api/commitment_api/customer_options	GET
Generate Customer Owned	/api/commitment_api/generateCustomerOwned	POST
Get Edit Calloff Data	/api/commitment_api/getEditCallOffData/{id}	GET

API Name	API Route	HTTP Method
Get Balance Item SOH	/api/commitment_api/getBalanceItemSOH	POST
Get Item List Magic Button	/api/commitment_api/get_item_list_magic_button	POST

6.2.2.2 Excel Generation

Table 25. Commitment – Excel Generation API Endpoints

API Name	API Route	HTTP Method
Generate Commitment Excel	/api/commitment_api/generateCommitmentExcel	GET
Generate Sales Order Excel	/api/commitment_api/generateSalesOrderExcel	GET

6.2.2.3 General Routes

Table 26. Commitment – General Routes API Endpoints

API Name	API Route	HTTP Method
Get Privilege	/api/commitment_api/get_privilege	GET
Grouping Item	/api/commitment_api/groupingItem/{id}	POST
Get PRTE No	/api/commitment_api/getPrteNo	POST
Save PRTE	/api/commitment_api/savePrte	POST
Get Create Calloff Modal	/api/commitment_api/get_create_calloff_modal	POST
Get Options Item Selection	/api/commitment_api/get_options_item_selection	GET
Get Options Lot Selection	/api/commitment_api/get_options_lot_selection	POST
Submit Well	/api/commitment_api/submitWell	POST
Cancel ERP PRTE	/api/commitment_api/cancelErpPrte	POST

API Name	API Route	HTTP Method
Submitted Form Calloff	/api/commitment_api/submittedFormCalloff	POST
Get Limit	/api/commitment_api/get_limit	POST
Save Address	/api/commitment_api/save_address	POST
Get WO List	/api/commitment_api/get_wo_list/{id}	GET
Get Detail Calloff	/api/commitment_api/get_detail_calloff	POST

6.2.2.4 Approval

Table 27. Commitment – Approval API Endpoints

API Name	API Route	HTTP Method
Approve Commitment	/api/commitment_api/approve	POST
Reject Commitment	/api/commitment_api/reject	POST
Change Status Mod Approval	/api/commitment_api/commitment/changeStatus Mod	POST
Upload Close File	/api/commitment_api/commitment/uploadCloseFil e	POST
Close File Delete	/api/commitment_api/commitment/closeFileDelete	POST
Update Commitment Status	/api/commitment_api/commitment/status	POST
Cancel Commitment	/api/commitment_api/commitment/cancel	POST
Delete Commitment	/api/commitment_api/commitment/delete	POST

6.2.2.5 Commitment Management Routes

Table 28. Commitment Management API Endpoints

API Name	API Route	HTTP Method
Get Create Data	/api/commitment_api/create	GET
Get Edit Data	/api/commitment_api/edit/{id}	GET
Get PIC Modal	/api/commitment_api/createPicModal	GET
Get Delivery Location Modal	/api/commitment_api/createDeliveryLocationModal	GET
Get Create Item	/api/commitment_api/creatItem/{id}	GET
Save Commitment	/api/commitment_api/saveCommitment	POST
Save Address	/api/commitment_api/saveAddress	POST
Create PIC	/api/commitment_api/createPIC	POST
Save Order Commitment File	/api/commitment_api/commitment/save_file	POST
Delete Order Commitment File	/api/commitment_api/commitment/delete_file	POST
Add Customer Files	/api/commitment_api/addcustomerfiles	POST
Delete Customer Files	/api/commitment_api/deletecustomerfiles	POST
Submit Customer Files	/api/commitment_api/submitcustomerfiles	POST
Change Customer Pending	/api/commitment_api/changecustomerpending/{commitmentId}	GET
Get Attachment	/api/commitment_api/getattachment/{id}	GET

6.2.2.6 Read-Only

Table 29. Commitment – Read Only API Endpoints

API Name	API Route	HTTP Method
Get Commitment Data	/api/commitment_api/data/{type?}	GET
Get Custom Field	/api/commitment_api/getCustomField/{id}	GET
Get User	/api/commitment_api/get_user`	GET
Get Commitment By Type	/api/commitment_api/commitment/{type?}	GET
Get Detail Delivery	/api/commitment_api/detailDelivery/{customerId}	GET
Get Commitment Detail	/api/commitment_api/commitmentDetail/{commitmentId}	GET
Modal Dispatch	/api/commitment_api/modalDispatch	POST
Generate Magic Button	/api/commitment_api/generateMagicButton	POST
Get Commitment Detail Item	/api/commitment_api/commitmentDetailItem	POST
Show Activity Note	/api/commitment_api/showActivityNote/{id}	GET
Get Return Data	/api/commitment_api/getreturndata	POST
Modal Quantity Reduce Commitment	/api/commitment_api/modalQtyReduceCommitment	POST
Modal Quantity Increase Commitment	/api/commitment_api/modalQtyIncreaseCommitment	POST
Check PONO Duplicate	/api/commitment_api/checkPonoDuplicate	POST
Close Short Commitment Miti	/api/commitment_api/closeShortCommitmentMiti	POST
Save PRTE	/api/commitment_api/savePRTE	POST
Get Item PRTE	/api/commitment_api/getItemPRTE	POST

API Name	API Route	HTTP Method
Cancel PRTE	/api/commitment_api/cancelPRTE	POST
Get Modal PRTE	/api/commitment_api/getModalPRTE	POST
Get Lot Selection PRTE	/api/commitment_api/getLotSelectionPRTE	POST
Get Lot Selection PRTE For Calloff	/api/commitment_api/getLotSelectionPRTEForCalloff	POST
Save Magic Button	/api/commitment_api/saveMagicButton	POST
Get Limit PRTE	/api/commitment_api/getLimitPRTE	POST
Save Customer Address PRTE	/api/commitment_api/saveCustomerAddressPRTE	POST
Check PRTE Calloff	/api/commitment_api/checkPRTECalloff	POST
Edit PRTE Calloff	/api/commitment_api/editPRTECalloff/{id}	GET
Get Item Commitment By PONO	/api/commitment_api/getItemCommitmentByPono	POST

6.2.2.7 Shared

Table 30. Commitment – Shared API Endpoints

API Name	API Route	HTTP Method
Load Notification	/api/shared/notification/load	GET
Download File	/api/shared/file/download/{id}	GET
Download Directory	/api/shared/directory/download/{id}	GET
Switch Account	/api/shared/switch_account/{id}	GET
Switch Account Back	/api/shared/switch_account_back	GET
Record Activity Tracker	/api/shared/activity_tracker	POST

6.2.2.8 User Authentication

Table 31. Commitment – User Authentication API Endpoints

API Name	API Route	HTTP Method
Get User Info	/api/user	GET

6.2.3 Forecast

Table 32. Forecast API Endpoints

API Name	API Route	HTTP Method
Get User	/forecast_api/get_user	GET
Get Privilege	/forecast_api/get_privilege	GET
Get Privilege Well Design	/forecast_api/get_privilege_well_design	GET
Get Item Options	/forecast_api/forecast/getItemOptions	POST
Get All Customer	/forecast_api/forecast/getAllCustomer	GET
Get Customer Forecast	/forecast_api/forecast/getCustomerForecast	GET
Get Dashboard Data	/forecast_api/forecast/dashboard_data{filter?}	GET
Get Options Filter	/forecast_api/forecast/getOptionsFilter	GET
Get Sidebar Data	/forecast_api/forecast/sidebar_data{filter?}	GET
Save Forecast	/forecast_api/forecast/saveForecast	POST
Move Forecast	/forecast_api/forecast/moveForecast	POST
Duplicate Forecast	/forecast_api/forecast/duplicateForecast	POST
Delete Forecast	/forecast_api/forecast/deleteForecast	POST
Get Well	/forecast_api/forecast/getWell	GET
Clear Forecast By Well	/forecast_api/forecast/clearForecastByWell	POST
Generate Excel	/forecast_api/forecast/generateExcel{filter?}	GET
Import Forecast	/forecast_api/forecast/importForecast	POST

6.3 Procurement

6.3.1 Lot Info

6.3.1.1 Lot Info List

Table 33. Lot Info – List API Endpoints

API Name	API Route	HTTP Method
Lot Info List	/lotinfo_api/list	POST
Lot Info Export	/lotinfo_api/list/export	POST

6.3.1.2 Create Lot Info

Table 34. Lot Info – Create Lot Info

API Name	API Route	HTTP Method
List Items	/lotinfo_api/listItems	GET
Create Lot Info	/lotinfo_api/create	POST

6.3.1.3 Lot Info Detail

Table 35. Lot Info – Detail API Endpoints

API Name	API Route	HTTP Method
Lot Info Detail	/lotinfo_api/detail/{id_item}/{batch}	GET
Approve Lot Info	/lotinfo_api/approve	POST
Complete Lot Info	/lotinfo_api/complete	POST
Update Arrival Date	/lotinfo_api/updateArrivalDate	POST
Update Alias	/lotinfo_api/updateAlias	POST
Save Reject Rate	/lotinfo_api/saveRejectRate	POST

6.3.1.4 MDR

Table 36. Lot Info – MDR API Endpoints

API Name	API Route	HTTP Method
Get All MDR (List)	/lotinfo_api/getAllMdr/{idItem}/{batch}	GET
Get MDR Folder Options	/lotinfo_api/getMdrFolderOptions	GET
Download Multiple MDR	/lotinfo_api/downloadMultipleMdr	POST
MDR Upload	/lotinfo_api/uploadMdr	POST
MDR Save	/lotinfo_api/saveMdr	POST
MDR Move	/lotinfo_api/moveMdr	POST
MDR Delete	/lotinfo_api/deleteMdr	POST
Import Tally from Excel	/lotinfo_api/importTally	POST

6.3.1.5 Tally List

Table 37. Lot Info – Tally List API Endpoints

API Name	API Route	HTTP Method
List Item Tally	/lotinfo_api/itemTally	POST
Item Tally Export	/lotinfo_api/itemTally/export	POST
Tally History	/lotinfo_api/getTallyHistory/{param}	GET
Get Data From Tally	/lotinfo_api/getDataFromTally	POST
Update Item Tally	/lotinfo_api/itemTally/update	POST
Delete Tally	/lotinfo_api/itemTally/delete	POST
Delete All Tally	/lotinfo_api/itemTally/deleteAll	POST

6.3.1.6 Inspection

Table 38. Lot Info – Inspection API Endpoints

API Name	API Route	HTTP Method
Inspection Record	/lotinfo_api/getInspection	POST
Delete Inspection	/lotinfo_api/deleteInspection	POST

6.3.2 Mill Set Up

Table 39. Mill Set Up API Endpoints

API Name	API Route	HTTP Method
Display Mill Set Up	{{host}}/index.php/setup/repairmill	GET
Set Mill	{{host}}/index.php/setup/repairmill/post	POST

6.4 Inventory

6.4.1 Stock on Hand

Table 40. Stock on Hand API Endpoints

API Name	API Route	HTTP Method
Stock View Lot V2	/stock_on_hand_api/viewV2/lot/{id_item?}/{batch?}/{owned?}/{yard?}	GET
Stock View Location V2	/stock_on_hand_api/viewV2/location/{id_item?}/{year?}/{batch?}/{owned?}	GET
Stock Ledger Detail	/stock_on_hand_api/ledgerDetail	GET
Stock Ledger Options	/stock_on_hand_api/ledgerDetail/option/{filter}	GET
Stock View Process	/stock_on_hand_api/viewProcess/{base64}	GET
Stock Options	/stock_on_hand_api/getOptions/{filter?}	GET
Stock Excel	/stock_on_hand_api/generateStockOnHandExcel	GET
Stock Ledger Excel	/stock_on_hand_api/generateLedgerDetailExcel	GET

6.4.2 Transfer

6.4.2.1 Authentication

Table 41. Transfer – Authentication API Endpoints

API Name	API Route	HTTP Method
Get Privilege	/transfer_api/get_privilege	GET

6.4.2.2 Transfer Management

Table 42. Transfer – Transfer Management API Endpoints

API Name	API Route	HTTP Method
Get Transfer Detail	/transfer_api/transfer/{id}	GET
Get Transfer Data	/transfer_api/data/{type}	GET
Create New Transfer	/transfer_api/createNewTransfer	POST
Save Transfer	/transfer_api/saveTransfer	POST
Save Draft Transfer	/transfer_api/saveDraftTransfer	POST
Post Jounaled	/transfer_api/postJounaled	POST
Edit Transfer	/transfer_api/edit/{id}	GET
Close Transfer	/transfer_api/closeTransfer	POST
Close Short Transfer	/transfer_api/closeShortTransfer	POST
Populate From Open Order	/transfer_api/populate	GET
Check Trust Limit	/transfer_api/trustLimit/{destinationId}/{editedTransferId?}	POST
Document Transfer	/transfer_api/documentTransfer/{code}	POST
Get Transfer Carrier	/transfer_api/getTransferCarrier	GET
Get Transfer Sales Order Reference	/transfer_api/getTransferSalesOrderReff	GET
Get Transfer Vessel	/transfer_api/getTransferVessel	GET

API Name	API Route	HTTP Method
Save Auto Transfer	/transfer_api/saveAutoTransfer	POST
Retrieve Populated Transfer SOH	/transfer_api/retrievePopulatedTransferSoh/{populatedId}	GET

6.4.2.3 Document Generation

Table 43. Transfer – Document Generation API Endpoints

API Name	API Route	HTTP Method
Generate Transfer PDF	/transfer_api/generateTransferPdf/{id}	GET
Generate Transfer Receive PDF	/transfer_api/generateTransferReceivePdf/{id}	GET
Generate Transfer Excel	/transfer_api/generateTransferExcel/{id}	GET
Generate Transfer Excel	/transfer_api/generateTransferExcel	GET
Generate Revision PDF	/transfer_api/generateRevisionPdf/{id}	GET
Generate Truck Load Checklist PDF	/transfer_api/generateTruckLoadChecklistPdf/{id}	GET
Generate Delivery Notes PDF	/transfer_api/generateDeliveryNotesPdf/{id}	GET
Generate QSF IC PDF	/transfer_api/generateQsfIcPdf/{id}	GET
Export Transfer Documents	/transfer_api/export_transfer_documents	POST

6.4.2.4 Delivery Note

Table 44. Transfer – Delivery Note API Endpoints

API Name	API Route	HTTP Method
Get Delivery Note Detail	/transfer_api/deliveryNoteDetail/{id}	GET
Add Delivery Note	/transfer_api/addDeliveryNote/{id}	GET
Save Dispatch	/transfer_api/addDeliveryNote	POST
Receive Delivery Note	/transfer_api/receiveDeliveryNote/{id}	GET
Edit Receive Delivery Note	/transfer_api/editReceiveDeliveryNote/{dispatchId}/{receiveId}	GET
Add Receive	/transfer_api/receiveDeliveryNote	POST
Cancel Receive Delivery Note	/transfer_api/cancelReceiveDeliveryNote/{receiveId}	POST
Get Document Delivery Note	/transfer_api/getDocumentDeliveryNote/{dispatchId}/{receiveId}	GET
Submit Document Delivery Note	/transfer_api/submitDocumentDeliveryNote	POST
Save Receive QSF	/transfer_api/saveReceiveQsf	POST
Get QSFIC Form	/transfer_api/getQsficForm	GET

6.4.2.5 Utility

Table 45. Transfer – Utility API Endpoints

API Name	API Route	HTTP Method
Get Origin Location	/transfer_api/origin	GET
Get Destination Location	/transfer_api/destination/{origin}/{editedTransferId?}	GET
Get Item Stock	/transfer_api/get_item_stock/{sourceId}/{editedTransfer?}	GET

API Name	API Route	HTTP Method
Get Lot Stock	/transfer_api/get_lot_stock/{sourceId}/{itemId}/{editedTransfer?}	GET
Get All Tags From Location	/transfer_api/getAllTagFromLocation/{idLocation}/{stringResponse}	GET
Get Receiving Location	/transfer_api/getReceivingLocation/{transferId?}	GET
Populate SOH	/transfer_api/populateSoh	POST

6.4.2.6 ERP Integration

Table 46. Transfer – ERP Integration API Endpoints

API Name	API Route	HTTP Method
Get ERP Sync Status	/transfer_api/getErpSyncStatus/{id}	GET
Get ERP Failed Log	/transfer_api/getErpFailedLog/{id}	GET
Get ERP Sync Status Receive	/transfer_api/getErpSyncStatusReceive/{id}	GET
Sync ERP	/transfer_api-syncErp	POST

6.4.2.7 Tablet Operation

Table 47. Transfer – Tablet Operation API Endpoints

API Name	API Route	HTTP Method
Post Pick	/transfer_api/postPick	POST
Post Dispatch Tablet	/transfer_api/postDispatchTablet	POST
Generate Delivery Notes PDF Tablet	/transfer_api/generateDeliveryNotesPdfTablet/{token}/{id}	GET
Get Tablet QSFIC Form	/incomingServicesV4/getTabletQsficForm	GET

API Name	API Route	HTTP Method
Get Incoming Update Receive	/incomingServicesV4/get_incoming_update_receive/{token}/{idIncoming}/{idReceive}/{idItem}/{batch}/{owned}/{condition}/{idQuarantine?}/{idYard?}	GET
Post Incoming	/incomingServicesV4/post_incoming	POST
Get Incoming List	/incomingServicesV4/get_incoming_list/{token}/{idIncoming}/{idReceive}	GET
Get Incoming	/incomingServicesV4/get_incoming/{token}	GET
Post Finalize	/incomingServicesV4/post_finalize	POST
Finalize QSFIC	/incomingServicesV4/finalize_qsfic/{token}	GET

6.4.2.8 Internal Notes

Table 48. Transfer – Internal Notes API Endpoints

API Name	API Route	HTTP Method
Save Internal Notes	/transfer_api/saveInternalNotes	POST
Delete Internal Notes	/transfer_api/deleteInternalNotes	POST
Edit Internal Notes	/transfer_api/editInternalNotes	POST

6.4.2.9 Attachments

Table 49. Transfer – Attachment API Endpoints

API Name	API Route	HTTP Method
Save Close Attachment	/transfer_api/closeTransfer/addAttachments	POST
Delete Close Attachment	/transfer_api/closeTransfer/deleteAttachments	POST
Save Delivery Note Attachment	/transfer_api/deliveryNote/addAttachment/{code}	POST
Delete Delivery Note Attachment	/transfer_api/deliveryNote/deleteAttachment/{code}	POST

API Name	API Route	HTTP Method
Add Internal Attachment	/transfer_api/addInternalAttachment	POST
Delete Internal Attachment	/transfer_api/deleteInternalAttachment	POST
Get Attachment	/transfer_api/getAttachment/{base64}	GET

6.4.2.10 Others

Table 50. Transfer – Others API Endpoints

API Name	API Route	HTTP Method
List Detail Instruction	/transfer_api/list_detail_instruction/{id}	GET
Get Transfer Related To List	/transfer_api/getTransferRelatedToList/{idTransfer}	GET

6.5 Vendor Management

6.5.1 List/Index Page

Table 51. List/Index Page API Endpoints

API Name	API Route	HTTP Method
Get User Information	/api/lisi_api/get_user	GET
Get User Privilege	/api/lisi_api/get_privilege	GET
Get List of 3rd Party Instruction (Tab Open)	/api/lisi_api/data/0	GET
Get List of 3rd Party Instruction (Tab Completed)	/api/lisi_api/data/1	GET
Get List of 3rd Party Instruction (Tab Cancelled)	/api/lisi_api/data/3	GET

API Name	API Route	HTTP Method
Generate excel of 3rd Party Instruction list	/api/lisi_api/generateInstructionExcel	GET

6.5.2 Create & Edit 3rd Party Instruction

Table 52. Create & Edit 3rd Party Instruction API Endpoints

API Name	API Route	HTTP Method
Get User Information	/api/lisi_api/get_user	GET
Get User Privilege	/api/lisi_api/get_privilege	GET
Get Form data	/lisi_api/get_data_instruction_form/{id?}	POST
Get Invoice To options	/lisi_api/get_invoice_to_option	GET
Get Charge To options	/lisi_api/get_charge_to_option	GET
Get Items options	/lisi_api/get_item_option/	GET
Get Transaction info number options	/lisi_api/get_transaction_info_number/	GET
Get Currency options	/lisi_api/get_currency_option/	GET
Add Attachment (type=support_doc)	/lisi_api/add_attachment_instruction/{id}/{type}	POST
Save 3rd Party Instruction (Draft/Submit)	/lisi_api/submit_data_instruction	POST

6.5.3 Detail 3rd Party Instruction

Table 53. Detail 3rd Party Instruction

API Name	API Route	HTTP Method
Get User Information	/api/lisi_api/get_user	GET
Get User Privilege	/api/lisi_api/get_privilege	GET
Get Detail of 3rd Party Instruction	/lisi_api/instruction/{id}	GET
Get list of invoice 3rd Party Instruction	/lisi_api/get_vendor_detail/{id}	GET
Delete 3rd Party Instruction	/lisi_api/delete_instruction/{id}	POST
Add Attachment Detail Support Doc (type=detail_support_doc)	/lisi_api/add_attachment_instruction/{id}/{type}	POST
Add Attachment Invoice Doc (type=invoice_doc)	/lisi_api/add_attachment_instruction/{id}/{type}	POST
Add Attachment Invoice Support Doc (type=invoice_support_doc)	/lisi_api/add_attachment_instruction/{id}/{type}	POST
Submit Vendor Invoice	/lisi_api/submit_vendor_invoice/{id}	POST
Receive all Invoice and set 3rd Party Instruction to Complete	/lisi_api/submit_all_received/{id}	POST
Export PDF 3rd Party Instruction	/lisi_api/report/{id}	GET
Get recipients for Send Email	/lisi_api/get_send_email_info/{id}	GET

API Name	API Route	HTTP Method
Send document 3rd Party Instruction via Email	/lisi_api/send_email	POST
Get User Information	/api/lisi_api/get_user	GET
Get User Privilege	/api/lisi_api/get_privilege	GET
Get Detail of 3rd Party Instruction	/lisi_api/instruction/{id}	GET
Get list of invoice 3rd Party Instruction	/lisi_api/get_vendor_detail/{id}	GET

6.6 Global Mill Advisor

6.6.1 Mill Audit Request

6.6.1.1 List/Index Page

Table 54. Mill Audit Request – List/Index API Endpoints

API Name	API Route	HTTP Method
Get user information	/api/mill_advisor_api/millaudit/get_user	GET
Get privilege	/api/mill_advisor_api/get_privilege	GET
Get index options before creating a new Mill Audit Request	/api/mill_advisor_api/millaudit/getIndexOptions	GET
Display list of Mill Audit Request	/api/mill_advisor_api/millaudit/list/{0/1}	GET
Export Excel	/api/mill_advisor_api/millaudit/generateMillListExcel	GET

6.6.1.2 Create & Edit Mill Audit Request

Table 55. Create & Edit Mill Audit Request API Endpoints

API Name	API Route	HTTP Method
Get user information	/api/mill_advisor_api/millaudit/get_user	GET
Get privilege	/api/mill_advisor_api/get_privilege	GET
Get Mill Audit Request form	/api/mill_advisor_api/millaudit/getDataMillAudit	POST
Get options of (Stakeholder, Customer, Mill Name, Reason, Scope)	/api/mill_advisor_api/millaudit/getCreateOptions	GET
Get user info	/api/mill_advisor_api/millaudit/get_user	GET
Save Mill Audit	/api/mill_advisor_api/millaudit/saveMillAudit	GET

6.6.1.3 Detail Mill Audit Request

API Name	API Route	HTTP Method
Get user information	/api/mill_advisor_api/millaudit/get_user	GET
Get privilege	/api/mill_advisor_api/get_privilege	GET
Get detail of Mill Audit Request	/api/mill_advisor_api/millaudit/data/{id}	GET
Get list of Stakeholder	/api/mill_advisor_api/millaudit/getStandardStakeholder	GET
Global QA Reviewed & Submit	/api/mill_advisor_api/millaudit/submit_reviewed	POST
Get MISI Response/Final Response	/api/mill_advisor_api/millaudit/get_misi_response/{id}	GET
Terminate Mill Audit Request	/api/mill_advisor_api/millaudit/terminate_mill_audit/{id}	POST

API Name	API Route	HTTP Method
Submit the MISI Approval/Final Approval	/api/mill_advisor_api/millaudit/submit_misi_approval	POST
Get Auditor options	/api/mill_advisor_api/millaudit/get_assign_audit_options	GET
Global QA Assign Lead Auditor	/api/mill_advisor_api/millaudit/submit_assign_audit_team	POST
Get Audit Plan form	/api/mill_advisor_api/millaudit/get_audit_plan_for_m/{id}	GET
Upload Attachment	/api/mill_advisor_api/millaudit/upload_attachment	POST
Save Audit Plan	/api/mill_advisor_api/millaudit/submit_audit_plan	POST
Get Start Audit info	/api/mill_advisor_api/millaudit/get_start_audit_date/{id}	GET
Set the Start Audit date	/api/mill_advisor_api/millaudit/submit_audit_date	POST
Get End Audit info	/api/mill_advisor_api/millaudit/get_start_audit_date/{id}	POST
Set the End Audit date	/api/mill_advisor_api/millaudit/submit_audit_date	POST
Get Audit Result form	/api/mill_advisor_api/millaudit/get_audit_result_form/{id}	GET
Get Mill NCR form	/api/mill_advisor_api/millaudit/get_mill_ncr_form/{id}	GET
Save the Mill NCR	/api/mill_advisor_api/millncr/save	POST
Save the Audit Result	/api/mill_advisor_api/millaudit/submit_audit_result	POST
Save the Lead Auditor Recommendation	/api/mill_advisor_api/millaudit/submit_lead_auditor_recommendation	POST
Save the Final Recommendation	/api/mill_advisor_api/millaudit/submit_final_recommendation	POST

6.6.2 Mill Listing

6.6.2.1 Mill Listing APIs

Table 56. Mill Listing API Endpoints

API Name	API Route	HTTP Method
Save Mill Listing	/saveMillListing	POST
Save Mill Listing Attachment	/saveMillListingAttachment	POST
Delete Mill Listing Attachment	/deleteMillListingAttachment	POST
Mill Listing Detail	/detail/{id}	GET
Product Range	/product_range/{id}	GET
Form Options	/options_new_form	GET
Edit Mill Listing	/editMillListing/{id}	GET
Export List	/export_list	GET
Weight Options	/options_weight/{id}	GET
Bulk Upload	/bulkUpload	POST
Submit Bulk Upload	/submitBulkUpload	POST
Submit Individual	/submitIndividual	POST
Mill listing list	/list	GET

6.6.2.2 Audit Related APIs

Table 57. Mill Listing – Audit Related API Endpoints

API Name	API Route	HTTP Method
Audit Log	/audit_log/{id}	GET

6.6.2.3 MoC APIs

Table 58. Mill Listing – MoC API Endpoints

API Name	API Route	HTTP Method
MoC Log	/moc_log/{id}	GET

6.6.3 Mill NCR

6.6.3.1 List/Index Page

Table 59. Mill NCR – List/Index Page API Endpoints

API Name	API Route	HTTP Method
Display List of Mill NCR	/api/mill_advisor_api/millncr/list?status={open, closed}	GET
Export Excel	/api/mill_advisor_api/millncr/list/export	GET

6.6.3.2 Edit Mill NCR

Table 60. Edit Mill NCR API Endpoints

API Name	API Route	HTTP Method
Get user information	/api/mill_advisor_api/millaudit/get_mill_ncr_form/{mill_audit_request_id}/{mill_ncr_id}	GET

6.6.3.3 Detail Mill NCR

Table 61. Detail Mill NCR API Endpoints

API Name	API Route	HTTP Method
Get user information	/api/mill_advisor_api/millaudit/get_user	GET
Get detail of Mill NCR	/api/mill_advisor_api/millncr/detail/{id}	GET
Cancel the Mill NCR	/api/mill_advisor_api/millncr/terminate	POST
Upload Attachment	/api/mill_advisor_api/millncr/upload_attachment	POST

API Name	API Route	HTTP Method
Save response Mill NCR	/api/mill_advisor_api/millncr/response/save	POST

6.6.4 Mill MoC

6.6.4.1 List/Index Page

Table 62. Mill MoC – List/Index API Endpoints

API Name	API Route	HTTP Method
Display list of Mill MOC	/api/mill_moc_api/list?status={open, closed}	GET
Export Excel	/api/mill_moc_api/generateMillMocListExcel	GET

6.6.4.2 Create & Edit MoC

Table 63. Create & Edit Mill MoC API Endpoints

API Name	API Route	HTTP Method
Get several options like {categories, mill_audit, mills, and risk level}	/api/mill_moc_api/getCreateOptions	GET
Upload Attachment	/api/mill_moc_api/mocs/attachment	POST
Save Mill MOC	/api/mill_moc_api/mocs/save	POST

6.6.4.3 Detail Mill MoC

Table 64. Detail Mill MoC API Endpoints

API Name	API Route	HTTP Method
Get Detail of Mill MOC	/api/mill_moc_api/mocs/{id}/detail	GET
Get Risk Controls form	/api/mill_moc_api/mocs/{id}/risk_controls	GET
Get list Approvers	/api/mill_moc_api/mocs/{id}/approvers	GET

API Name	API Route	HTTP Method
Upload Attachment	/api/mill_moc_api/mocs/attachment	POST
Save the Evaluation of Risk Control	/api/mill_moc_api/mocs/{id}/risk_controls/save	POST
Get the Assign Approver form	/api/mill_moc_api/mocs/{id}/approvers/form	GET
Assign Approver	/api/mill_moc_api/mocs/{id}/approvers/add	POST
Delete Approver	/api/mill_moc_api/mocs/{id}/approvers/delete	POST
Confirm reject/approve Approver	/api/mill_moc_api/mocs/{mocId}/approvers/{approverId}/confirm	POST

6.7 Setup

6.7.1 User

Table 65. Setup User API Endpoints

API Name	API Route	HTTP Method
Open User List	/setup/admin	GET
Open User Detail	/setup/admin/detail/{id}	GET
Save User	/setup/admin/post	POST
Delete User	/setup/admin/delete	POST
Approve User in Waiting for Approval	/setup/admin/approveuser	POST
Reject User in Waiting for Approval	/setup/admin/rejectuser	POST
Resend activation email	/setup/admin/resend	POST
Get Contract	/setup/admin/get_contract/{idparent}	GET

6.7.2 Privilege

Table 66. Setup Privilege API Endpoints

API Name	API Route	HTTP Method
Open Setup Privilege	/auth/privilege	GET
Open Specific Privilege	/auth/privilege/privilege_form/{code}	GET
Save Privilege	/auth/privilege/save/{code}	POST

6.7.3 Tablet Access

6.7.3.1 List/Index Page

Table 67. Setup Tablet – List/Index Page API Endpoints

API Name	API Route	HTTP Method
Get list of Active Tablet	{{host}}/index.php/setup/tablet/data/active	GET
Get list of Archive Tablet	{{host}}/index.php/setup/tablet/data/archive	GET
Deactivate Tablet	{{host}}/index.php/setup/tablet/deactive	POST
Delete Tablet	{{host}}/index.php/setup/tablet/delete	POST

6.7.3.2 Create & Edit Tablet Setup

Table 68. Create & Edit Table Setup

API Name	API Route	HTTP Method
Create & Edit Tablet View	{{host}}//index.php/setup/tablet/edit/{id?}	GET
Submit Tablet Form	{{host}}//index.php/setup/tablet/post	POST

6.7.3.3 Detail Tablet Setup

Table 69. Detail Tablet Setup

API Name	API Route	HTTP Method
Display detail Tablet	{{host}}//index.php/setup/tablet/det/{id}	GET

6.7.4 Contractor

Table 70. Setup Contractor API Endpoints

API Name	API Route	HTTP Method
Get all contractors data	index.php/setup/contractor/data	GET
View contractor details	index.php/setup/contractor/detail/{id}	GET
Edit contractor form	index.php/setup/contractor/edit/{id}	GET
Create/Update contractor	index.php/setup/contractor/post	POST
Delete contractor	index.php/setup/contractor/delete	POST
Get list of racks for a contractor	index.php/setup/contractor/listrack/{contractor_id}	GET
Add new rack to contractor	index.php/setup/contractor/post_newrack/{contractor_id}	POST
Delete rack from contractor	index.php/setup/contractor/delete_rack/{contractor_id}	POST
Get list of contractor users	index.php/setup/contractor/get_user	GET
Get specific contractor user details	index.php/setup/contractor/get_user/{user_id}	GET
Delete contractor user	index.php/setup/contractor/delete_user	POST

API Name	API Route	HTTP Method
Get contractor documents	index.php/setup/contractor/get_doc	GET
Upload new document	index.php/setup/contractor/new_other_doc/{contractor_id}	POST
Download document	index.php/setup/contractor/download_reference_doc/{file_name}	GET
Delete document	index.php/setup/contractor/delete_reference_doc/{yard}/{doc_id}	GET
View contractor stamp	index.php/setup/contractor/showStamp/{fname}/{lname}	GET
Get contractor addresses	index.php/setup/contractor/get_list_address/{contractor_id}	GET
Save contractor address	index.php/setup/contractor/save_address	POST
Delete contractor address	index.php/setup/contractor/delete_address	POST

6.7.5 Port

Table 71. Setup Port API Endpoints

API Name	API Route	HTTP Method
Get port list	index.php/setup/port	GET
Get port list data	index.php/setup/port/data	GET
Save port	index.php/setup/port/post	POST
Create or edit port	index.php/setup/port/edit/{?id}	GET
Detail port	index.php/setup/port/detail/{id}	GET
Delete port	index.php/setup/port/delete	POST

6.7.6 Item

Table 72. Setup Item API Endpoints

API Name	API Route	HTTP Method
Type Options	/setup_api/item/typeOptions	GET
Specification Options	/setup_api/item/specificationOptions/{typeItem?}	GET
Load Data	/setup_api/item/loadData	POST
Detail Data	/setup_api/item/detailData	POST
Item List	/setup_api/item/data/{status}	GET
Activate Item	/setup_api/item/activateItem	POST
Deactivate Item	/setup_api/item/deactivateItem	POST
Submit Data	/setup_api/item/submitDataItem	POST
Export	/setup_api/item/export	GET
Get Privilege	/setup_api/item/get_privilege	GET
Upload Attachment	/setup_api/item/uploadAttachment	POST
Get Item Options	/setup_api/item/getItemOptions	GET
Get Confirmation Data	/setup_api/item/getConfirmationData	POST
Submit Assembly	/setup_api/item/submitAssemblyConfirmation	POST
Validate Assembly	/setup_api/item/submitAssemblyConfirmation/{validate}	POST
Convert D365 Code	/setup_api/item/convertD365Code	POST

6.7.7 OD

Table 73. Setup OD API Endpoints

API Name	API Route	HTTP Method
Get OD List	/setup_api/od/data	GET
Get OD Edit	/setup_api/od/edit/{id}	GET

API Name	API Route	HTTP Method
Save OD	/setup_api/od/save	POST
Delete OD	/setup_api/od/delete	POST

6.7.8 Grade

Table 74. Setup Grade API Endpoints

API Name	API Route	HTTP Method
Get Grade Options	/setup_api/grade/options	GET
Get Grade List	/setup_api/grade/data	GET
Get Grade Detail	/setup_api/grade/det/{id}	GET
Get Grade Edit	/setup_api/grade/edit/{id}	GET
Delete Grade	/setup_api/grade/delete	POST
Save Grade	/setup_api/grade/save	POST

6.7.9 Connection

Table 75. Setup Connection API Endpoints

API Name	API Route	HTTP Method
Get Connection List	/setup_api/connection/data	GET
Get Connection Detail	/setup_api/connection/det/{id}	GET
Get Connection Edit	/setup_api/connection/edit/{id}	GET
Get Connection Options	/setup_api/connection/options	GET
Save Connection	/setup_api/connection/save	POST
Delete Connection	/setup_api/connection/delete	POST

6.7.10 Special Condition

Table 76. Setup Special Condition API Endpoints

API Name	API Route	HTTP Method
Get Detail	/setup_api/spcond/data/{id}	GET
Get List	/setup_api/spcond/data	GET
Save	/setup_api/spcond/save	POST
Create	/setup_api/spcond/create	GET
Delete	/setup_api/spcond/delete	POST

6.7.11 Yard

Table 77. Setup Yard API Endpoints

API Name	API Route	HTTP Method
Get yard list data	index.php/setup/yard/data	GET
Save yard	index.php/setup/yard/post	POST
Create or edit yard	index.php/setup/yard/edit/{?id}	GET
Detail yard	index.php/setup/yard/detail/{id}	GET
Delete yard	index.php/setup/yard/delete	POST
List yard rack	index.php/setup/yard/listrack/{id_yard}	GET
Post yard rack	index.php/setup/yard/post_newrack/{id_yard}	POST
Delete yard rack	index.php/setup/yard/delete_rack/{id_yard}	POST
List yard user	index.php/setup/yard/get_user/{?id_user}	POST
Delete yard user	index.php/setup/yard/delete_user	POST

6.8 Support

6.8.1 Jira API

Table 78. Support – Jira API Endpoints

API Name	API Route	HTTP Method
Jira API - Add Temporary Attachment	/jira_api/add-temporary-attachment	POST
Jira API - Create Issue Ticket	/jira_api/create-issue-ticket	POST
Jira API - Create Comment	/jira_api/create-comment	POST
Jira API - Get List Issue Ticket	/jira_api/get-list-issue-ticket	GET
Jira API - Sync Issue Ticket	/jira_api-sync-issue-ticket	GET
Jira API - Get Last Log	/jira_api/get-last-log/{type}	GET
Jira API - Get List Comment Ticket	/jira_api/{issueKey}/get-list-comment-ticket	GET
Jira API - Download Attachment	/jira_api/download/{path}	GET
Jira API - Get Detail Issue Ticket	/jira_api/get-detail-issue-ticket/{issueKey}	GET
Jira API - Get Options List	/jira_api/get-options-list	GET
Jira API - Submit Related Issue	/jira_api/{issueKey}/submit-related-issue	POST

6.8.2 Support API

Table 79. Support API Endpoints

API Name	API Route	HTTP Method
Support API - MI Contact	/support_api/mi_contact	GET
Support API - FAQ	/support_api/faq	GET
Support API - Other Service	/support_api/other_service	GET
Support API - Submit FAQ	/support_api/submit_faq	POST
Support API - Submit Tutorial Video	/support_api/submit_tutorial_video	POST
Support API - Submit Attachment Video	/support_api/submit_attachment_video	POST
Support API - Delete FAQ	/support_api/delete_faq	POST
Support API - Delete Tutorial	/support_api/delete_tutorial	POST
Support API - Delete Tutorial Attachment	/support_api/delete_tutorial_attachment	POST
Support API - Get Privilege	/support_api/get_privilege	GET
Support API - User Options	/support_api/user_options	GET
Support API - Activity Feed	/support_api/activity_feed	GET
Support API - Download Excel	/support_api/download_excel	GET
Support API - Pop Up	/support_api/pop_up	GET

API Name	API Route	HTTP Method
Support API - Set Pop Up	/support_api/set_pop_up	POST

7. Data Flow Diagram

In Tubestream, the Data Flow Diagram (DFD) illustrates how data moves between users, services, and the database. It highlights key interactions such as creating inspections, screening inspections, reviewing inspections, and completing inspections. These services process data, update records, and trigger events, ensuring seamless inspection workflows. The diagram helps visualize dependencies, optimize system performance, and maintain efficient data handling in Tubestream.

7.1 Market Index

7.1.1 Market Index - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Market Index in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the create index process.

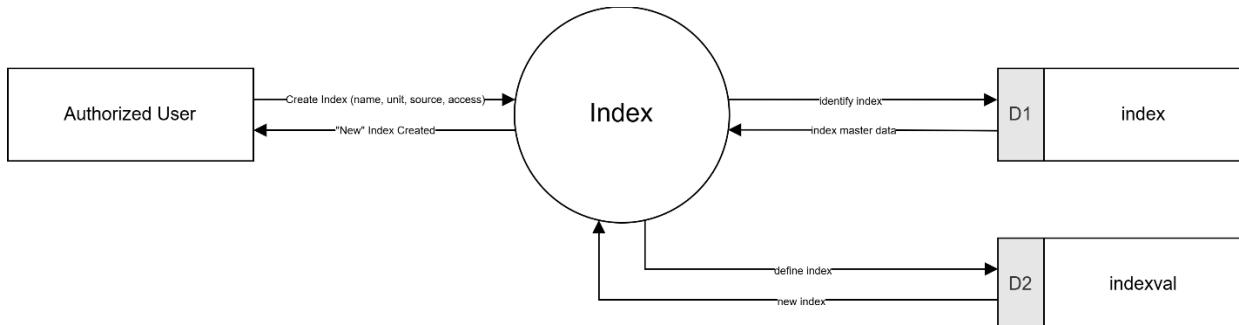


Figure 40. Market Index - Data Flow Diagram Level 0

This diagram illustrates the Indexing process in the system, showing how data flows between the authorized user and the indexing components. The process begins when the Authorized User creates an index by inputting relevant parameters such as name, unit, source, and access level. This information is then processed in the Index system, which stores the index in the index data store (D1) and the corresponding values in the indexval data store (D2). Additionally, users can perform actions such as retrieving index data and updating master data as needed.

The indexing process ensures systematic data management by capturing, processing, and storing index-related data. It enables authorized users to efficiently generate and retrieve index information, thereby supporting accurate and timely data referencing across the system.

7.2 Quality & HSE

7.2.1 SPR / NCR

7.2.1.1 NCR - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of NCR in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the NCR process.

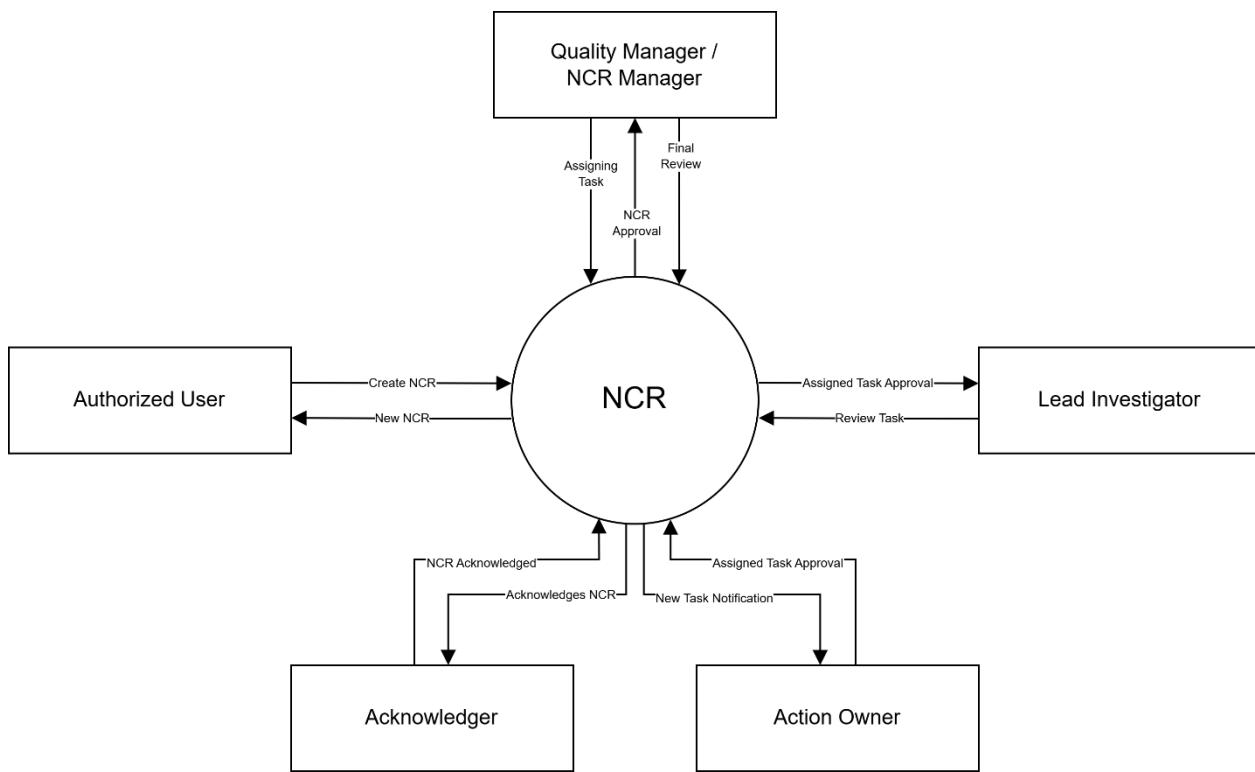


Figure 41. NCR - Data Flow Diagram Level 0

This diagram illustrates the Non-Conformance Report (NCR) process, showing how data flows between key roles and the NCR system. The process begins when the Quality & HSE User creates a new NCR. This report is then submitted to the Quality Manager / NCR Manager, who is responsible for reviewing the report, assigning the lead investigator, and approving the NCR.

The Lead Investigator is tasked with reviewing the NCR and providing their approval for task assignment. Once approved, the Action Owner is assigned specific tasks to resolve the non-conformance, and progress is tracked through task verification.

The Acknowledger also plays a role by confirming that the NCR has been acknowledged in the system.

This NCR process ensures systematic handling of non-conformities through defined responsibilities and approvals. It supports traceability and accountability by involving relevant roles in every step—from initial report creation to task completion and acknowledgment—thus maintaining quality standards and regulatory compliance.

7.2.1.2 NCR - Data Flow Diagram Level 1

This Data Flow Diagram (DFD) level 1 visualizes the NCR process in Tubestream, depicting the flow of data between users, processes, and databases. It offers a more detailed view compared to the Level 0 diagram above. The image below presents the DFD for Tubestream's NCR process.

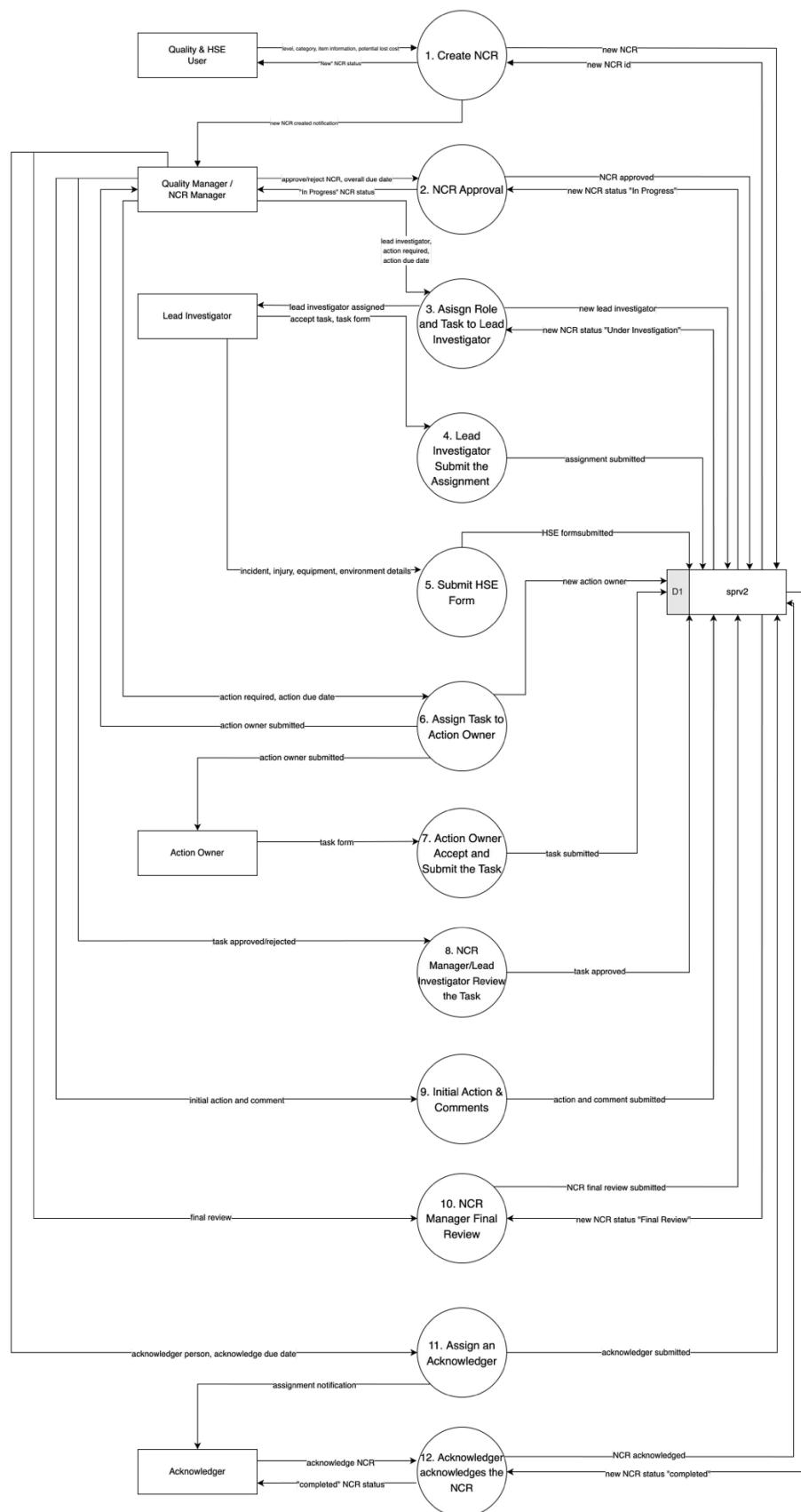


Figure 42. NCR - Data Flow Diagram Level 1

The key components are explained on the table below.

Table 80. NCR - Data Flow Diagram level 1 key components

No.	User	Input	Process	Output
1	Quality & HSE User	Create NCR	Dept, category, item info, potential lost cost	New NCR, NCR ID in sprv2 (D1)
2	Quality Manager / NCR Manager	NCR Approval	NCR submitted	NCR approved, status "In Progress"
3	Lead Investigator	Assign Role and Task to Lead Investigator	Lead investigator, task form	Assignment accepted, status "Under Investigation"
4	Lead Investigator	Submit the Assignment	Completed assignment	Assignment submitted
5	Lead Investigator	Submit HSE Form	Incident, injury, equipment, environment details	HSE form submitted in sprv2 (D1)
6	NCR Manager	Assign Task to Action Owner	Action required, due date	Task assigned
7	Action Owner	Accept and Submit Task	Task form	Task submitted
8	NCR Manager / Lead Investigator	Review Task	Task submitted	Task approved/rejected
9	NCR Manager / Lead Investigator	Initial Action & Comments	Initial action, comment	Action and comment submitted
10	NCR Manager	Final Review	Final review input	Status "Final Review"
11	NCR Manager	Assign an Acknowledger	Acknowledger name, due date	Acknowledger assigned
12	Acknowledger	Acknowledge the NCR	NCR acknowledged	NCR status "Completed" in sprv2 (D1)

7.2.2 MoC

7.2.2.1 MoC - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of MoC in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the MoC process.

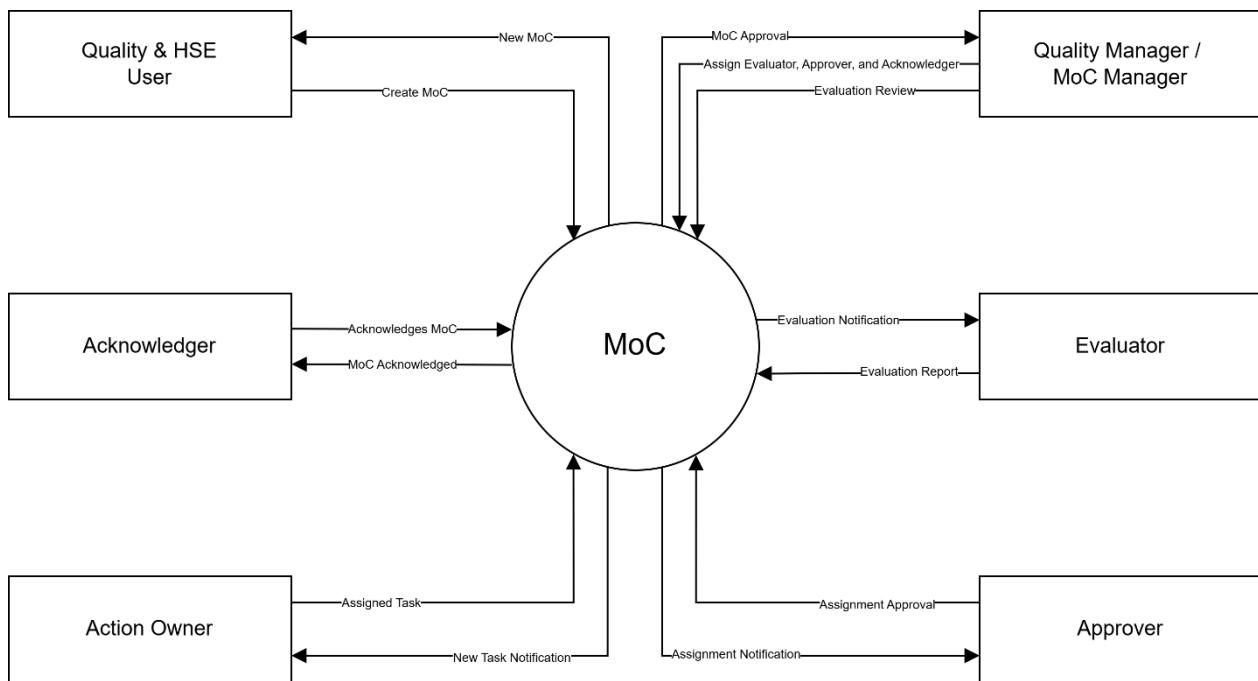


Figure 43. MoC - Data Flow Diagram Level 0

This diagram illustrates the Management of Change (MoC) process, showing the data flow between key roles and the MoC system. The process starts when the Quality & HSE User creates a new MoC entry. This submission is then reviewed by the Quality Manager / MoC Manager, who assigns the Evaluator, Approver, and Acknowledger.

The Evaluator receives a notification to review the change and submits an Evaluation Report back to the system. The Approver then reviews the evaluation and provides approval for task assignment. Once approved, an Assignment Notification is sent, and the Action Owner is assigned a specific task. A New Task Notification is generated, and the Action Owner proceeds with the task.

Simultaneously, the Acknowledger is notified and confirms acknowledgment of the MoC. Once acknowledged, a confirmation is sent back to the system.

This MoC process ensures that changes are properly assessed, approved, and documented. It provides clear traceability and accountability by involving all relevant stakeholders, thereby promoting safety, compliance, and continuous improvement across operations.

7.2.2.2 MoC - Data Flow Diagram Level 1

This Data Flow Diagram (DFD) level 1 visualizes the MoC process in Tubestream, depicting the flow of data between users, processes, and databases. It offers a more detailed view compared to the Level 0 diagram above. The image below presents the DFD for Tubestream's MoC process.

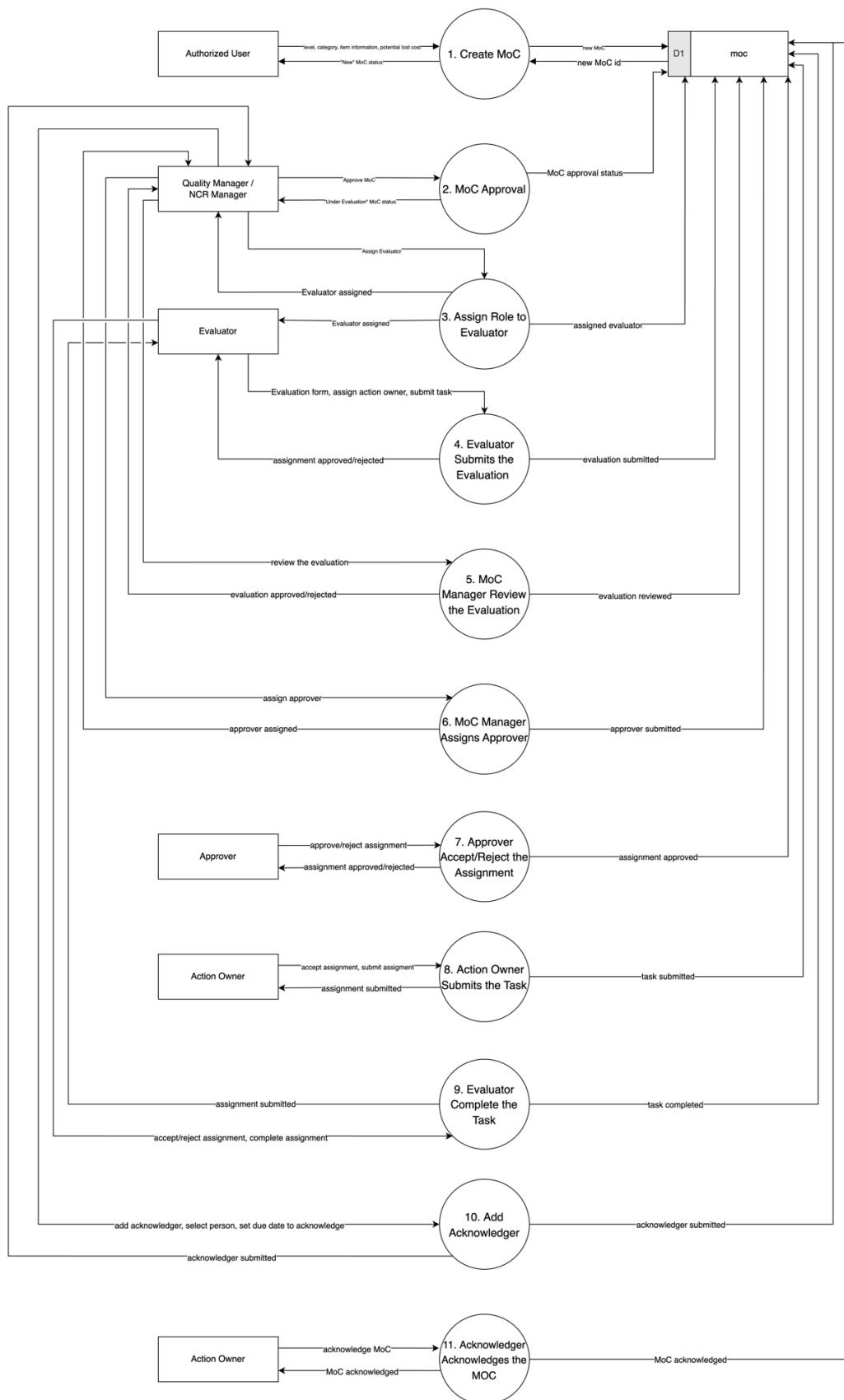


Figure 44. MoC - Data Flow Diagram Level 1

The key components are explained on the table below.

Table 81. MoC - Data Flow Diagram level 1 key components

No.	User	Input	Process	Output
1	Authorized User	Create MoC	Dept, category, item info, potential lost cost	New MoC, MoC ID in moc (D1)
2	Quality Manager / NCR Manager	MoC Approval	MoC request	Approval, status "Under Evaluation"
3	MoC Manager	Assign Role to Evaluator	Evaluator selection	Evaluator assigned
4	Evaluator	Submit the Evaluation	Evaluation form, task info	Evaluation submitted
5	MoC Manager	Review the Evaluation	Submitted evaluation	Evaluation approved/rejected
6	MoC Manager	Assign Approver	Approver info	Approver assigned
7	Approver	Approver Accept/Reject Assignment	Assignment	Assignment approved/rejected
8	Action Owner	Action Owner Submits the Task	Task form	Task submitted
9	Evaluator	Evaluator Complete the Task	Review of task	Task completed
10	MoC Manager / Evaluator	Add Acknowledger	Acknowledger, due date	Acknowledger submitted
11	Action Owner	Acknowledger Acknowledges the MoC	MoC acknowledgment	MoC status "Acknowledged" in moc D1)

7.2.3 HSE

7.2.3.1 HSE - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of HSE in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the HSE process.

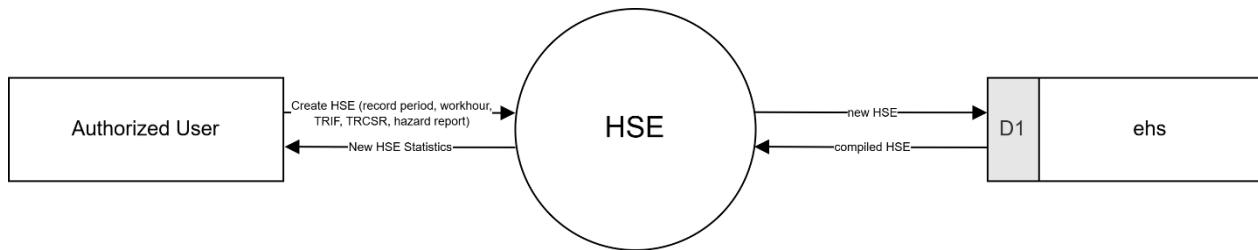


Figure 45. HSE - Data Flow Diagram Level 0

This diagram illustrates the HSE (Health, Safety, and Environment) process in the system, outlining the data flow between the Authorized User and the HSE module. The process begins when an Authorized User inputs HSE-related data, including the record period, work hours, TRIF (Total Recordable Injury Frequency), TRCSR (Total Recordable Case Severity Rate), and hazard reports. This information is processed in the HSE system, which then stores new entries in the ehs data store (D1).

The system also compiles HSE data, which is then made available to the user in the form of updated HSE statistics. This process supports accurate tracking of safety performance indicators and ensures timely updates to HSE records, facilitating compliance, safety analysis, and continuous improvement within the organization.

7.2.4 IVMS

7.2.4.1 IVMS - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of IVMS in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the IVMS process.

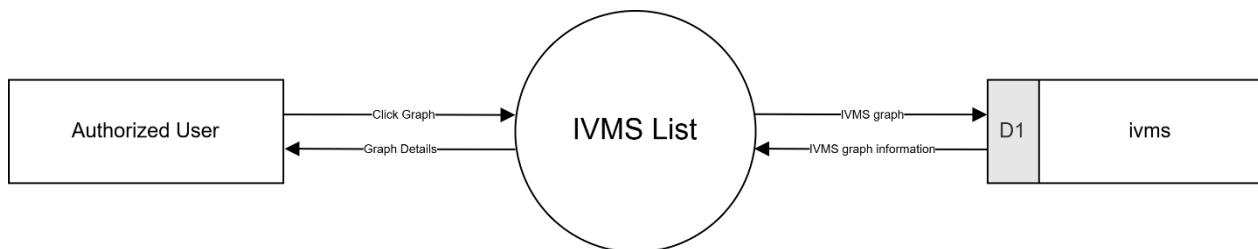


Figure 46. IVMS - Data Flow Diagram Level 0

This diagram illustrates the IVMS List process in the system. The process begins when an Authorized User interacts with the system by clicking on a graph to view specific data. The IVMS List module retrieves the corresponding graph information from the ivms data store (D1) and presents the details to the user.

This functionality enables users to visualize and analyze vehicle monitoring data efficiently. The system ensures accurate data retrieval and display, helping users track vehicle performance, incidents, or behavioral trends through interactive graph representations.

7.2.5 Lesson Learned

7.2.5.1 Lesson Learned - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Lesson Learned in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the lesson learned process.

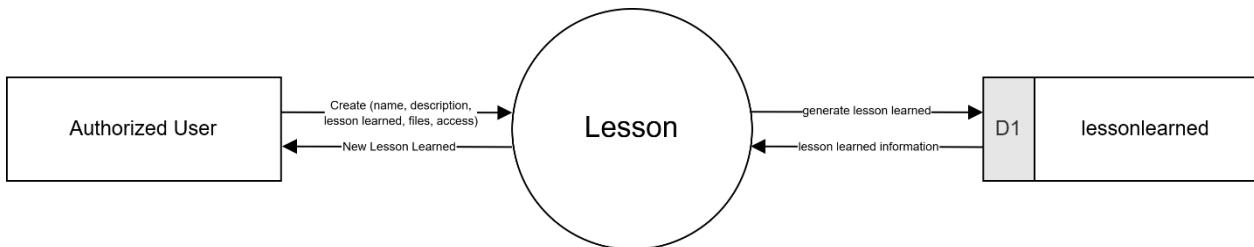


Figure 47. Lesson Learned - Data Flow Diagram Level 0

This diagram represents the Lesson Learned process, which captures and stores organizational insights. An Authorized User creates a new lesson entry by providing a name, description, lesson learned content, attached files, and access parameters. The system processes this data in the Lesson module and stores the information in the lessonlearned data store (D1).

Once stored, the system can generate and return compiled lesson information to the user, making it available for future reference. This process supports organizational learning by ensuring critical knowledge and past experiences are documented, shared, and accessible to improve future performance and decision-making.

7.2.6 SDS

7.2.6.1 SDS - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of SDS in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the SDS process.

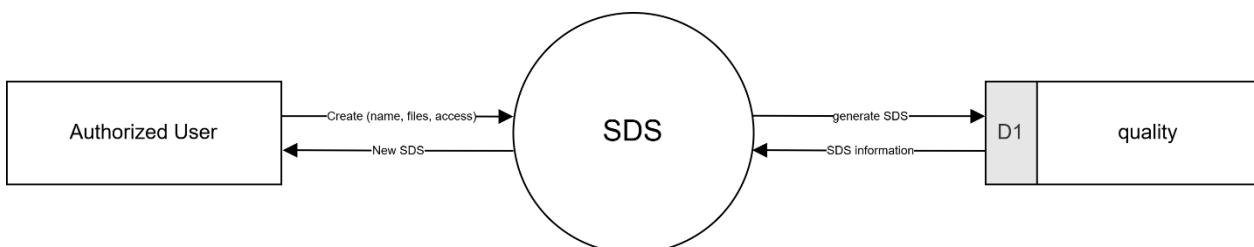


Figure 48. SDS - Data Flow Diagram Level 0

This diagram shows the SDS (Safety Data Sheet) process within the system. The Authorized User initiates the process by creating an SDS entry that includes a name, file attachments, and access settings. The SDS module handles this input and stores the generated SDS in the quality data store (D1).

The system then retrieves and compiles SDS information, returning the newly created SDS back to the user. This ensures that safety documentation is properly organized and accessible, contributing to regulatory compliance, workplace safety, and effective quality management.

7.2.7 Audit > Schedule

7.2.7.1 Audit > Schedule - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Audit Schedule in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the audit schedule process.

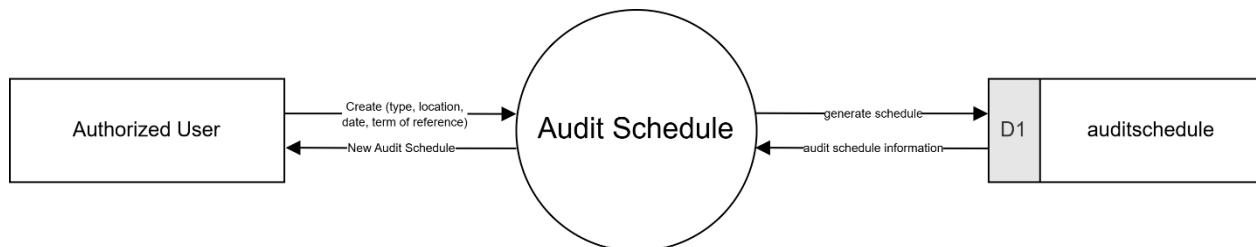


Figure 49. Audit > Schedule - Data Flow Diagram Level 0

This diagram represents the Audit Schedule process in the system. An Authorized User initiates the process by creating an audit schedule with details such as audit type, location, date, and the term of reference. The Audit Schedule module processes this input and stores the generated schedule in the auditschedule data store (D1).

The system retrieves and compiles audit schedule information and returns it to the user, ensuring that all upcoming audits are properly documented and accessible. This process helps organizations manage audit planning effectively, ensuring clarity, compliance, and timely execution of audit activities.

7.2.8 Audit > Audit Report

7.2.8.1 Audit > Audit Report - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Audit Report in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the audit report process.



Figure 50. Audit > Audit Report - Data Flow Diagram Level 0

This diagram illustrates the Audit Report generation process. The Authorized User creates a report by submitting key audit data such as type, location, date, auditor name, audit summary, score, and full audit report. The Audit Report module processes this input and saves the generated report to the quality data store (D1).

Once stored, the system provides the user with the compiled audit report. This function enables thorough documentation and tracking of audit outcomes, promoting transparency, performance assessment, and corrective action planning within the organization.

7.2.9 Audit > TPI Qualification

7.2.9.1 TPI Qualification - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of TPI Qualification in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the TPI qualification process.

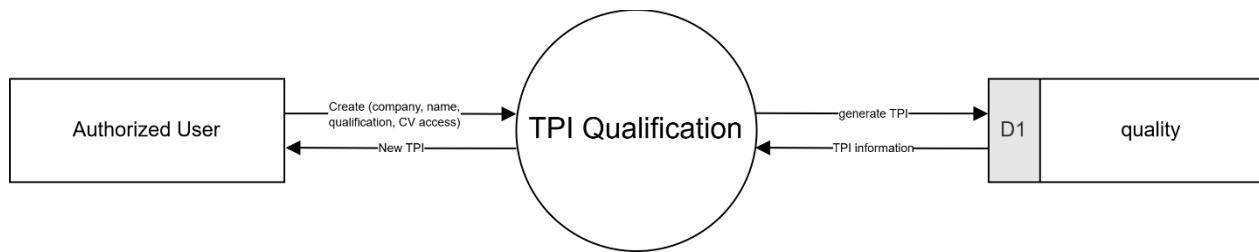


Figure 51. TPI Qualification - Data Flow Diagram Level 0

This diagram shows the process of managing TPI (Third Party Inspector) Qualification within the system. An Authorized User creates a new TPI entry by providing data such as company name, inspector name, qualifications, and CV access. The TPI Qualification module processes and stores this information in the quality data store (D1).

The system retrieves and compiles TPI qualification details and returns them to the user for further use or verification. This process ensures that inspector qualifications are recorded, verifiable, and accessible, helping maintain quality standards and regulatory compliance with inspection operations.

7.2.10 Inspection > Inspection Report

7.2.10.1 Inspection > Inspection Report - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of On Spot Inspection in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the inspection report process.

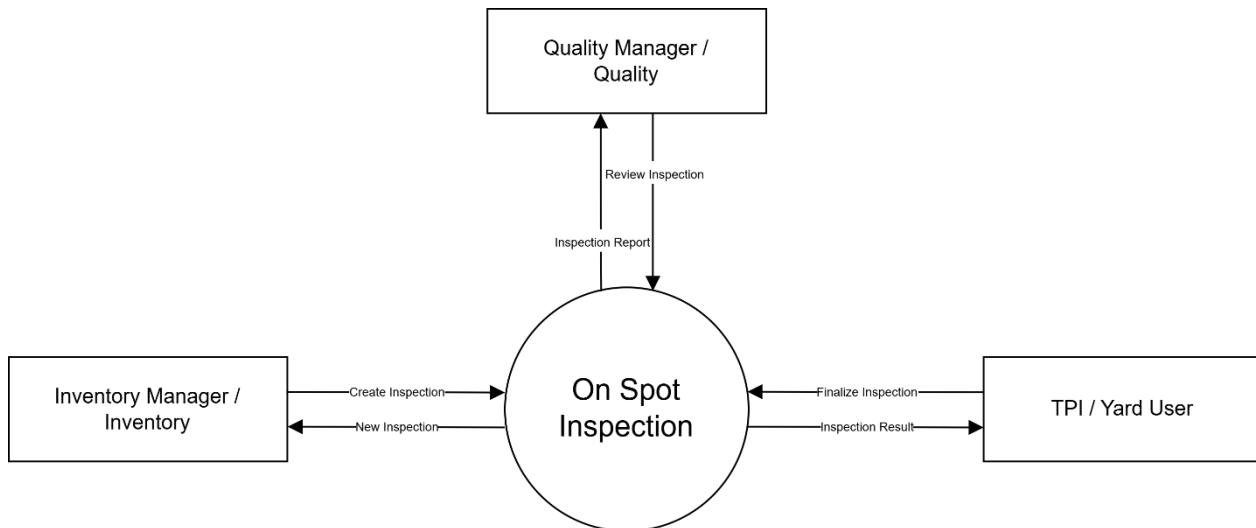


Figure 52. Inspection > Inspection Report - Data Flow Diagram Level 0

This diagram illustrates the On Spot Inspection process in Tubestream, showing how data flows between key roles and the system. The Inventory Manager initiates the process by creating a new inspection request, which is then processed in the On Spot Inspection system. The TPI/Yard User finalizes the inspection and submits the results, while the Quality Manager reviews the inspection and provides a report.

The On Spot Inspection process ensures proper tracking and validation by receiving, processing, and updating inspection data. It interacts with all key stakeholders, ensuring that inspection requests are handled systematically, results are reviewed, and necessary updates are communicated to maintain accuracy and compliance.

7.2.10.2 Inspection > Inspection Report - Data Flow Diagram Level 1

This Data Flow Diagram (DFD) level 1 visualizes the inspection report process in Tubestream, depicting the flow of data between users, processes, and databases. It offers a more detailed view compared to the Level 0 diagram above. The image below presents the DFD for Tubestream's inspection process.

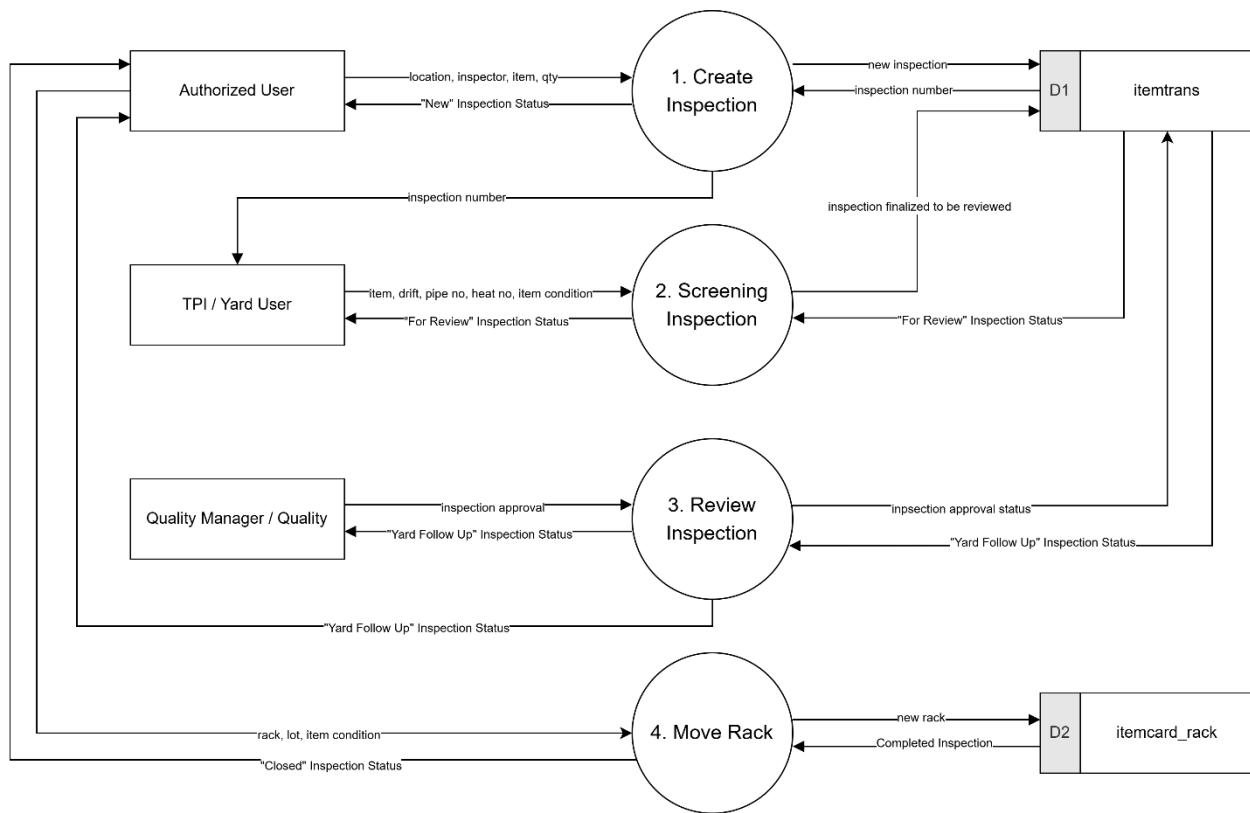


Figure 53. Inspection > Inspection Report - Data Flow Diagram Level 1

The key components are explained on the table below.

Table 82. Inspection > Inspection Report Data Flow Diagram level 1 key components

No.	User	Input	Process	Output
1	Inventory Manager	Location, Inspector, Item, Quantity	Create inspection: Creates a new inspection with status "New"	New inspection record stored in D1 (itemtrans) table
2	TPI/Yard User	Inspection number, item details (drift, pipe number, heat number, condition)	Screening inspection: Performs item screening, updates status to "For Review"	Inspection finalized for review; data stored in D1 (itemtrans) table
3	Quality Manager/Quality	Inspection approval decision	Review inspection: Approves inspection and updates status to "Yard Follow Up"	Final inspection approval status stored in D1

No.	User	Input	Process	Output
				(itemtrans) table
4	Yard User	Rack, lot number, item condition	Move rack: Moves inspected items to a new rack, updates status to "Closed"	New rack information stored in D2 (itemcard_rack) table

7.2.11 Inspection > Quarantine Lot

7.2.11.1 Inspection > Quarantine Lot - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Quarantine Lot in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the quarantine lot process.

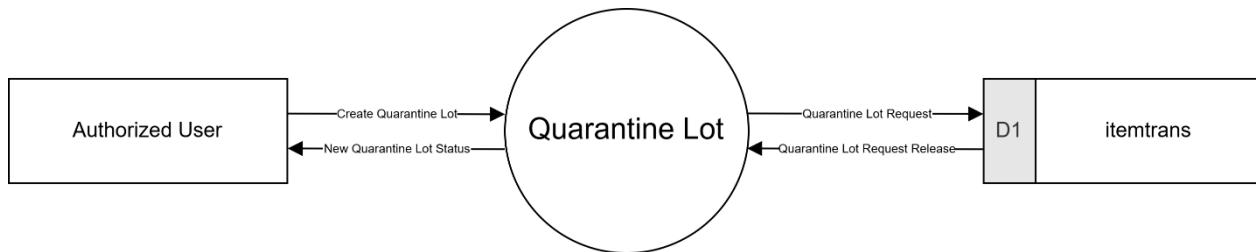


Figure 54. Inspection > Quarantine Lot - Data Flow Diagram Level 0

This diagram outlines the Quarantine Lot process within the system. An Authorized User initiates the process by creating a quarantine lot, which is handled by the Quarantine Lot module. The system then sends a Quarantine Lot Request to the itemtrans data store (D1), storing the relevant data.

Once processed, the system returns the updated Quarantine Lot Request Release and provides the New Quarantine Lot Status to the user. This process is crucial for managing items that need to be isolated for quality, compliance, or safety reasons, ensuring traceability and controlled handling before release.

7.2.11.2 Inspection > Quarantine Lot - Data Flow Diagram Level 1

This Data Flow Diagram (DFD) level 1 visualizes the quarantine lot process in Tubestream, depicting the flow of data between users, processes, and databases. It offers a more detailed view compared to the Level 0 diagram above. The image below presents the DFD for Tubestream's quarantine lot process.

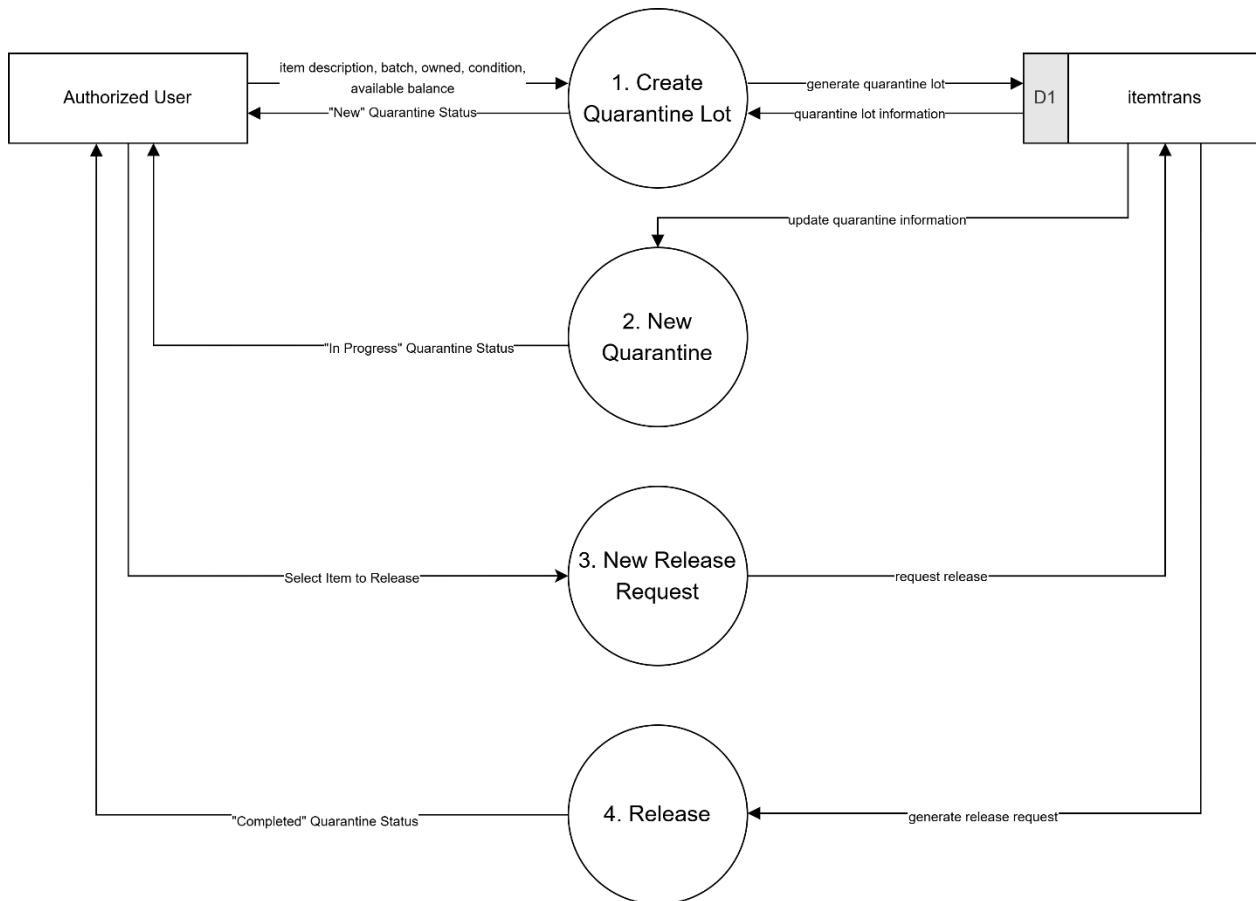


Figure 55. Inspection > Quarantine Lot - Data Flow Diagram Level 1

The key components are explained on the table below.

Table 83. Inspection > Quarantine Lot - Data Flow Diagram level 1 key components

No.	User	Input	Process	Output
1	Authorized User	Description, batch, serial, condition, inspection status	Create a new quarantine lot with status “New”	New quarantine lot record stored in D1 (itemtrans)
2	Authorized User	Quarantine lot information	Initiate quarantine process for selected lot	Updated quarantine record with status “Quarantined”
3	Authorized User	Quarantine lot and release request	Submit release request	New release request with status “Requested”

No.	User	Input	Process	Output
4	Quality Manager	Release request with lot information	Approve release and update lot status	Updated lot status to "Released" and final information sent to D1

7.2.12 Inspection > Template SoW

7.2.12.1 Inspection > Template SoW - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Template SoW in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the template SoW process.

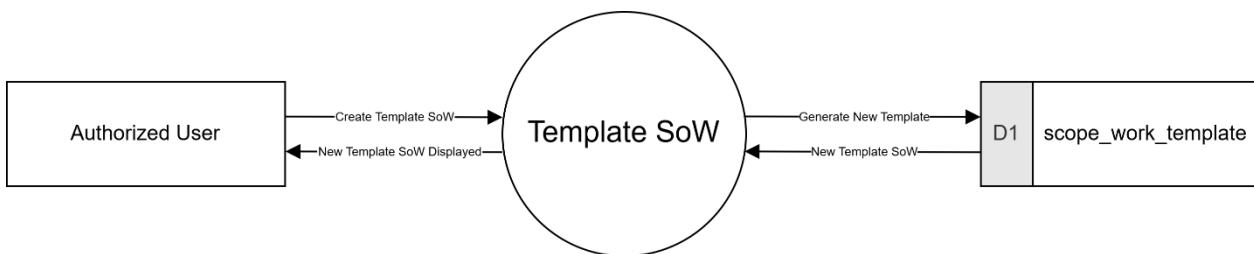


Figure 56. Inspection > Template SoW - Data Flow Diagram Level 0

This diagram illustrates the process for managing Template Scope of Work (SoW). An Authorized User creates a new template by entering SoW details, which are processed by the Template SoW module. The system generates and stores the new template in the scope_work_template data store (D1).

After storing, the system retrieves the New Template SoW and displays it back to the user. This function helps standardize the creation of scopes of work, ensuring consistency, completeness, and efficiency in planning and project documentation.

7.3 Commercial

7.3.1 Contract

7.3.1.1 Contract - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Contract in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the contract process.

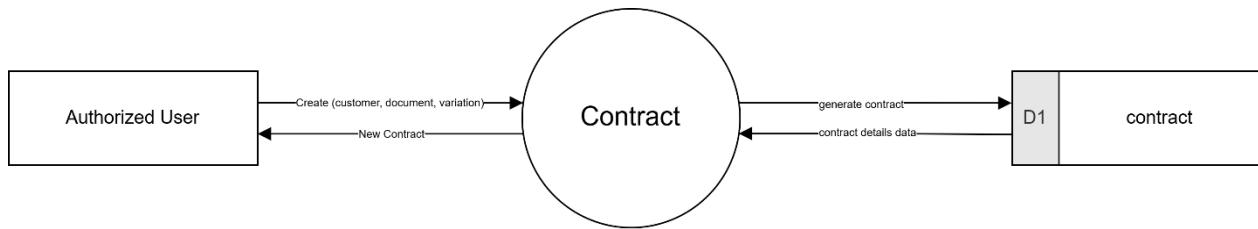


Figure 57. Contract - Data Flow Diagram Level 0

This diagram illustrates the Contract Generation process, detailing the interaction between the Authorized User and the system. The process begins when the Authorized User creates a contract by providing input data such as customer details, document type, and variation parameters. The system then generates the contract and stores it in the contract data store (D1). The contract details are retrieved and returned to the user as a new contract record.

The contract process ensures the systematic handling of contractual information by receiving, generating, and storing contract data. It provides Authorized Users with a seamless interface to create and manage contracts while ensuring data is securely stored for retrieval and reference.

7.3.2 Account Balance

7.3.2.1 Account Balance - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Account Balance in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the account balance process.

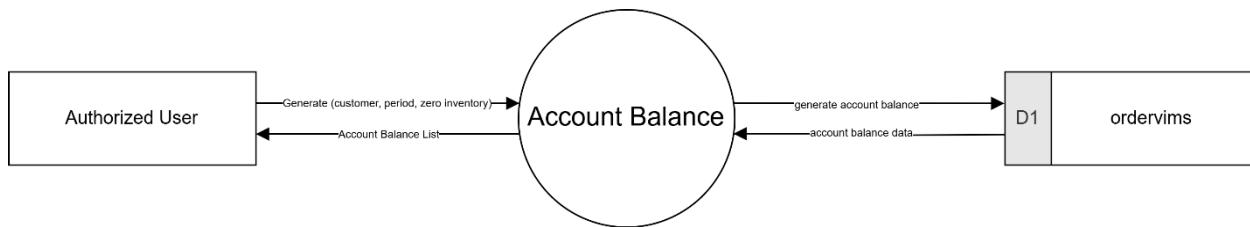


Figure 58. Account Balance - Data Flow Diagram Level 0

This diagram describes the Account Balance Generation process, which manages data flow between the Authorized User and the account balance system. The user initiates the process by submitting inputs such as customer identity, period, and a zero-inventory flag. The system processes this information, generates the account balance, and stores the result in the ordervims data store (D1). The generated balance data is then returned to the user as an account balance list.

The account balance process ensures accurate financial visibility by generating customer account data based on defined criteria. It supports operational decision-making and enables consistent tracking of inventory and financial standings.

7.3.3 Commitment

7.3.3.1 Commitment - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Commitment in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the commitment process.

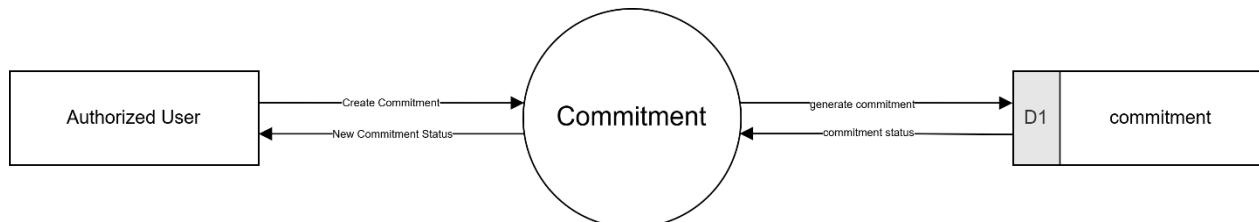


Figure 59. Commitment - Data Flow Diagram Level 0

This diagram represents the Commitment process within the system. An Authorized User initiates commitment by submitting necessary data through a creation request. The system processes this data to generate a new commitment and stores it in the commitment data store (D1). The updated status of the commitment is then sent back to the user as confirmation.

The commitment process facilitates reliable tracking of obligation records by allowing users to create, manage, and monitor commitment statuses. It ensures that commitment data is generated and maintained with consistency, providing up-to-date information for operational and contractual management.

7.3.4 Forecast

7.3.4.1 Forecast - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Forecast in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the forecast process.

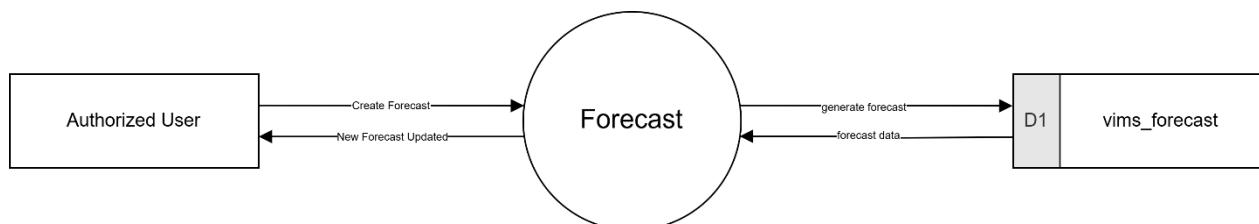


Figure 60. Forecast - Data Flow Diagram Level 0

This diagram illustrates the Forecast Generation process that supports planning and decision-making functions. The process starts when an Authorized User submits a request to create a forecast. The system processes the input and stores the forecast data in the vims_forecast data store (D1). Updated forecast information is then provided back to the user.

The forecast process plays a key role in forward planning by enabling users to input and retrieve forecast data efficiently. It ensures data consistency and supports forecasting accuracy for future inventory, sales, or operational needs.

7.3.5 Material Requisition

7.3.5.1 Material Requisition - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Material Requisition in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the material requisition process.

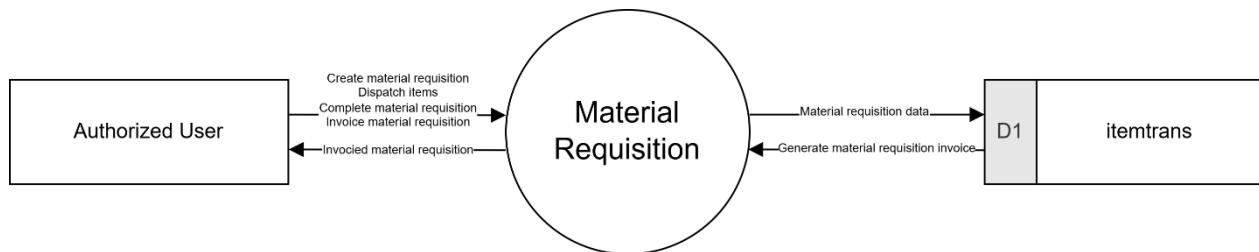


Figure 61. Material Requisition - Data Flow Diagram Level 0

This diagram represents the material requisition process. An authorized user initiates various actions, including creating a requisition, dispatching items, and completing and invoicing the requisition. The system records this data in the itemtrans data store and generates the material requisition invoice, which is returned to the user. This process facilitates accurate tracking, fulfillment, and invoicing of material requests in the supply chain.

7.3.5.2 Material Requisition - Data Flow Diagram Level 1

This Data Flow Diagram (DFD) level 1 visualizes the material requisition process in Tubestream, depicting the flow of data between users, processes, and databases. It offers a more detailed view compared to the Level 0 diagram above. The image below presents the DFD for Tubestream's material requisition process.

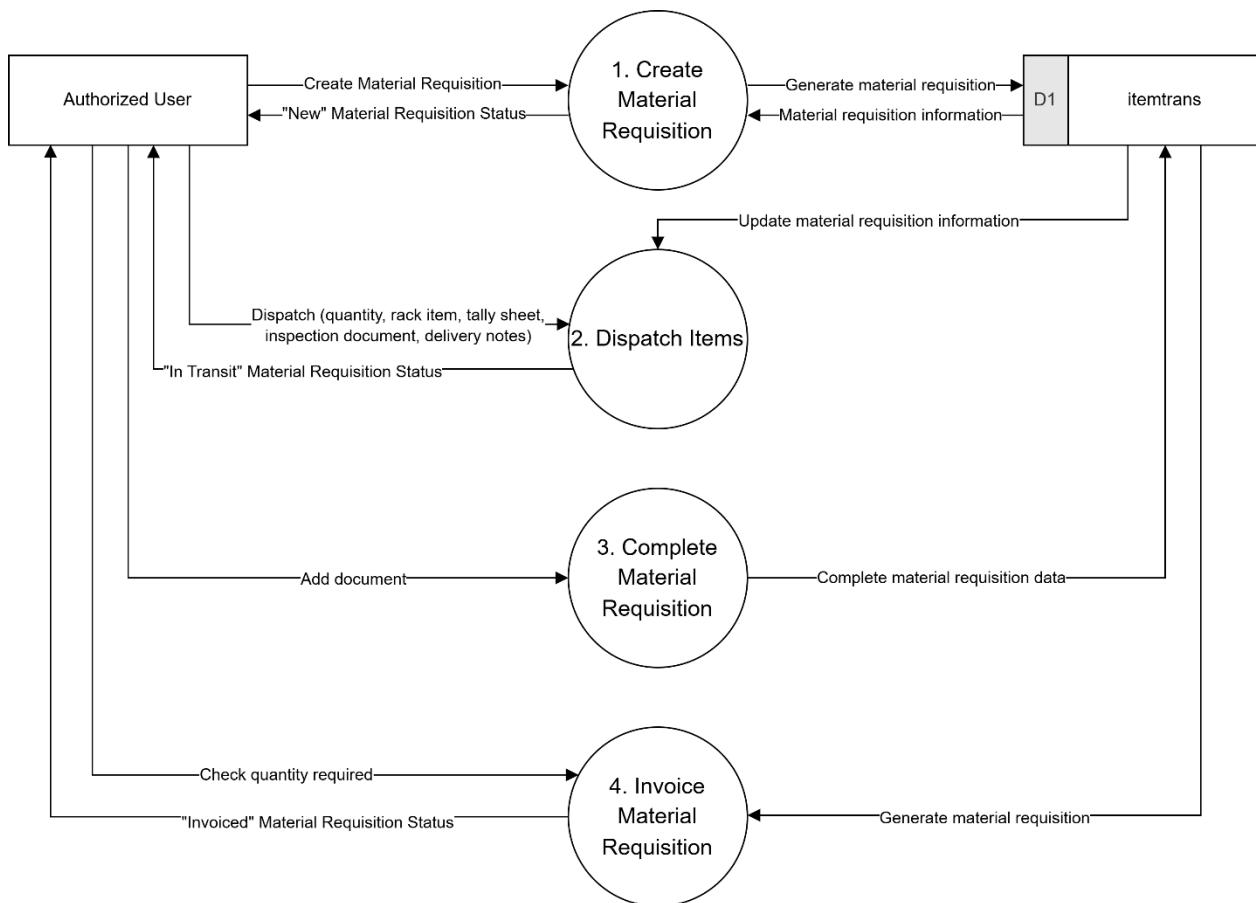


Figure 62. Material Requisition - Data Flow Diagram Level 1

The key components are explained on the table below.

Table 84. Material Requisition - Data Flow Diagram level 1 key components

No.	User	Input	Process	Output
1	Authorized User	Material requisition details	Create material requisition with status "New"	Material requisition stored in D1 (itemtrans)
2	Authorized User	Dispatch details (quantity, rack item, tally sheet, etc.)	Update requisition with status "In Transit"	Updated requisition status in itemtrans
3	Authorized User	Additional document	Complete material requisition	Material requisition marked as "Completed" in itemtrans

No.	User	Input	Process	Output
4	Authorized User	Quantity check	Generate invoice requisition	Material requisition status updated to "Invoiced" in itemtrans

7.3.6 Return

7.3.6.1 Return - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Return in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the return process.

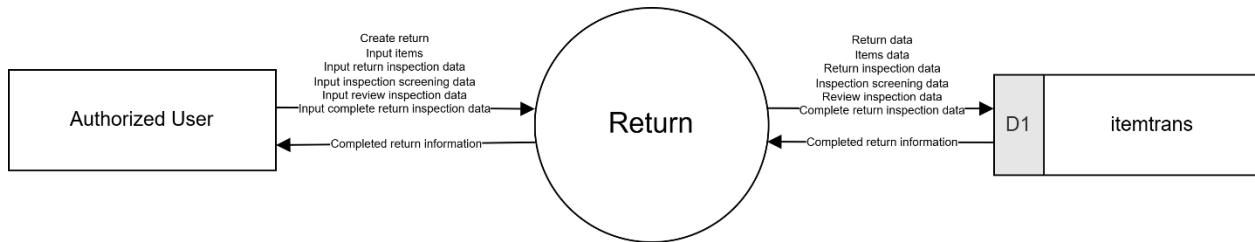


Figure 63. Return - Data Flow Diagram Level 0

The return process begins when an authorized user creates a return request and inputs various inspection data. The system processes return, item, and inspection details, updating them in the itemtrans data store. Completed return information is then provided back to the user. This process ensures detailed handling of returned items and associated inspections to maintain product quality and inventory integrity.

7.3.6.2 Return - Data Flow Diagram Level 1

This Data Flow Diagram (DFD) level 1 visualizes the create return process in Tubestream, depicting the flow of data between users, processes, and databases. It offers a more detailed view compared to the Level 0 diagram above. The image below presents the DFD for Tubestream's return process.

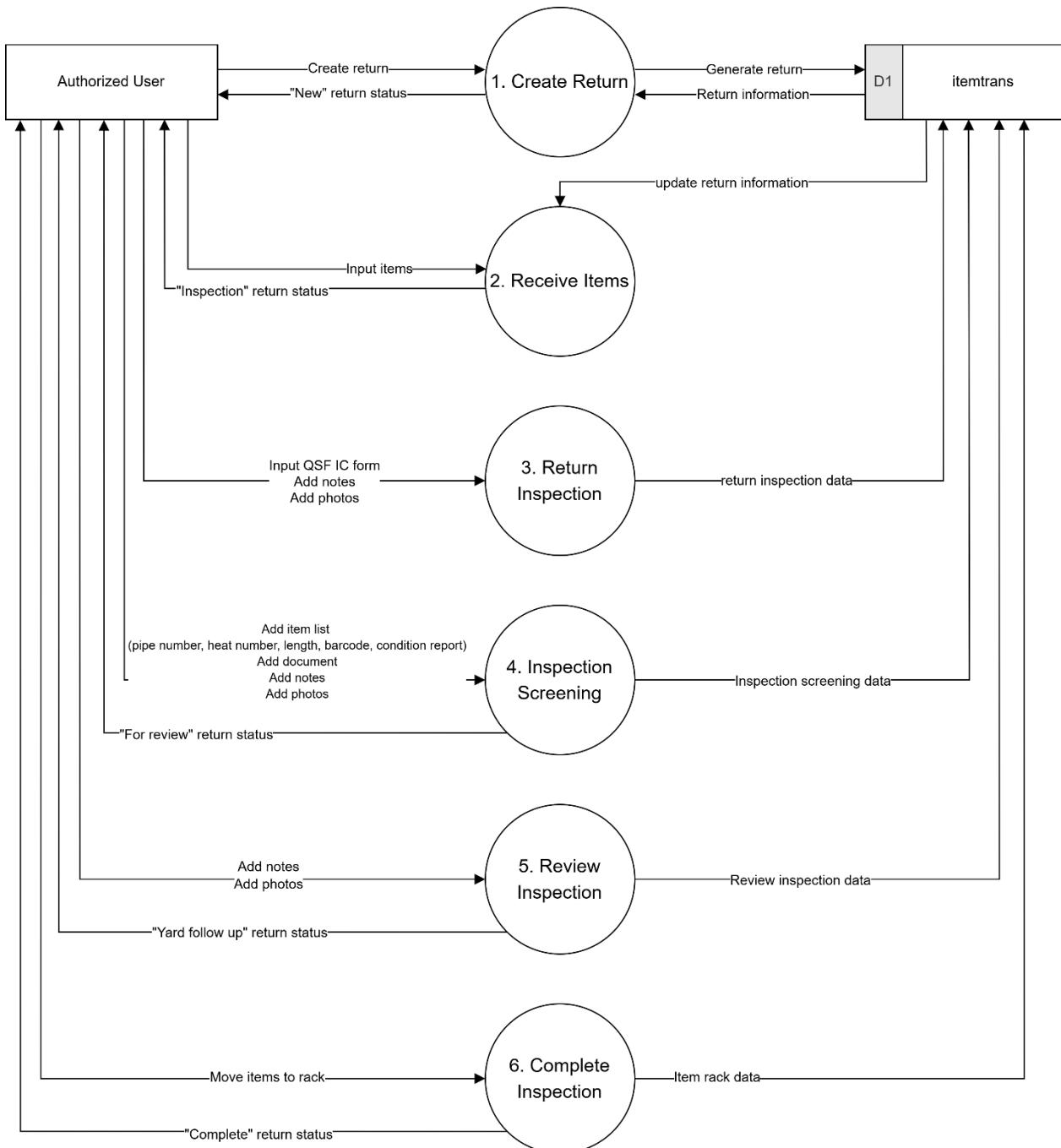


Figure 64. Return - Data Flow Diagram Level 1

The key components are explained on the table below.

Table 85. Return - Data Flow Diagram level 1 key components

No.	User	Input	Process	Output
1	Authorized User	Return details	Create return with status “New”	Return record stored in D1 (itemtrans)

No.	User	Input	Process	Output
2	Authorized User	Returned items	Update return with status “Inspection”	Updated return data in itemtrans
3	Authorized User	QSF IC form, notes, photos	Conduct return inspection	Return inspection data saved in itemtrans
4	Authorized User	Item list, documents, notes, photos	Perform inspection screening	Inspection screening data stored in itemtrans
5	Authorized User	Notes, photos	Review inspection	Return updated with status “Yard follow up” in itemtrans
6	Authorized User	Move items to rack	Complete inspection	Return marked “Complete” with rack data in itemtrans

7.3.7 Ownership Transfer

7.3.7.1 Ownership Transfer - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Ownership Transfer in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the ownership transfer process.

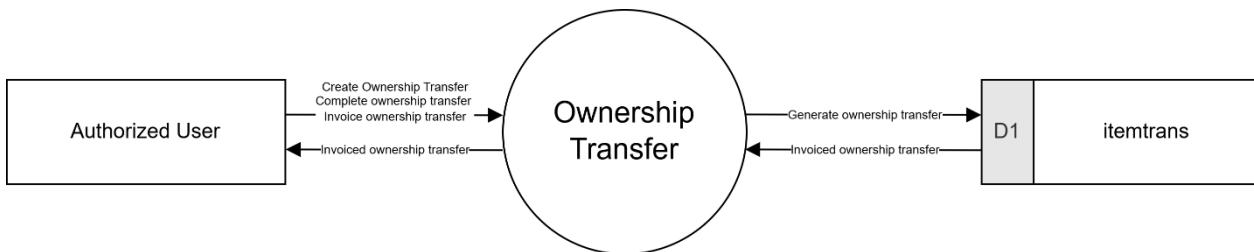


Figure 65. Ownership Transfer - Data Flow Diagram Level 0

This diagram outlines the ownership transfer process. The authorized user initiates the process by creating and completing a transfer, including invoicing details. The system generates the ownership transfer data and updates it in the itemtrans data store. Invoiced ownership transfer details are returned to the user. This process supports accurate tracking and documentation of ownership changes within the system.

7.3.7.2 Ownership Transfer - Data Flow Diagram Level 1

This Data Flow Diagram (DFD) level 1 visualizes the ownership transfer process in Tubestream, depicting the flow of data between users, processes, and databases. It offers a more detailed view compared to the Level 0 diagram above. The image below presents the DFD for Tubestream's ownership transfer process.

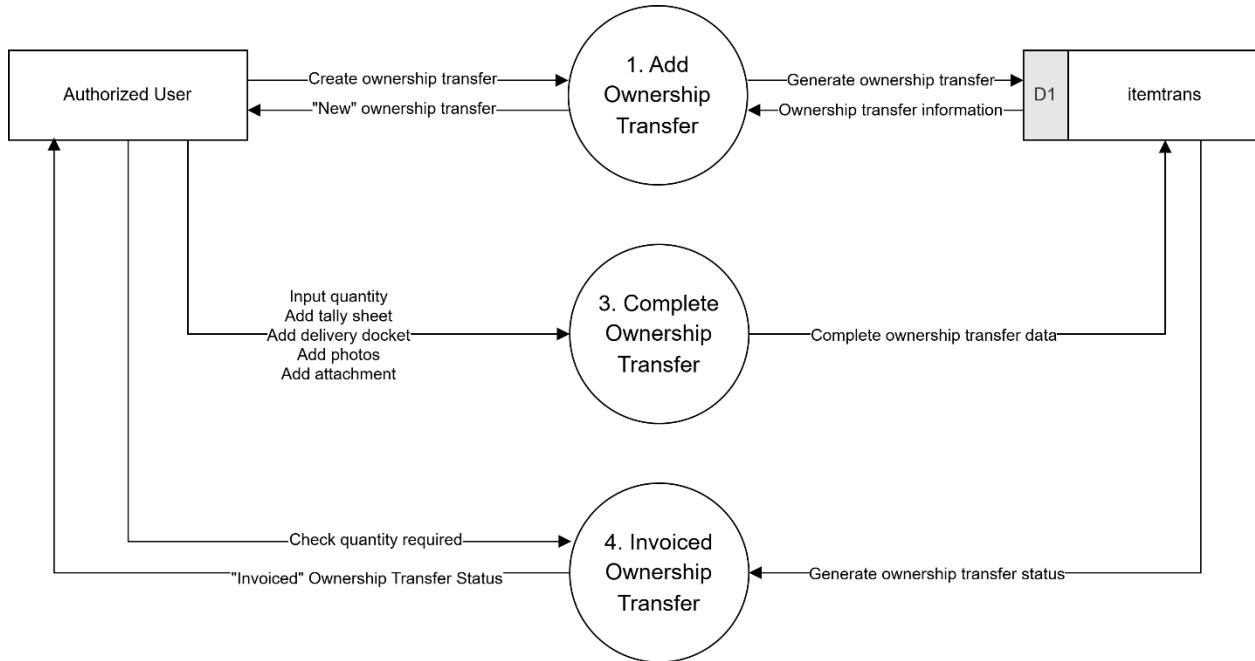


Figure 66. Ownership Transfer - Data Flow Diagram Level 1

The key components are explained on the table below.

Table 86. Ownership Transfer - Data Flow Diagram level 1 key components

No.	User	Input	Process	Output
1	Authorized User	Ownership transfer details	Create ownership transfer with status “New”	Ownership transfer record saved in D1 (itemtrans)
2	Authorized User	Quantity, tally sheet, delivery docket, attachments	Complete ownership transfer	Completed data stored in itemtrans
3	Authorized User	Quantity check	Generate invoiced status	Ownership transfer status updated to “Invoiced” in itemtrans

7.4 Procurement

7.4.1 Lot Info

7.4.1.1 Lot Info - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Lot Info in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the lot info process.

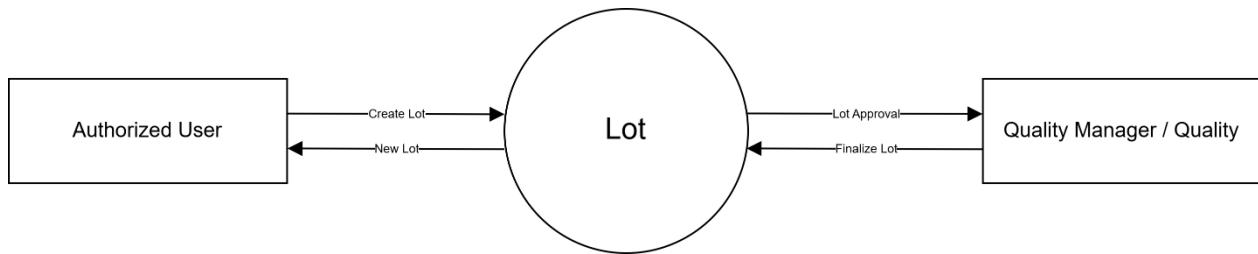


Figure 67. Lot Info - Data Flow Diagram Level 0

This diagram illustrates the Lot creation and approval process in Tubestream, outlining the data interactions between users and the system. The process begins when an Authorized User initiates the creation of a new Lot. The system processes this request and generates the Lot data. Once created, the Lot is submitted to the Quality Manager for approval. The Quality Manager then reviews the Lot and either approves or finalizes it, sending the finalized Lot data back to the system, which is accessible by the Authorized User.

The Lot management process ensures accurate tracking and validation of production or inventory items by enabling structured creation, review, and approval workflows. It facilitates collaboration between users and quality management to maintain standardization, traceability, and compliance within the system.

7.4.1.2 Lot Info - Data Flow Diagram Level 1

This Data Flow Diagram (DFD) level 1 visualizes the lot process in Tubestream, depicting the flow of data between users, processes, and databases. It offers a more detailed view compared to the Level 0 diagram above. The image below presents the DFD for Tubestream's lot info process.

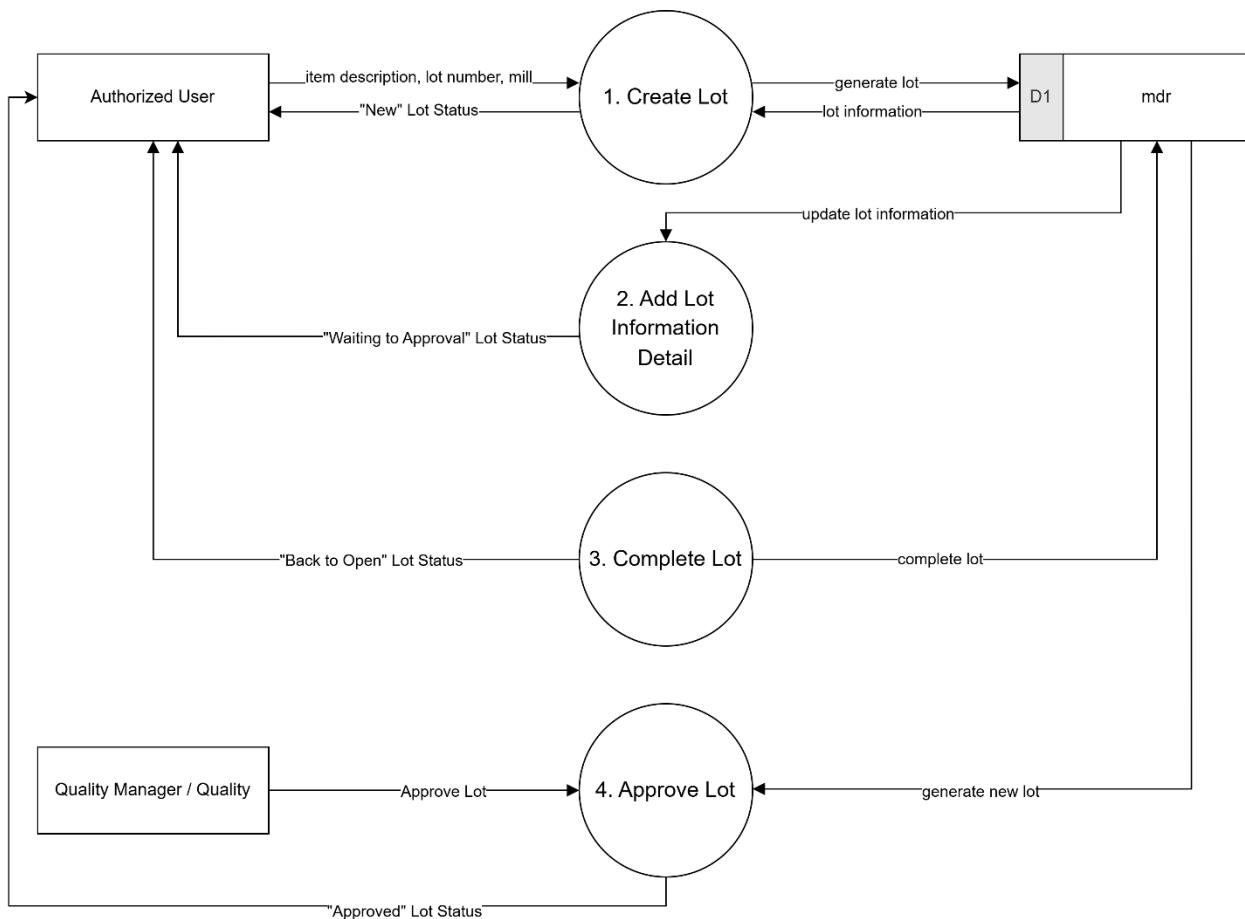


Figure 68. Lot Info - Data Flow Diagram Level 1

The key components are explained on the table below.

Table 87. Lot Info - Data Flow Diagram level 1 key components

No.	User	Input	Process	Output
1	Authorized User	Lot description, mill, etc.	Create Lot	New lot stored in mdr (D1)
2	Authorized User	Additional details	Add Lot Information Detail	Updated lot in mdr (D1)
3	Authorized User	Confirm completion	Complete Lot	Completed lot in mdr (D1)
4	Quality Manager/Quality	Approval action	Approve Lot	Approved lot in mdr (D1)

7.4.2 Open Order

7.4.2.1 Open Order - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Open Order in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the open order process.

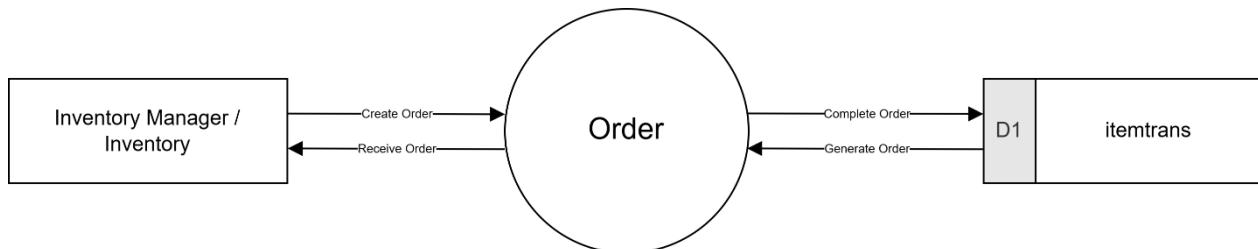


Figure 69. Open Order - Data Flow Diagram Level 0

This diagram illustrates the Order management process in Tubestream, highlighting how the Inventory Manager interacts with the system to create and receive orders. The process begins when the Inventory Manager initiates a new order, which is processed through the Order system. The system then retrieves the required item data from the itemtrans data store and completes the order. The completed order is sent back to the Inventory Manager for confirmation and receipt.

This process ensures systematic order creation, accurate item tracking, and data consistency by integrating user actions with item transaction records. It enhances inventory operations by streamlining the order fulfillment workflow.

7.4.2.2 Open Order - Data Flow Diagram Level 1

This Data Flow Diagram (DFD) level 1 visualizes the creation of order process in Tubestream, depicting the flow of data between users, processes, and databases. It offers a more detailed view compared to the Level 0 diagram above. The image below presents the DFD for Tubestream's open order process.

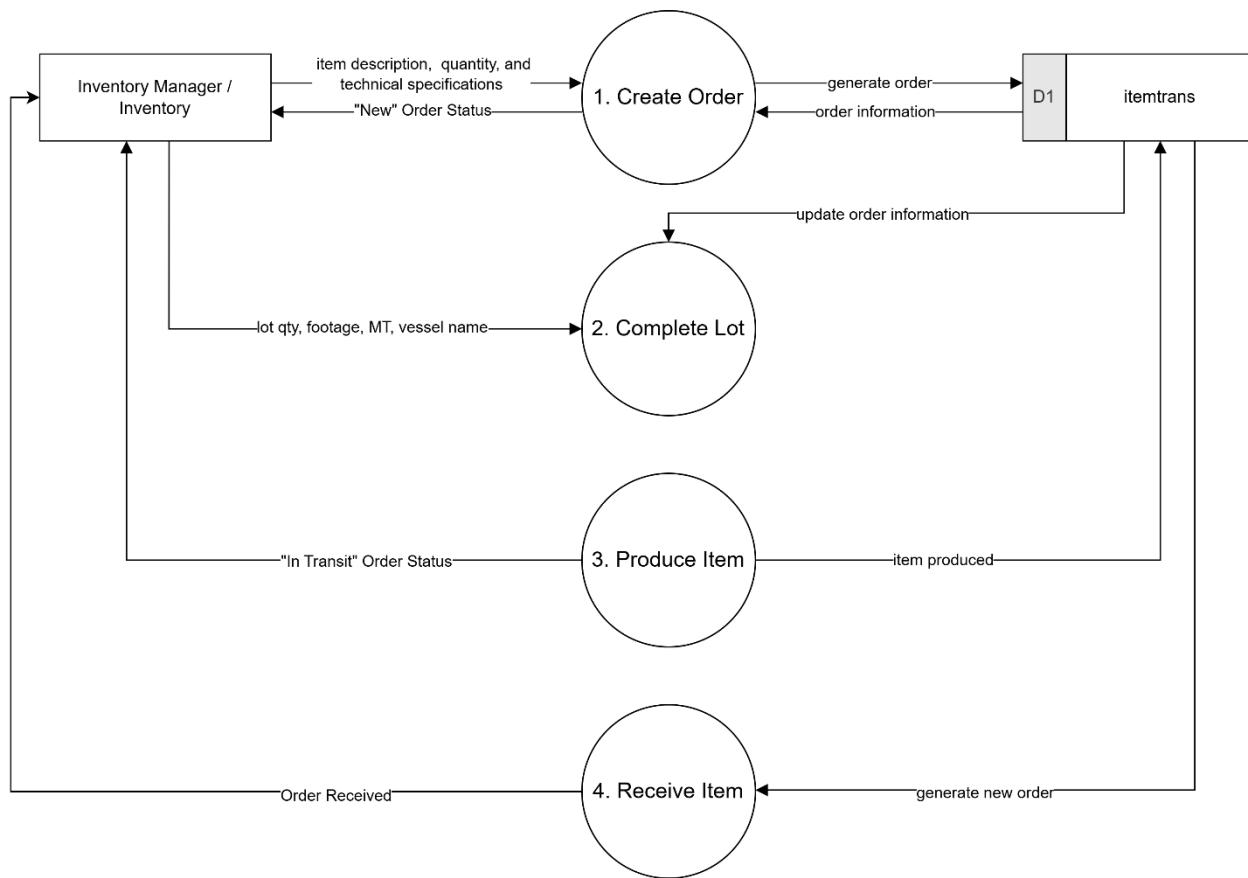


Figure 70. Open Order - Data Flow Diagram Level 1

The key components are explained on the table below.

Table 88. Open Order - Data Flow Diagram level 1 key components

No.	User	Input	Process	Output
1	Inventory Manager	Order details, specs	Create Order	New order in itemtrans (D1)
2	Inventory Manager	Lot qty, vessel name	Complete Lot	Order status updated in itemtrans (D1)
3	Inventory Manager	Production confirmation	Produce Item	Produced item data in itemtrans (D1)
4	Inventory Manager	Acknowledge receipt	Receive Item	Order received in itemtrans (D1)

7.4.3 Mill Set Up

7.4.3.1 Mill Set Up - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Mill Set Up in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the mill tag process.

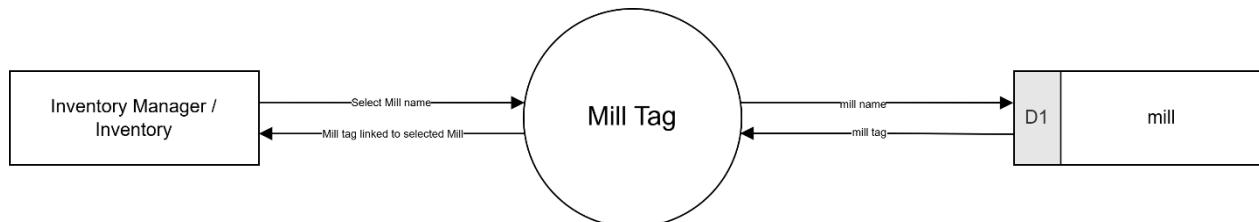


Figure 71. Mill Set Up - Data Flow Diagram Level 0

This diagram represents the Mill Tag selection and linking process. The Inventory Manager selects a Mill name through the system, which retrieves the corresponding information from the mill data store. The Mill Tag is then generated and linked to the selected Mill. The finalized tag data is sent back to the Inventory Manager.

This process ensures that every Mill Tag is correctly associated with its respective Mill, enabling traceability and supporting inventory classification based on mill origin.

7.4.4 GIMS Inquiry > Search

7.4.4.1 GIMS Inquiry > Search - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of GIMS Inquiry (Search) in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the search GIMS inquiry process.

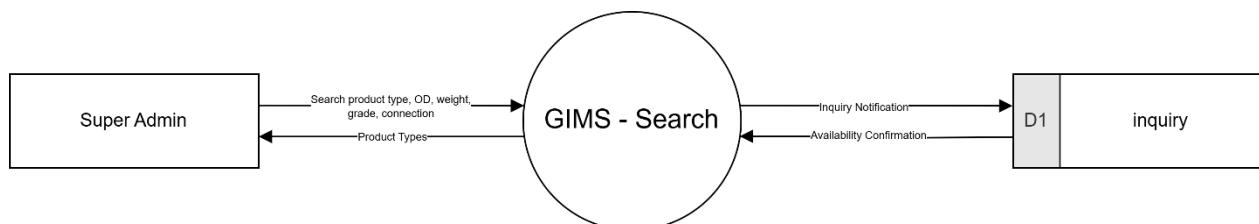


Figure 72. GIMS Inquiry > Search - Data Flow Diagram Level 0

This diagram shows how the Super Admin utilizes the GIMS - Search feature to perform product queries. The user inputs parameters such as product type, OD, weight, grade, and connection. The system accesses the inquiry data store to confirm availability and returns the product types and related information. Additionally, the system sends inquiry notifications to track requested searches.

This process supports efficient and informed decision-making by providing real-time product availability and ensuring systematic communication of inquiries within the system.

7.4.5 GIMS Inquiry > History

7.4.5.1 GIMS Inquiry > History - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of GIMS Inquiry (History) in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the history of GIMS inquiry process.

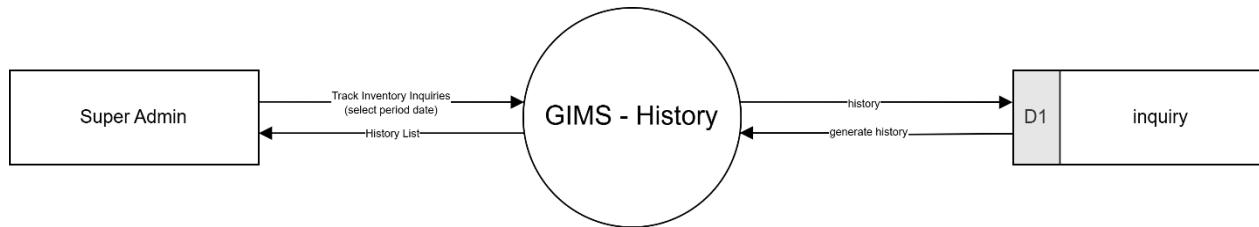


Figure 73. GIMS Inquiry > History - Data Flow Diagram Level 0

This diagram illustrates the GIMS - History feature, which allows the Super Admin to review historical inquiry data. The user selects a specific date range to track inventory inquiries. The system accesses the inquiry data store to generate the requested history, which is then returned as a list to the user.

The GIMS - History process provides comprehensive traceability of past inventory inquiries, enabling better inventory control, audit tracking, and performance review over time.

7.5 Inventory

7.5.1 Stock on Hand

7.5.1.1 Stock on Hand - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Stock on Hand in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the stock on hand process.

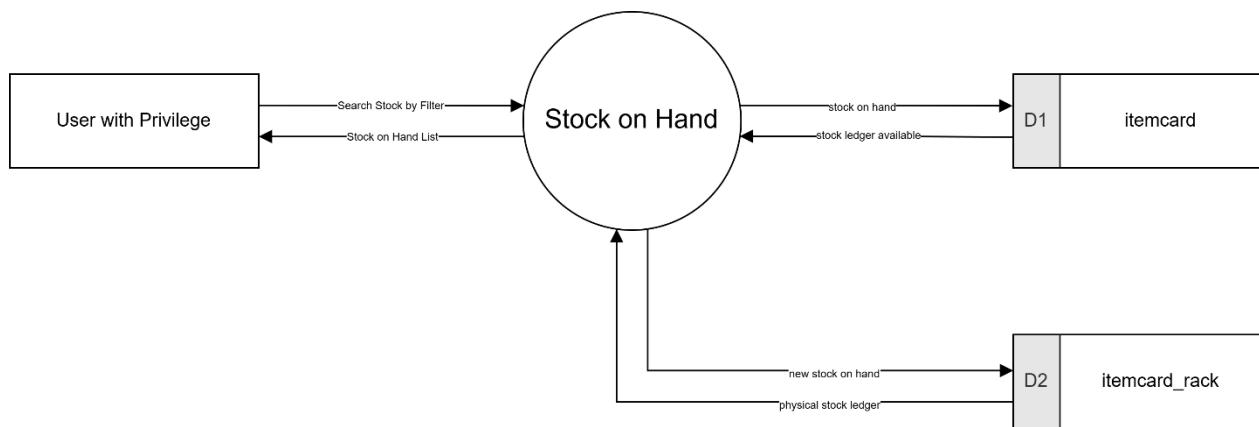


Figure 74. Stock on Hand - Data Flow Diagram Level 0

This diagram illustrates the Stock on Hand retrieval process, where a user with privileges searches inventory using filters. The system processes the request by accessing the itemcard and itemcard_rack data stores to retrieve current stock levels and physical stock ledger data. It then returns a stock on hand list to the user for review.

This process supports accurate and efficient inventory monitoring by enabling filtered searches and providing real-time stock visibility across physical and recorded data sources.

7.5.2 Adjustment

7.5.2.1 Adjustment - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Adjustment in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the creating opening balance process.

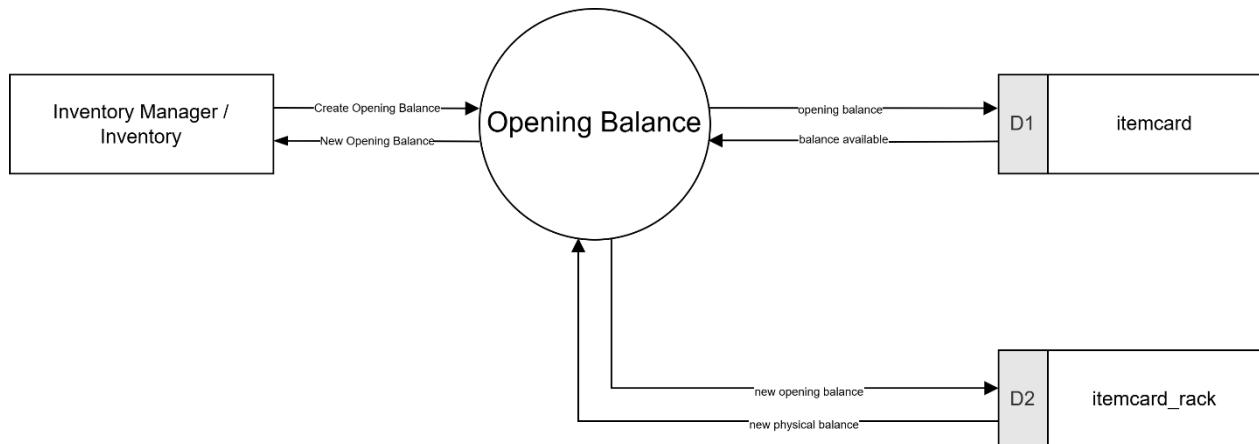


Figure 75. Adjustment - Data Flow Diagram Level 0

This diagram outlines the Opening Balance setup process. The Inventory Manager initiates the process by creating a new opening balance. The system pulls the available balance data from both itemcard and itemcard_rack to generate and store updated opening balances. The newly created balance is then returned to the Inventory Manager for confirmation.

This function ensures a reliable starting point for stock tracking by establishing an accurate baseline of item quantities in both system records and physical storage.

7.5.3 Transfer > Port / Yard

7.5.3.1 Transfer > Port / Yard - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Yard Transfer in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the yard transfer process.

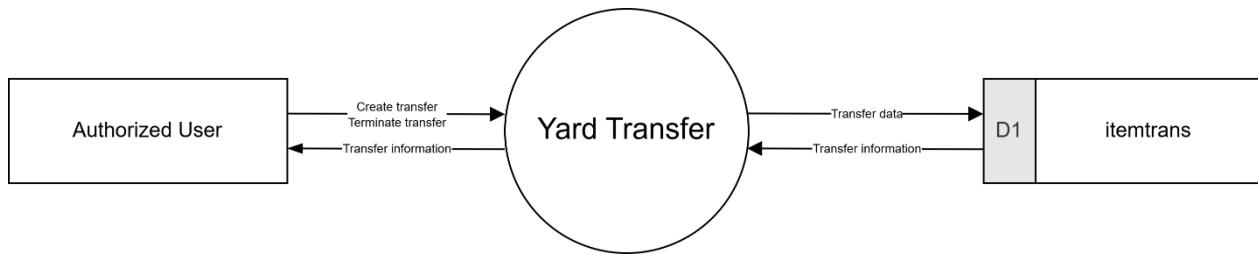


Figure 76. Transfer > Port / Yard - Data Flow Diagram Level 0

This diagram describes the Yard Transfer process in which an Authorized User creates or terminates a stock transfer between yard locations. The system communicates with the itemtrans data store to record and retrieve relevant transfer data. Transfer information is sent back to the user for review or confirmation.

This process facilitates controlled movement of inventory within the yard, enhancing transparency, accountability, and traceability of stock transfers.

7.5.3.2 Transfer > Port / Yard - Data Flow Diagram Level 1

This Data Flow Diagram (DFD) level 1 visualizes the yard transfer process in Tubestream, depicting the flow of data between users, processes, and databases. It offers a more detailed view compared to the Level 0 diagram above. The image below presents the DFD for Tubestream's yard transfer process.

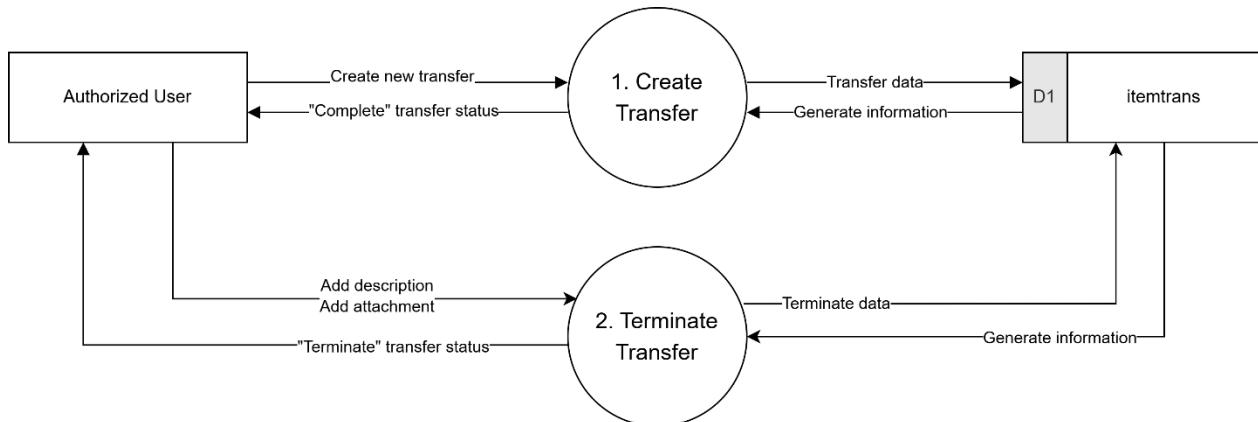


Figure 77. Transfer > Port / Yard - Data Flow Diagram Level 1

The key components are explained on the table below.

Table 89. Transfer > Port / Yard - Data Flow Diagram level 1 key components

No.	User	Input	Process	Output
1	Authorized User	Transfer details	Create Transfer	New transfer in itemtrans (D1)
2	Authorized User	Description, attachment	Terminate Transfer	Termination stored in itemtrans (D1)

7.5.4 Transfer > Allocation

7.5.4.1 Transfer > Allocation - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Allocation Transfer in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the allocation transfer process.

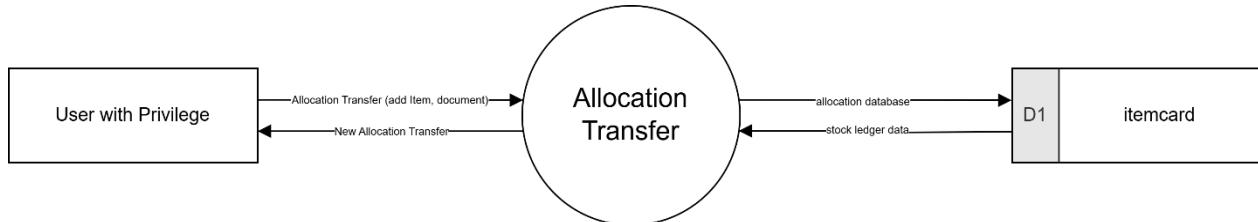


Figure 78. Transfer > Allocation - Data Flow Diagram Level 0

This diagram shows the Allocation Transfer process, where a user with privileges allocates items (along with supporting documents) for specific purposes. The system updates and retrieves data from the itemcard data store to reflect allocation changes and stock ledger information, which is returned to the user.

This ensures that inventory allocations are properly tracked, and updates are accurately reflected in the stock records, minimizing errors and enhancing operational planning.

7.5.5 Transfer > Rack

7.5.5.1 Transfer > Rack - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Rack Transfer in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the rack transfer process.

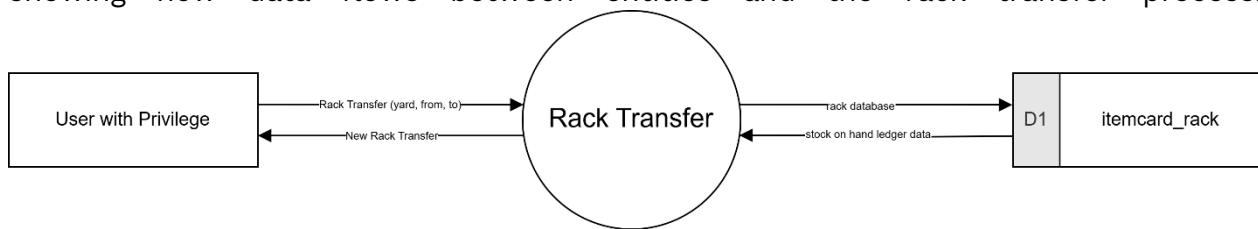


Figure 79. Transfer > Rack - Data Flow Diagram Level 0

This diagram illustrates the Rack Transfer process, where a user with privileges manages the movement of inventory between racks (defined by yard, source, and destination locations). The system interacts with the itemcard_rack data store to update and retrieve rack-specific stock data. The updated rack transfer information is sent back to the user.

This process improves warehouse organization and stock control by ensuring that stock transfers between storage racks are systematically recorded and traceable.

7.6 Vendor Management

7.6.1 3rd Party Instruction

7.6.1.1 3rd Party Instruction - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of 3rd Party Instruction in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the 3rd party instruction process.

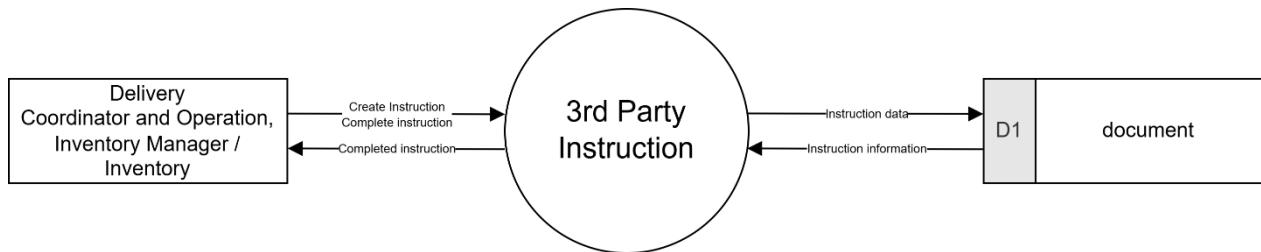


Figure 80. 3rd Party Instruction - Data Flow Diagram Level 0

This diagram illustrates the 3rd Party Instruction process, demonstrating how information flows among key stakeholders and the system. The process begins with the Delivery Coordinator and Operation or Inventory Manager, who creates and completes instruction data, then transmits it to the 3rd Party Instruction system. The system processes this data and forwards the instruction information to the document repository (D1), ensuring that all instruction records are properly documented and stored.

The 3rd Party Instruction process facilitates structured coordination by managing the creation, completion, and storage of instructions. It ensures that instructions are accurately generated, processed, and communicated to relevant parties, with proper documentation maintained for operational transparency and accountability.

7.6.1.2 3rd Party Instruction - Data Flow Diagram Level 1

This Data Flow Diagram (DFD) level 1 visualizes the 3rd party instruction process in Tubestream, depicting the flow of data between users, processes, and databases. It offers a more detailed view compared to the Level 0 diagram above. The image below presents the DFD for Tubestream's 3rd party instruction process.

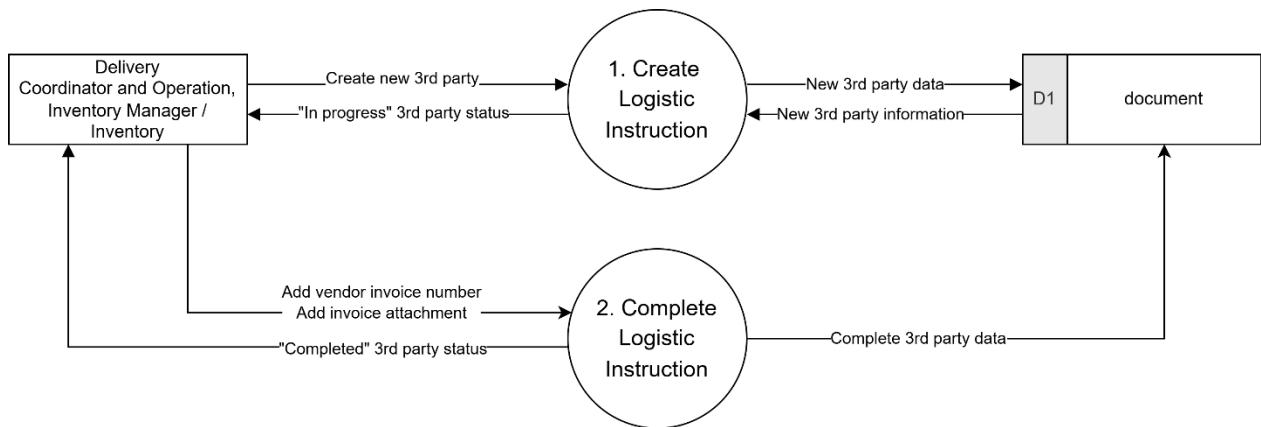


Figure 81. 3rd Party Instruction - Data Flow Diagram Level 1

The key components are explained on the table below.

Table 90. 3rd Party Instruction - Data Flow Diagram level 1 key components

No.	User	Input	Process	Output
1	Delivery Coordinator / Inventory	New 3 rd party info	Create Logistic Instruction	New 3 rd party in document (D1)
2	Delivery Coordinator / Inventory	Invoice number, attachment	Complete Logistic Instruction	Completed 3rd party in document (D1)

7.7 Production

7.7.1 Production

7.7.1.1 Production - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Item Conversion in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the production process.

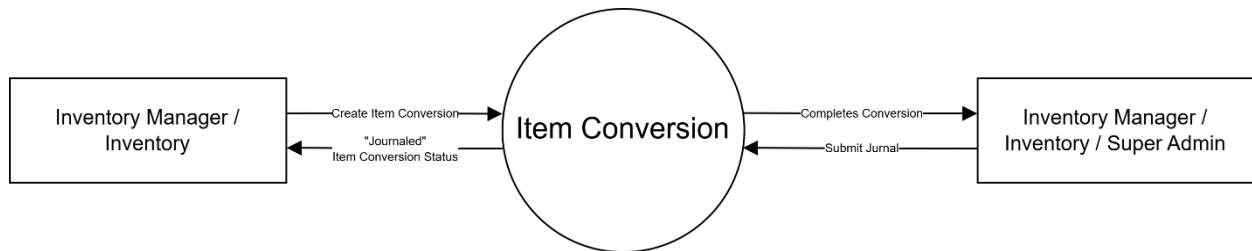


Figure 82. Production - Data Flow Diagram Level 0

This diagram represents the Item Conversion process, focusing on how inventory changes are managed and recorded. The process is initiated by the Inventory Manager, who creates an item conversion entry. The system processes this and passes it to the Inventory Manager, Inventory, or Super Admin to complete the conversion and submit a journal. Once the journal is submitted, the system updates the status to “Jounaled” and sends it back to the Inventory Manager.

This process ensures proper handling and traceability of inventory changes. By involving relevant roles for creation, validation, and journaling, the system maintains accurate records and supports audit-ready inventory management.

7.7.1.2 Production - Data Flow Diagram Level 1

This Data Flow Diagram (DFD) level 1 visualizes the item conversion process in Tubestream, depicting the flow of data between users, processes, and databases. It offers a more detailed view compared to the Level 0 diagram above. The image below presents the DFD for Tubestream's production process.

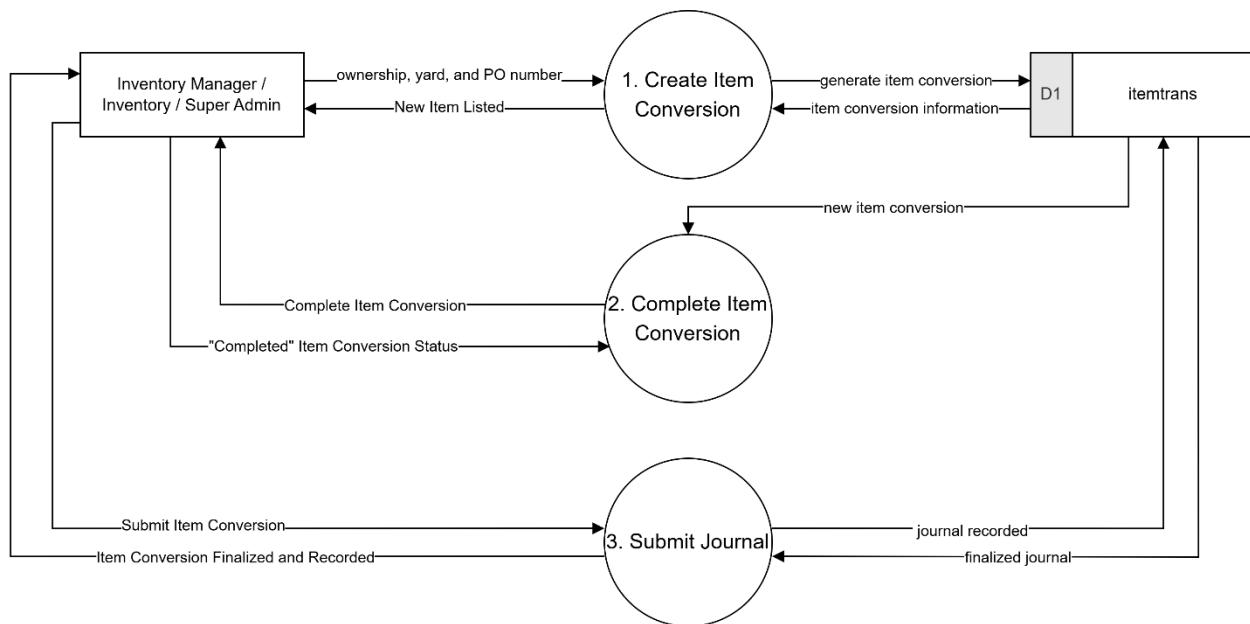


Figure 83. Production - Data Flow Diagram Level 1

The key components are explained on the table below.

Table 91. Production - Data Flow Diagram level 1 key components

No.	User	Input	Process	Output
1	Inventory Manager / Super Admin	Ownership, PO, yard info	Create Item Conversion	New conversion in itemtrans (D1)
2	Inventory Manager / Super Admin	Conversion completion info	Complete Item Conversion	Completed status in itemtrans (D1)
3	Inventory Manager / Super Admin	Journal data	Submit Journal	Finalized journal in itemtrans (D1)

7.8 KPI & Report

7.8.1 KPI

7.8.1.1 Account Summary - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Account Summary in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the account summary process.

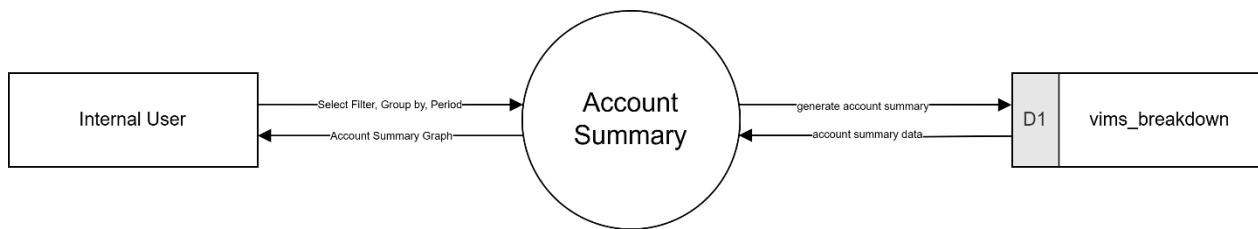


Figure 84. Account Summary - Data Flow Diagram Level 0

This diagram shows the Account Summary process, where internal users interact with the system to generate financial or account-based summaries. Internal Users select specific filters, groupings, or time periods to customize the report. The system then generates a summary using data from the vims_breakdown database and sends the resulting graph back to the user.

The Account Summary process streamlines internal financial reporting by allowing flexible data grouping and period selection. It supports informed decision-making by delivering tailored visual summaries based on up-to-date data.

7.8.1.2 Yard Info - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Yard Info in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the yard info process.

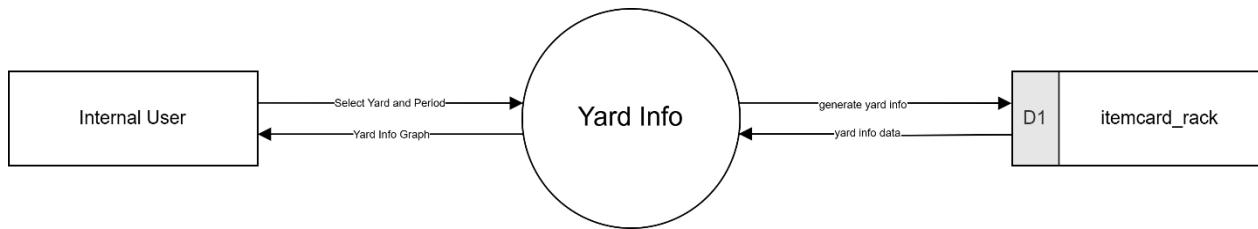


Figure 85. Yard Info - Data Flow Diagram Level 0

This diagram shows the Yard Info process, where internal users interact with the system to generate yard info summaries. Internal Users select specific yard and time periods to customize the report. The system then generates an information using data from the itemcard_rack database and sends the resulting graph back to the user.

7.8.1.3 Commitment Statistic - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Commitment Statistic in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the commitment statistic process.

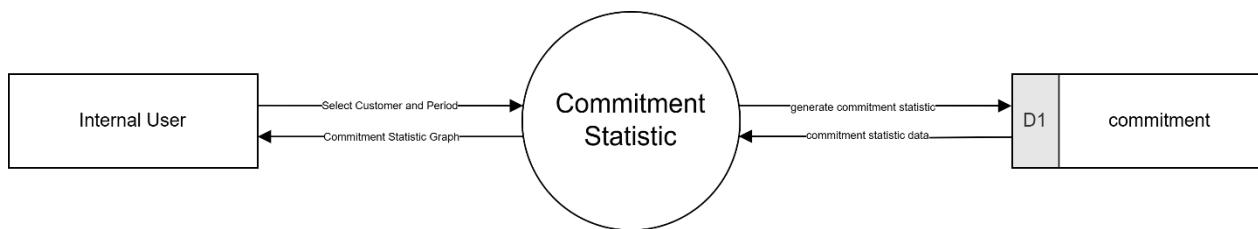


Figure 86. Commitment Statistic - Data Flow Diagram Level 0

This diagram shows the Commitment Statistic process, where internal users interact with the system to generate commitment statistic summaries. Internal Users select specific customer and time periods to customize the report. The system then generates an information using data from the commitment database and sends the resulting graph back to the user.

7.8.1.4 Commitments Ageing - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Commitment Ageing in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the commitment ageing process.

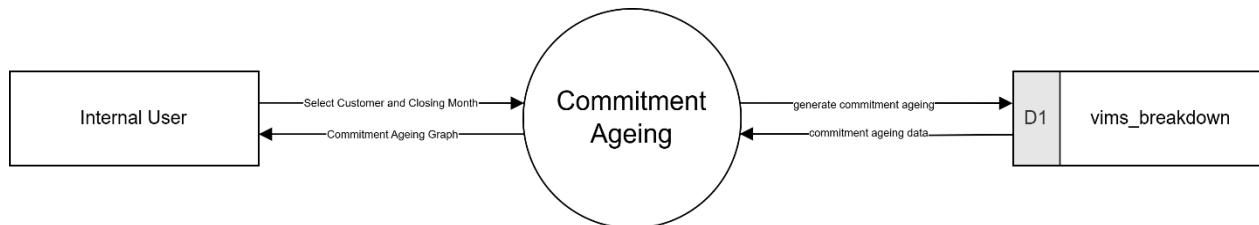


Figure 87. Commitments Ageing - Data Flow Diagram Level 0

This diagram shows the Commitment Ageing process, where internal users interact with the system to generate commitment ageing summaries. Internal Users select specific customer and closing month to customize the report. The system then generates an information using data from the vims_breakdown database and sends the resulting graph back to the user.

7.8.1.5 Invoice - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Invoice in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the invoice process.

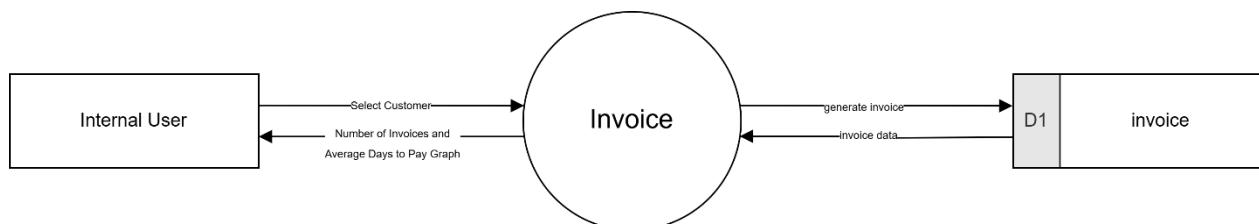


Figure 88. Invoice - Data Flow Diagram Level 0

This diagram shows the Invoice process, where internal users interact with the system to generate invoice summaries. Internal Users select specific customer to customize the report. The system then generates an information using data from the invoice database and sends the resulting graph to the user.

7.8.1.6 MR Order Notification - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of MR Order Notification in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the MR order notification process.

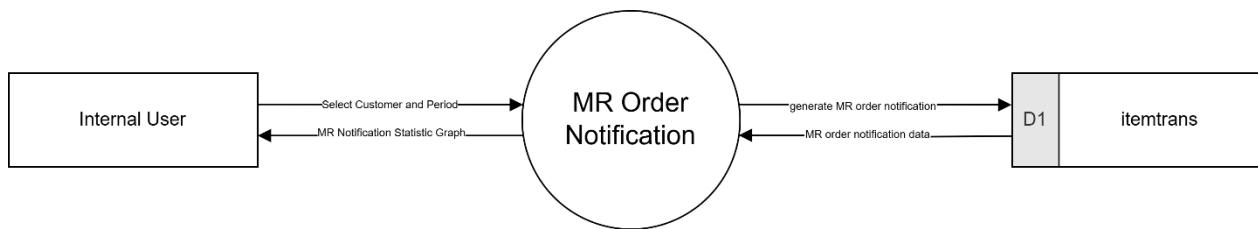


Figure 89. MR Order Notification - Data Flow Diagram Level 0

This diagram shows the MR Order Notification process, where internal users interact with the system to generate MR order notification summaries. Internal Users select specific customer and period to customize the report. The system then generates an information using data from the itemtrans database and sends the resulting statistic graph to the user.

7.8.1.7 Material Requisition Compliance to issue rig / jetty - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Material Requisition Compliance to issue rig / jetty in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the material requisition compliance to issue rig / jetty process.

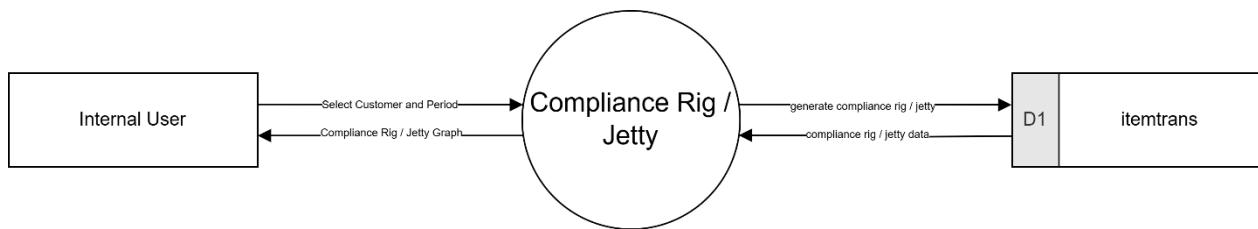


Figure 90. Material Requisition Compliance to issue rig / jetty - Data Flow Diagram Level 0

This diagram shows the MR Compliance to Issue Rig / Jetty process, where internal users interact with the system to generate compliance to issue rig / jetty summaries. Internal Users select specific customer and period to customize the report. The system then generates an information using data from the itemtrans database and sends the resulting graph to the user.

7.8.1.8 Stock View - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Stock View in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the stock view process.

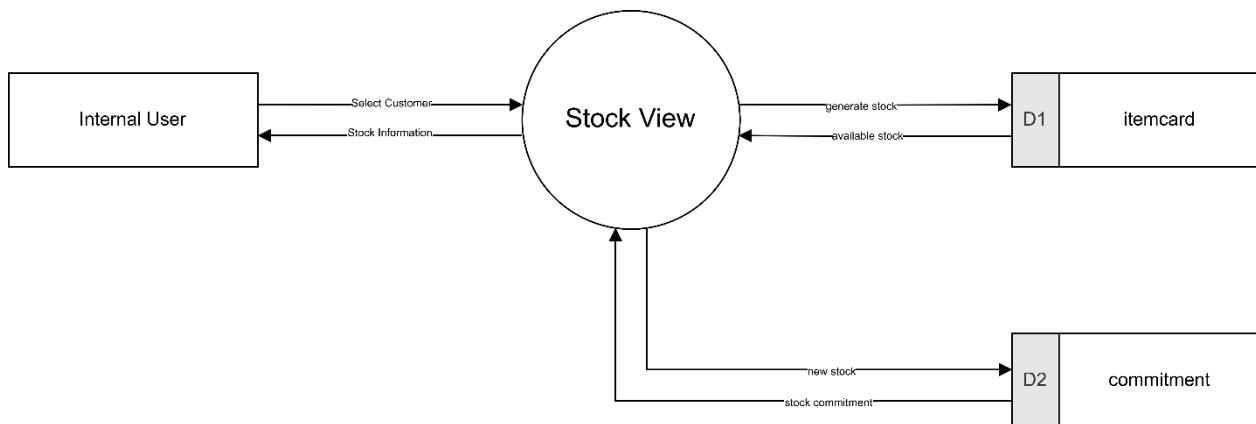


Figure 91. Stock View - Data Flow Diagram Level 0

This diagram illustrates the Stock View process, showing how data flows between internal users and the system. The process begins when an Internal User selects specific customer data to request stock information. The system then retrieves stock availability from the itemcard data store and stock commitment information from the commitment data store. Using this data, the system generates a comprehensive stock report and presents it back to the Internal User.

The Stock View process ensures accurate and up-to-date stock visibility by integrating available and committed stock data. It helps internal stakeholders make informed decisions by providing timely stock insights based on customer context, ensuring efficient inventory monitoring and management.

7.8.1.9 Inventory Profile - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Inventory Profile in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the inventory profile process.

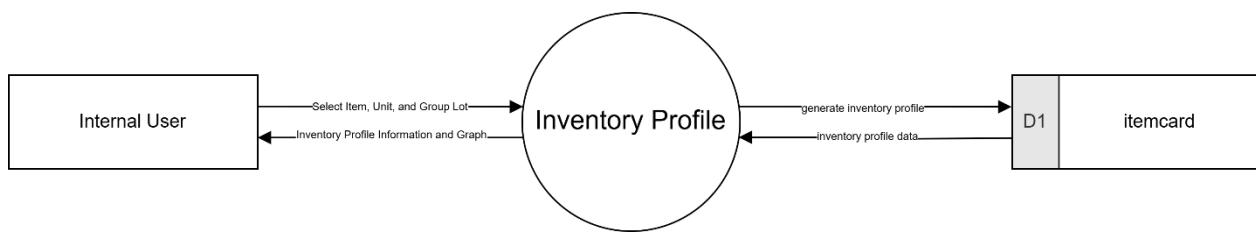


Figure 92. Inventory Profile - Data Flow Diagram Level 0

This diagram shows the Inventory Profile process, where internal users interact with the system to generate inventory profile summaries. Internal Users select specific item, unit, and group lot to customize the report. The system then generates an information using data from the itemcard database and sends the resulting graph to the user.

7.8.1.10 Material Requisition Statistic - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Material Requisition Statistic in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the material requisition statistic process.

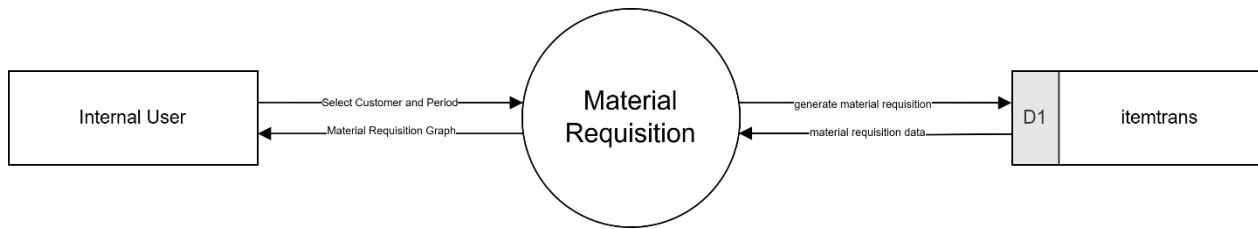


Figure 93. Material Requisition Statistic - Data Flow Diagram Level 0

This diagram shows the MR Statistic process, where internal users interact with the system to generate MR statistic summaries. Internal Users select specific customer and period to customize the report. The system then generates an information using data from the itemtrans database and sends the resulting graph to the user.

7.8.1.11 Ageing - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Ageing in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the ageing process.

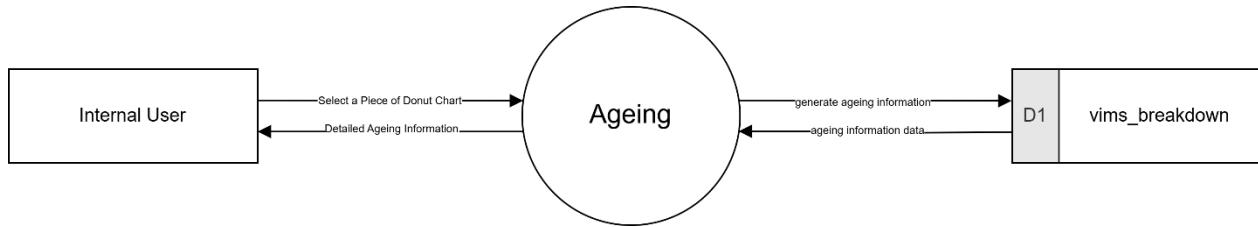


Figure 94. Ageing - Data Flow Diagram Level 0

This diagram shows the Ageing process, where internal users interact with the system to generate compliance to ageing summaries. Internal Users select specific a piece of donut chart to customize the report. The system then generates an information using data from the vims_breakdown database and sends the resulting detailed ageing information to the user.

7.8.1.12 Material Requisition Compliance - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Material Requisition Compliance in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the material requisition compliance process.

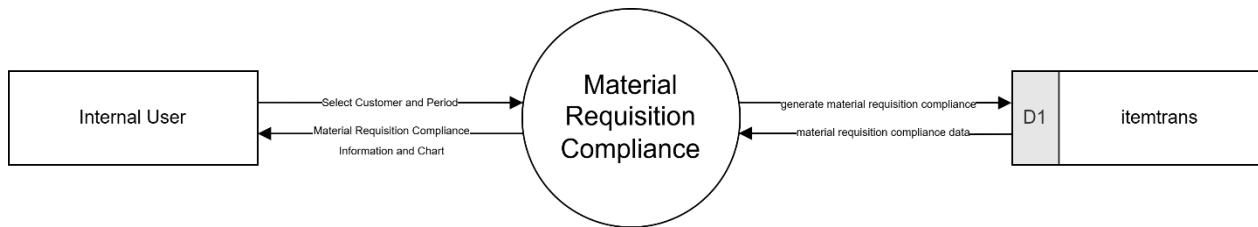


Figure 95. Material Requisition Compliance - Data Flow Diagram Level 0

This diagram shows the MR Compliance process, where internal users interact with the system to generate MR compliance summaries. Internal Users select specific customer and period to customize the report. The system then generates an information using

data from the itemtrans database and sends the resulting information and chart to the user.

7.8.1.13 Stock Management – Open End - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Stock Management – Open End in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the stock management – open end process.

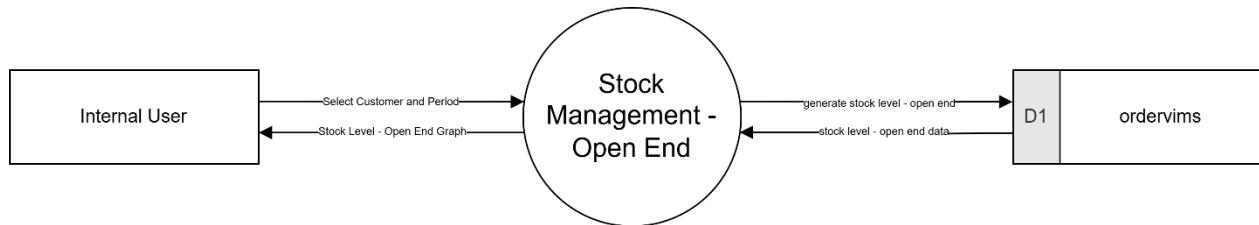


Figure 96. Stock Management – Open End – Data Flow Diagram Level 0

This diagram shows the Stock Management – Open End process, where internal users interact with the system to generate stock management – open end summaries. Internal Users select specific customer and period to customize the report. The system then generates an information using data from the ordervims database and sends the resulting graph to the user.

7.8.1.14 Stock Management – Closed End - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Stock Management – Closed End in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the stock management – closed end process.



Figure 97. Stock Management – Closed End – Data Flow Diagram Level 0

This diagram shows the Stock Management – Closed End process, where internal users interact with the system to generate stock management – closed end summaries. Internal Users select specific customer, period, and item to customize the report. The system then generates an information using data from the ordervims database and sends the resulting graph to the user.

7.8.2 Report

7.8.2.1 Adnoc Requirement

7.8.2.1.1 ADNOC ERP Log - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of ADNOC ERP Log in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the ADNOC ERP log process.

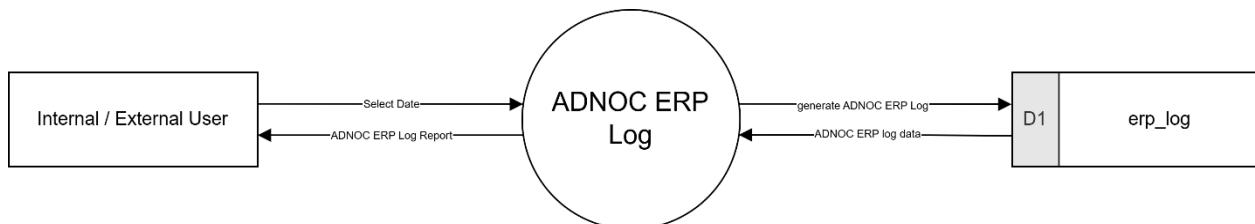


Figure 98. ADNOC ERP Log - Data Flow Diagram Level 0

This diagram describes how Internal or External Users access the ADNOC ERP Log. By selecting a date, the user triggers the system to retrieve and generate log data from the erl_log database. The system then provides a detailed ERP Log Report back to the user.

The ADNOC ERP Log process ensures transparency and traceability of system events by allowing users to view and extract logs within a specified timeframe for monitoring or auditing purposes.

7.8.2.1.2 Material Requisition - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Material Requisition in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the material requisition process.

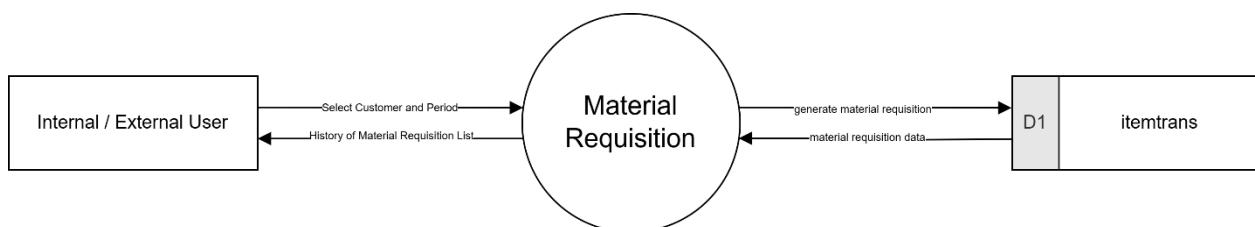


Figure 99. Material Requisition - Data Flow Diagram Level 0

This diagram describes how Internal or External Users access the Material Requisition. By selecting a specific customer and period, the user triggers the system to retrieve and generate material requisition data from the itemtrans database. The system then provides a detailed history of material requisition list back to the user.

7.8.2.1.3 Rig Return - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Rig Return in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the rig return process.

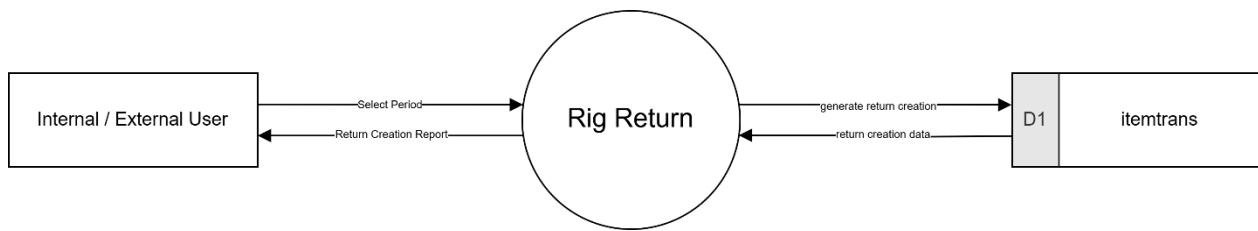


Figure 100. Rig Return - Data Flow Diagram Level 0

This diagram describes how Internal or External Users access the Rig Return. By selecting a specific period, the user triggers the system to retrieve and generate return creation data from the itemtrans database. The system then provides a detailed return creation report back to the user.

7.8.2.1.4 Stock Aging - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Stock Aging in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the stock aging process.

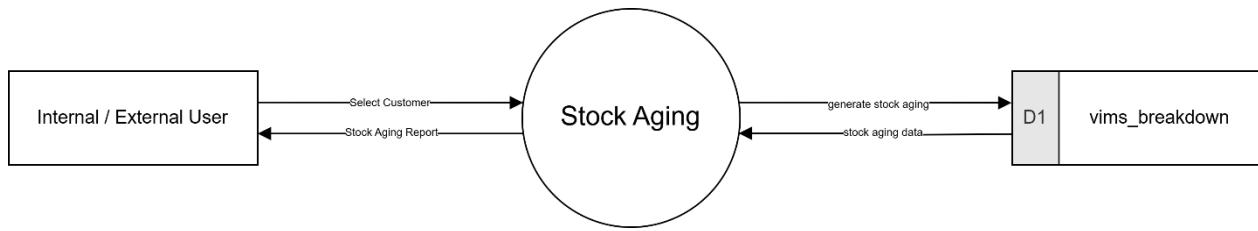


Figure 101. Stock Aging - Data Flow Diagram Level 0

This diagram describes how Internal or External Users access the Stock Aging. By selecting a specific customer, the user triggers the system to retrieve and generate stock aging data from the vims_breakdown database. The system then provides a detailed stock aging report back to the user.

7.8.2.1.5 Stock View - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Stock View in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the stock view process.

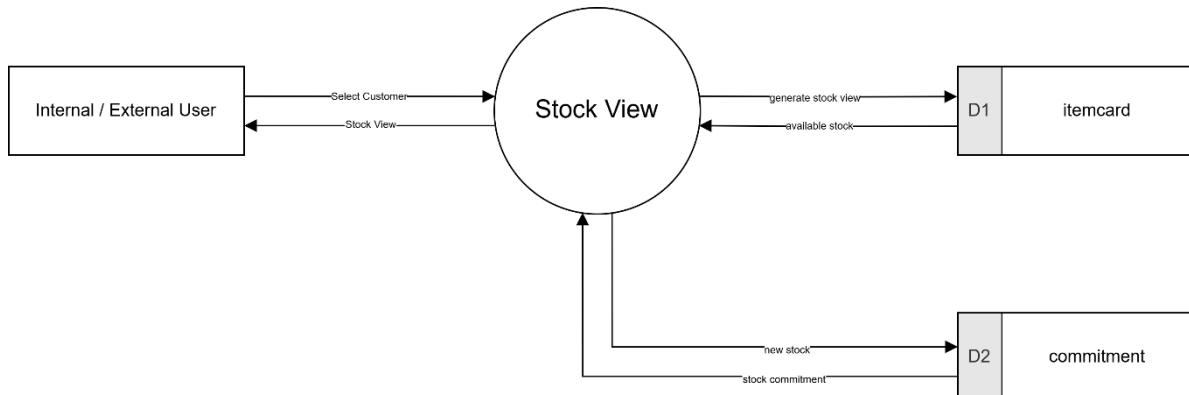


Figure 102. Stock View - Data Flow Diagram Level 0

This diagram illustrates the Stock View process in Tubestream, detailing how stock information is accessed and processed by various system components. The process begins when an internal or external user submits a customer-related query. The system retrieves the relevant stock data from the itemcard data store and commitment information from the commitment data store. It then generates a consolidated stock view that includes both available stock and committed quantities, which is then returned to the user.

The Stock View process ensures accurate and real-time visibility of inventory by integrating multiple data sources. It allows users to view filtered stock information, enhancing transparency and decision-making for both internal operations and external requests.

7.8.2.1.6 Transaction Report - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Transaction Report in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the transaction report process.

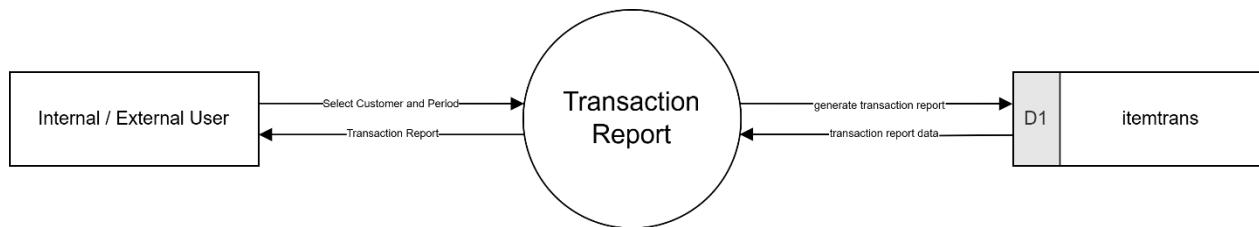


Figure 103. Transaction Report - Data Flow Diagram Level 0

This diagram describes how Internal or External Users access the Transaction Report. By selecting a specific customer and period, the user triggers the system to retrieve and generate transaction report data from the itemtrans database. The system then provides a detailed transaction report back to the user.

7.8.2.2 Commercial

7.8.2.2.1 ACCS Commitment Reports - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of ACCS Commitment Reports in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the ACCS commitment reports process.



Figure 104. ACCS Commitment Reports - Data Flow Diagram Level 0

This diagram outlines the ACCS Commitment Report generation process for Internal Users. The user selects a customer to initiate the request, which the system uses to

generate commitment report data from the commitment database. The resulting ACCS commitment report is then returned to the user.

This process supports accountability and decision-making by providing users with timely access to customer-specific commitment information stored within the ACCS system.

7.8.2.2.2 Ageing Detail Report - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Ageing Detail Report in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the ageing detail report process.

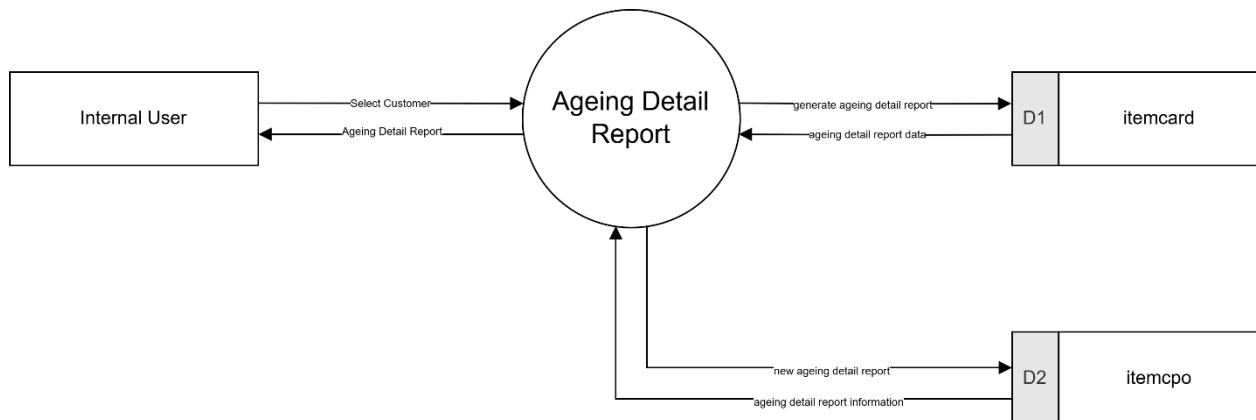


Figure 105. Ageing Detail Report - Data Flow Diagram Level 0

This diagram illustrates the Ageing Detail Report process in Tubestream. An internal user initiates the process by selecting a customer. The system generates a detailed ageing report by retrieving relevant data from the itemcard and itemcpo data stores. The processed report is then returned to the user for review.

The Ageing Detail Report provides granular visibility into inventory aging, helping internal users monitor product shelf life and identify slow-moving items, supporting informed inventory control decisions.

7.8.2.2.3 Ageing Recap - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Ageing Recap in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the ageing recap process.

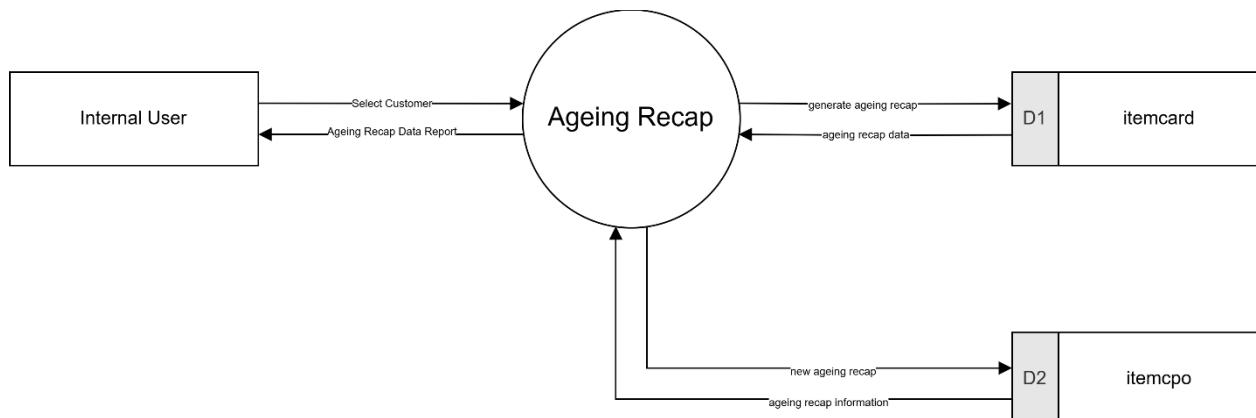


Figure 106. Ageing Recap - Data Flow Diagram Level 0

This diagram describes the Ageing Recap process, which summarizes inventory aging information for internal users. After selecting a customer, the system compiles data from the itemcard and itemcpc data stores to generate the ageing recap report. The compiled report is then provided back to the user.

The Ageing Recap process streamlines the presentation of aging data, allowing users to quickly assess overall stock age distribution and take timely actions for inventory optimization.

7.8.2.2.4 Commitment Ageing Recap - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Commitment Ageing Recap in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the commitment ageing recap process.



Figure 107. Commitment Ageing Recap - Data Flow Diagram Level 0

This diagram outlines the Commitment Ageing Recap generation process for Internal Users. The user selects a specific closing month to initiate the request, which the system uses to generate commitment ageing recap data from the commitment database. The resulting Commitment Ageing Recap data report is then returned to the user.

7.8.2.2.5 Dispatch Reports - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Dispatch Reports in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the dispatch reports process.



Figure 108. Dispatch Reports - Data Flow Diagram Level 0

This diagram outlines the Dispatch Reports generation process for Internal Users. The user selects a customer to initiate the request, which the system uses to generate dispatch report data from the itemtrans database. The resulting dispatch report is then returned to the user.

7.8.2.2.6 Material Requisition - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Material Requisition in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the material requisition process.

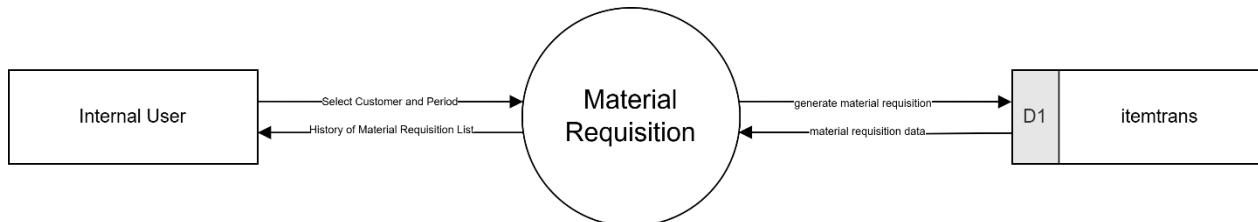


Figure 109. Material Requisition - Data Flow Diagram Level 0

This diagram outlines the Material Requisition generation process for Internal Users. The user selects a customer and period to initiate the request, which the system uses to generate material requisition data from the itemtrans database. The resulting history of material requisition list is then returned to the user.

7.8.2.2.7 Material Requisition Invoice - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Material Requisition Invoice in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the material requisition invoice process.

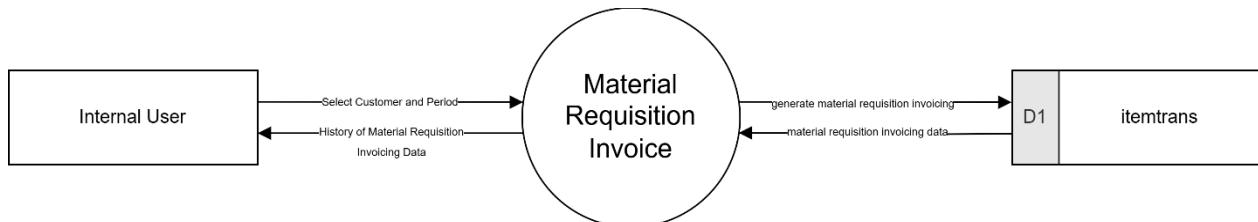


Figure 110. Material Requisition Invoice - Data Flow Diagram Level 0

This diagram outlines the Material Requisition Invoice generation process for Internal Users. The user selects a customer and period to initiate the request, which the system uses to generate material requisition invoicing data from the itemtrans database. The resulting history of material requisition invoicing data is then returned to the user.

7.8.2.2.8 OCTG Commitment Reports - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of OCTG Commitment Reports in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the OCTG commitment reports process.



Figure 111. OCTG Commitment Reports - Data Flow Diagram Level 0

This diagram outlines the OCTG Commitment Reports generation process for Internal Users. The user selects a customer to initiate the request, which the system uses to generate OCTG commitment and incoming reports data from the commitment database. The resulting OCTG commitment and incoming reports are then returned to the user.

7.8.2.2.9 Return Creation - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Return Creation in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the return creation process.

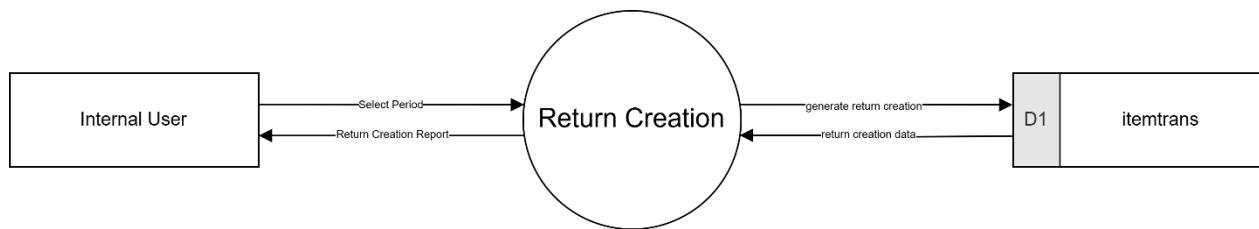


Figure 112. Return Creation - Data Flow Diagram Level 0

This diagram outlines the Return Creation generation process for Internal Users. The user selects a period to initiate the request, which the system uses to generate return creation data from the itemtrans database. The resulting return creation report is then returned to the user.

7.8.2.3 Inventory

7.8.2.3.1 Inventory Report - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Inventory Report in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the inventory report process.



Figure 113. Inventory Report - Data Flow Diagram Level 0

This diagram shows the Inventory Report process where an internal user selects a customer and a specific period. The system retrieves relevant inventory data from the itemcard data store and compiles it into a report, which is returned to the user.

This process supports effective reporting by enabling users to analyze inventory levels over time, track changes, and ensure alignment with operational goals and audit requirements.

7.8.2.3.2 Inventory Review - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Inventory Review in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the inventory review process.



Figure 114. Inventory Review - Data Flow Diagram Level 0

This diagram outlines the Inventory Review process. An internal user filters inventory by various criteria (e.g., customer, location, quarantine status, non-forecasted items, damage, and production orders). The system pulls matching data from the itemcard data store and compiles a review report, which is returned to the user.

The Inventory Review process supports comprehensive inventory assessment, helping stakeholders identify issues, validate inventory accuracy, and plan corrective actions effectively.

7.8.2.3.3 Stock Summary - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Stock Summary in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the stock summary process.

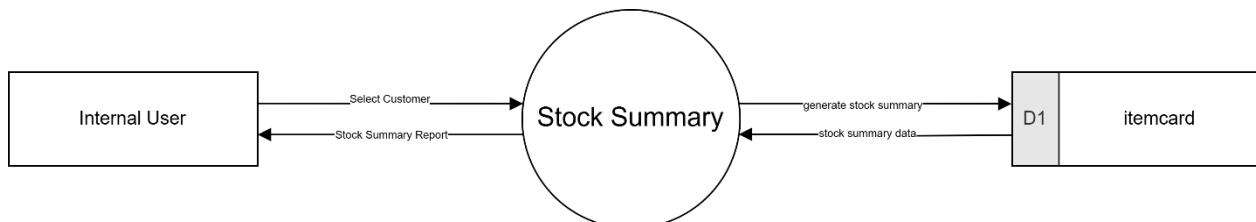


Figure 115. Stock Summary - Data Flow Diagram Level 0

This diagram illustrates the Stock Summary process where an internal user selects a customer to view stock data. The system retrieves summary-level inventory information from the itemcard data store and returns a concise stock summary report to the user.

This process simplifies inventory monitoring by providing a high-level view of current stock positions, enabling quick insights and streamlined reporting.

7.8.2.4 Operational Report

7.8.2.4.1 Movement Detail Report - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Movement Detail Report in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the movement detail report process.



Figure 116. Movement Detail Report - Data Flow Diagram Level 0

This diagram illustrates the Movement Detail Report process, showing how data flows between the Internal User and the system. The Internal User initiates the process by selecting a section, customer, and period to generate a report. The system retrieves and processes the relevant data from the itemtrans data store to generate the movement detail report. Once the report is prepared, it is sent back to the Internal User for review.

The Movement Detail Report process ensures accurate tracking and analysis of item movement by utilizing key input parameters. It supports informed decision-making by delivering detailed transaction data filtered by section, customer, and period. This process helps maintain transparency, improve operational monitoring, and ensure accountability across internal operations.

7.8.2.4.2 Movement Report - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Movement Report in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the movement report process.

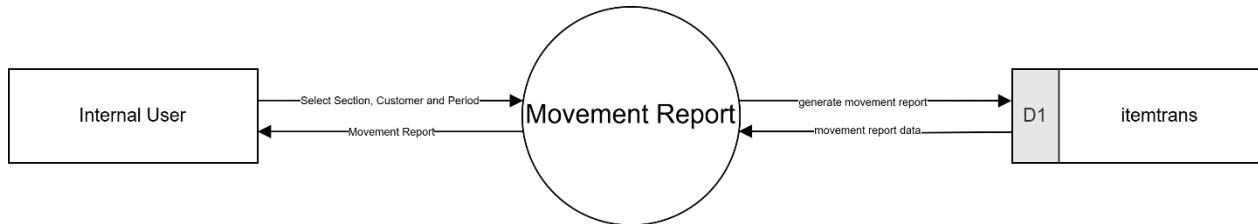


Figure 117. Movement Report - Data Flow Diagram Level 0

This diagram illustrates the Movement Report process, where Internal Users generate detailed reports on item transactions. By selecting the section, customer, and period, the Internal User initiates the report request. The system then retrieves and processes the necessary transaction data from the itemtrans data store and generates the report. The completed Movement Report is then returned to the Internal User. This ensures efficient and accurate reporting of item movements based on specified filters.

7.8.2.4.3 Ownership Transfer & Return Ledger Report - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Ownership Transfer & Return Ledger Report in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the ownership transfer & return ledger report process.



Figure 118. Ownership Transfer & Return Ledger Report - Data Flow Diagram Level 0

This diagram describes the Ownership Transfer & Return Ledger Report process. The Internal User begins by selecting a date range to define the report period. The system processes relevant ownership transfer and return data from the itemtrans data store and generates a ledger report. Once compiled, the report is sent back to the Internal User. This supports comprehensive tracking of inventory ownership changes and return activities, ensuring audit readiness and accountability.

7.8.2.5 Quality & HSE

7.8.2.5.1 Damaged Inspection Result - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Damaged Inspection Result in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the damaged inspection result process.



Figure 119. Damaged Inspection Result - Data Flow Diagram Level 0

This diagram outlines the process for generating a Damaged Inspection Result report. Internal or External Users can initiate the process by selecting a consignment or customer. The system retrieves damage inspection data from the itemtally data store and generates a detailed report. The completed report is returned to the user. This process supports effective tracking of damaged goods and facilitates responsive decision-making for corrective actions or claims.

7.8.2.5.2 Inspection Creation - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Inspection Creation in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the inspection creation process.

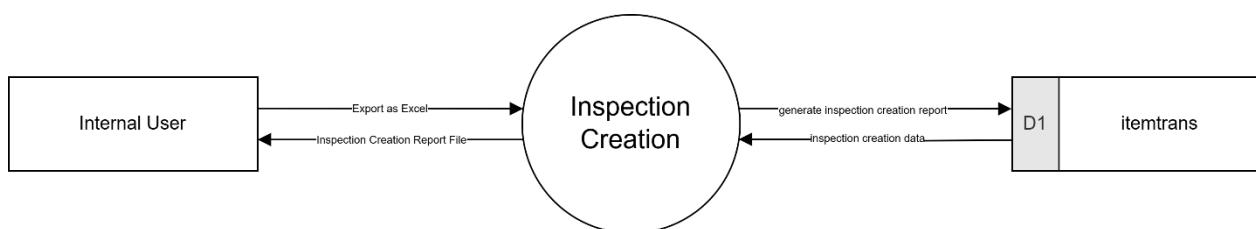


Figure 120. Inspection Creation - Data Flow Diagram Level 0

This diagram represents the Inspection Creation process. Internal Users initiate the action by exporting data to Excel for reporting purposes. The system then processes inspection-related data from the itemtrans data store and generates an inspection creation report. Once finalized, a report file is returned to the user. This process ensures that inspections are systematically recorded, documented, and made available for further analysis or regulatory compliance.

7.8.2.6 System Report

7.8.2.6.1 Inactive User - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Inactive User in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the inactive user process.

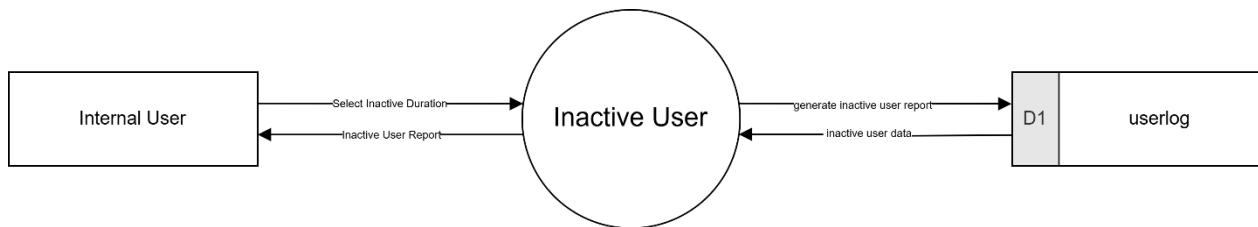


Figure 121. Inactive User - Data Flow Diagram Level 0

This diagram illustrates the Inactive User report process, showing how data flows between internal users and the system to monitor user activity. The process begins when an Internal User selects a specific duration to identify inactivity. The system then retrieves relevant login data from the userlog data store, processes it, and generates an Inactive User Report, which is then returned to the Internal User.

The Inactive User process helps internal stakeholders monitor user engagement by identifying accounts that have been inactive for a selected period. This enables better management of user access, facilitates auditing, and supports decision-making regarding user activity within the system.

7.8.2.6.2 List Super Admin - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of List Super Admin in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the list super admin process.



Figure 122. List Super Admin - Data Flow Diagram Level 0

This diagram illustrates the List Super Admin process, where Internal Users request and retrieve a list of Super Admins in the system. The system queries the user data store to generate the list and then returns the relevant Super Admin data to the

Internal User. This process ensures visibility and accessibility of administrative roles for internal management and auditing purposes.

7.8.2.6.3 Login Recap - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Login Recap in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the login recap process.

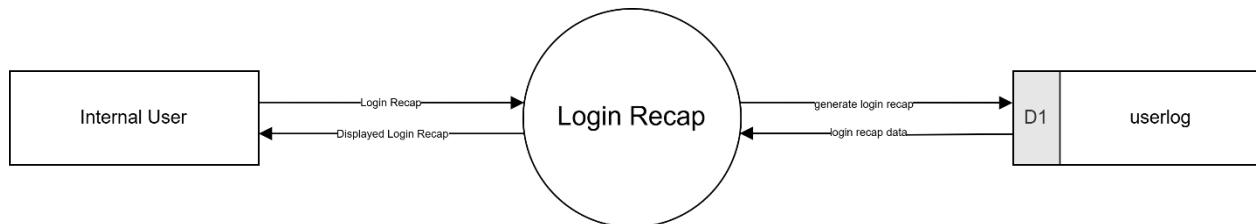


Figure 123. Login Recap - Data Flow Diagram Level 0

This diagram describes the Login Recap process, where Internal Users can review system login activity. The system retrieves login history from the userlog data store and compiles a recap report, which is then displayed to the Internal User. This functionality supports monitoring of access patterns, security auditing, and user activity tracking.

7.8.2.7 Traceability Report

7.8.2.7.1 Batch Movement Report - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Batch Movement Report in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the batch movement report process.

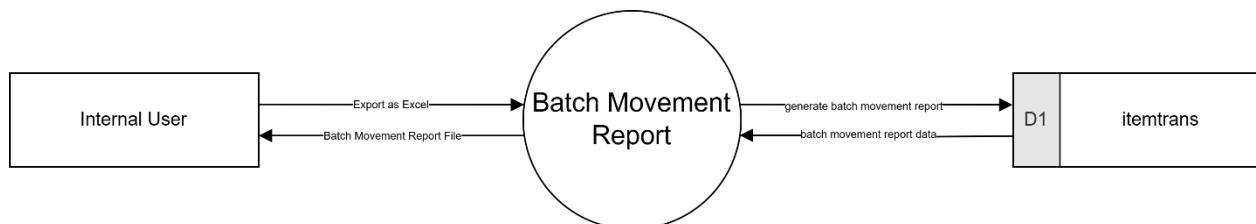


Figure 124. Batch Movement Report - Data Flow Diagram Level 0

This diagram outlines the Batch Movement Report process, where Internal Users export transaction data into batch reports. The system accesses item transaction records from the itemtrans data store, processes them, and generates a batch movement report, which is then delivered back to the user. This feature enables effective tracking and reporting of bulk inventory movements.

7.8.2.7.2 Traceability Report - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Traceability Report in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the traceability report process.



Figure 125. Report Preview - Data Flow Diagram Level 0

This diagram depicts the Traceability Report process. Internal Users define report parameters by selecting section, customer, period, and filters. The system processes these parameters and retrieves item transaction data from the itemtrans data store to generate a preview report. The resulting preview is then presented to the Internal User. This process helps in tracing item history and ensuring data integrity across transactions.

7.9 Resources

7.9.1 Resources

7.9.1.1 Resources - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Resources in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the resources process.

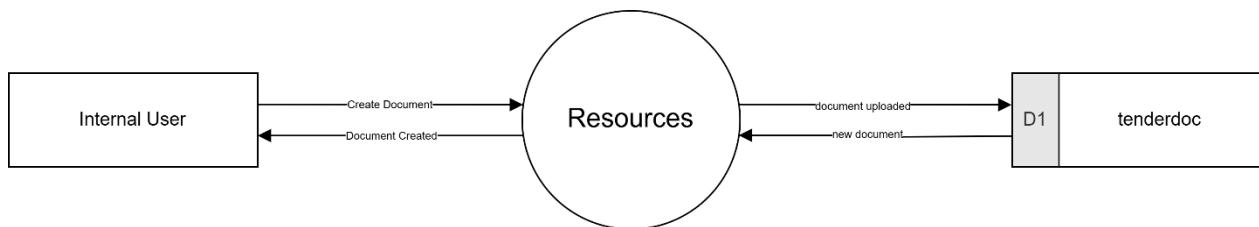


Figure 126. Resources - Data Flow Diagram Level 0

This diagram outlines the Resources document management process. Internal Users create documents which are then processed by the Resources system. Once uploaded, the system stores the document in the tenderdoc database and confirms successful creation back to the user.

The Resources process supports efficient internal documentation workflows by enabling creation, uploading, and storage of documents. It ensures that relevant documents are securely archived and accessible, promoting consistency and traceability in organizational documentation.

7.10 Global Mill Advisor

7.10.1 Mill Audit Request

7.10.1.1 Mill Audit Request - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Mill Audit Request in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the mill audit request process.

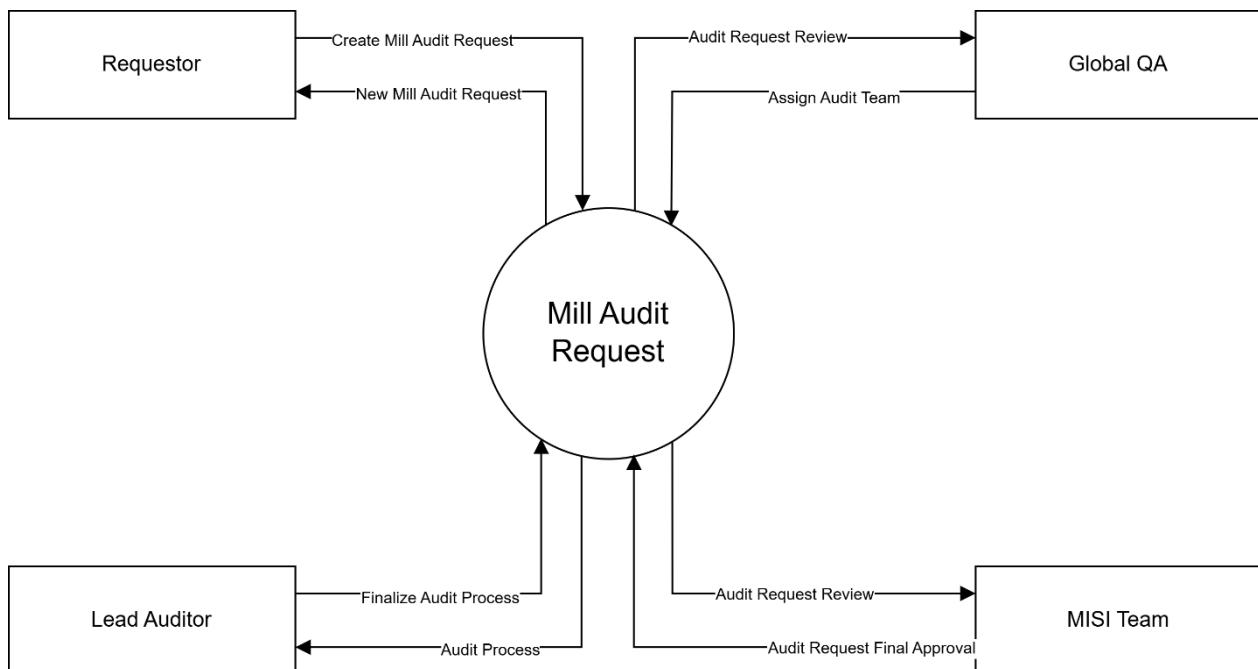


Figure 127. Mill Audit Request - Data Flow Diagram Level 0

This diagram illustrates the Mill Audit Request process, which involves coordination between multiple stakeholders to initiate and complete an audit. The process starts when a Requestor creates a Mill Audit Request. This request is then reviewed by both the Global QA and MISI Team. The Global QA assigns an appropriate audit team, while the MISI Team performs a review and gives final approval. The Lead Auditor is responsible for executing and finalizing the audit process.

The Mill Audit Request process ensures structured review, assignment, and completion of audits. It promotes accountability and transparency by involving key parties in the creation, approval, and execution phases, ultimately ensuring audits are conducted systematically and finalized with proper oversight.

7.10.1.2 Mill Audit Request - Data Flow Diagram Level 1

This Data Flow Diagram (DFD) level 1 visualizes the mill audit request process in Tubestream, depicting the flow of data between users, processes, and databases. It offers a more detailed view compared to the Level 0 diagram above. The image below presents the DFD for Tubestream's mill audit request process.

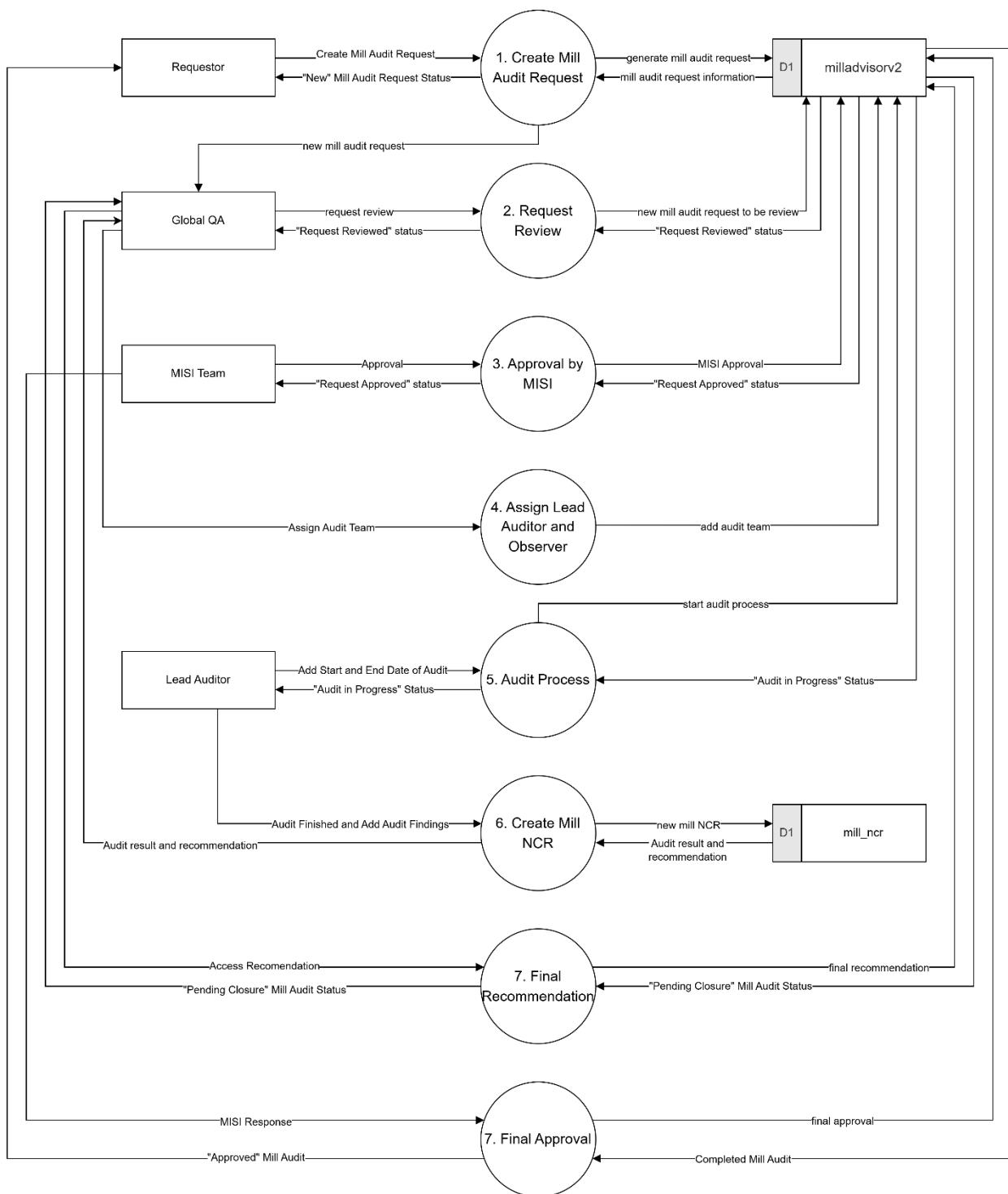


Figure 128. Mill Audit Request - Data Flow Diagram Level 1

The key components are explained on the table below.

Table 92. 3rd Mill Audit Request - Data Flow Diagram level 1 key components

No.	User	Input	Process	Output
1	Requestor	Audit request	Create Mill Audit Request	Request stored in milladvisorv2 (D1)
2	Global QA	Review request	Request Review	Reviewed status in milladvisorv2 (D1)
3	MISI Team	Approval decision	Approval by MISI	Approved status in milladvisorv2 (D1)
4	Global QA	Assign lead and observer	Assign Lead Auditor	Assigned team in milladvisorv2 (D1)
5	Lead Auditor	Start/end date, audit inputs	Audit Process	In-progress status in milladvisorv2 (D1)
6	Lead Auditor	Findings	Create Mill NCR	NCR stored in mill_ncr (D1)
7	Global QA	Review audit result	Final Recommendation	Recommendation in milladvisorv2 (D1)
8	MISI Team	Approval decision	Final Approval	Final approval in milladvisorv2 (D1)

7.10.2 Mill Listing

7.10.2.1 Mill Listing - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Mill Listing in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the mill listing process.

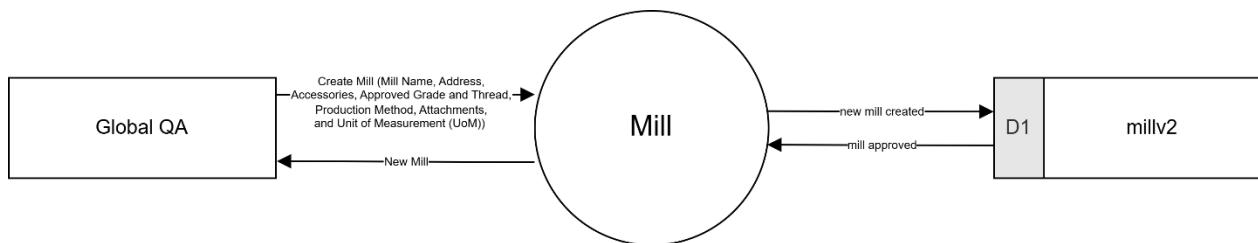


Figure 129. Mill Listing - Data Flow Diagram Level 0

This diagram illustrates the Mill creation process, initiated by the Global QA team. The Global QA provides all necessary mill details, including name, address, accessories, approved grade and thread, production method, attachments, and unit of measurement (UoM). Once this data is submitted, the system registers the new mill and stores it in the millv2 database. Confirmation of creation and approval is then communicated back to the system.

The Mill process ensures that all new mill records are created with complete and standardized information. It supports structured data entry, approval tracking, and centralized storage to enable traceability and compliance in the mill registration workflow.

7.10.3 Mill NCR

7.10.3.1 Mill NCR - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Audit Mill NCR in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the audit mill NCR process.

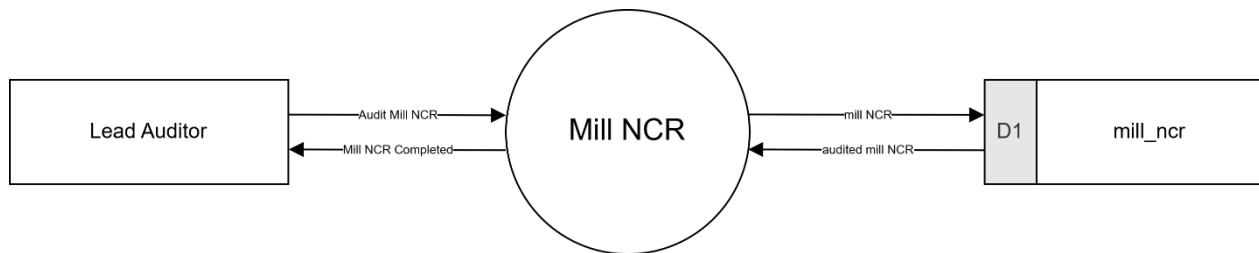


Figure 130. Mill NCR - Data Flow Diagram Level 0

This diagram represents the Mill NCR (Non-Conformance Report) process. The Lead Auditor initiates an NCR audit for a mill, and the system records the report and stores it in the mill_ncr database. After the audit is completed, the audited NCR is returned to the Lead Auditor for review and closure.

The Mill NCR process ensures non-conformities are properly identified, documented, and addressed. It enables thorough audit trails and accountability by managing NCR status from initiation through completion, with all data safely stored and retrievable.

7.10.4 Mill MoC

7.10.4.1 Mill MoC - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Mill MoC in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the mill MoC process.

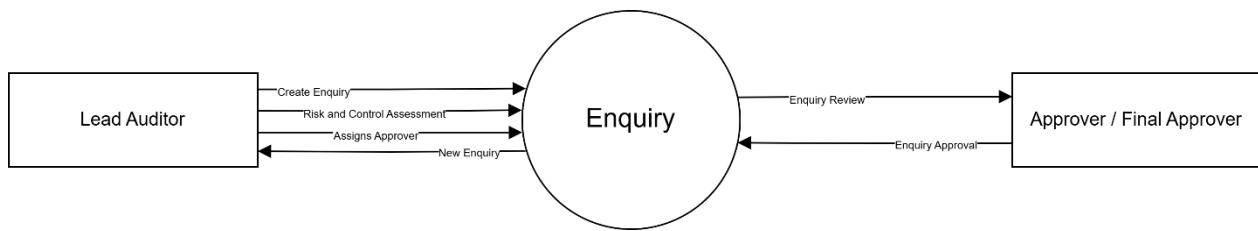


Figure 131. Mill MoC - Data Flow Diagram Level 0

This diagram describes the Enquiry process, initiated by the Lead Auditor. The auditor submits a new enquiry, performs a risk and control assessment, and assigns an approver. The Approver or Final Approver then reviews and either approves or rejects the enquiry. The outcome is communicated back to the Lead Auditor.

The Enquiry process allows structured risk management through formal enquiry submission and assessment. It ensures that appropriate stakeholders are involved in decision-making, promoting transparency, traceability, and compliance in the handling of audit-related enquiries.

7.10.4.2 Mill MoC - Data Flow Diagram Level 1

This Data Flow Diagram (DFD) level 1 visualizes the mill MoC process in Tubestream, depicting the flow of data between users, processes, and databases. It offers a more detailed view compared to the Level 0 diagram above. The image below presents the DFD for Tubestream's mill MoC process.

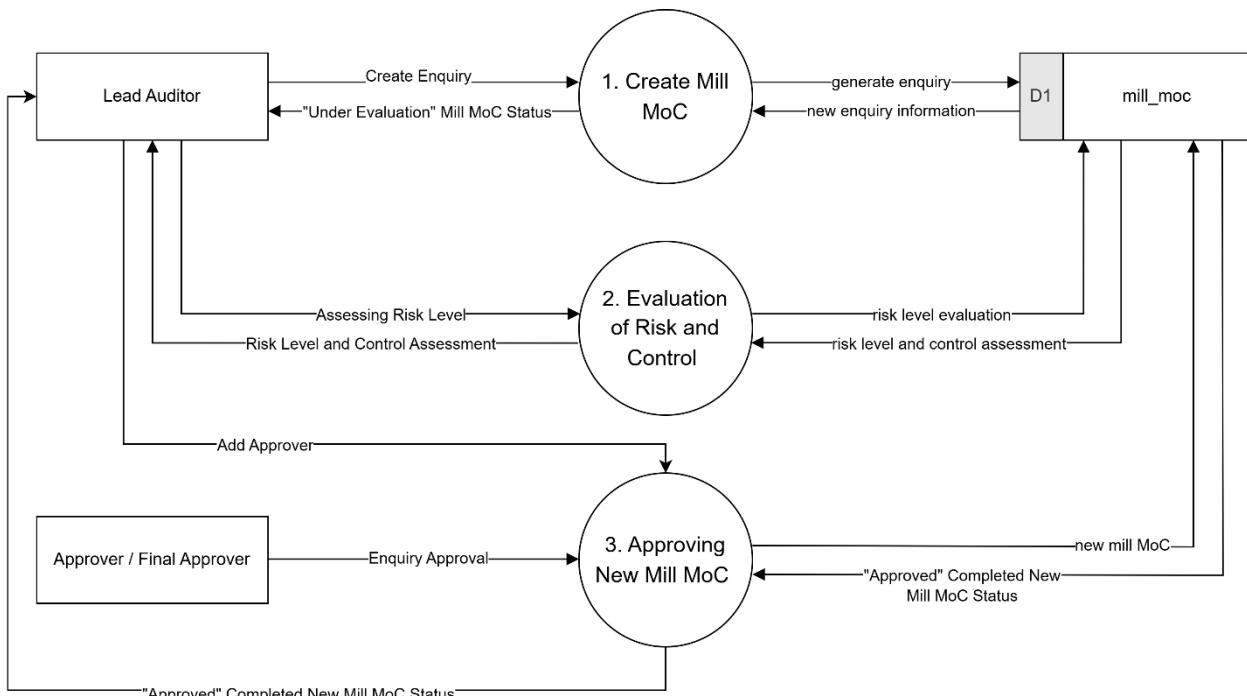


Figure 132. Mill MoC - Data Flow Diagram Level 1

The key components are explained on the table below.

Table 93. 3rd Mill MoC - Data Flow Diagram level 1 key components

No.	User	Input	Process	Output
1	Lead Auditor	Enquiry	Create Mill MoC	New MoC in mill_moc (D1)
2	Lead Auditor	Risk level and control info	Evaluation of Risk and Control	Evaluation in mill_moc (D1)
3	Approver	Approval input	Approving New Mill MoC	Approved MoC in mill_moc (D1)

7.11 Set Up

7.11.1 User

7.11.1.1 User - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of User in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the user process.

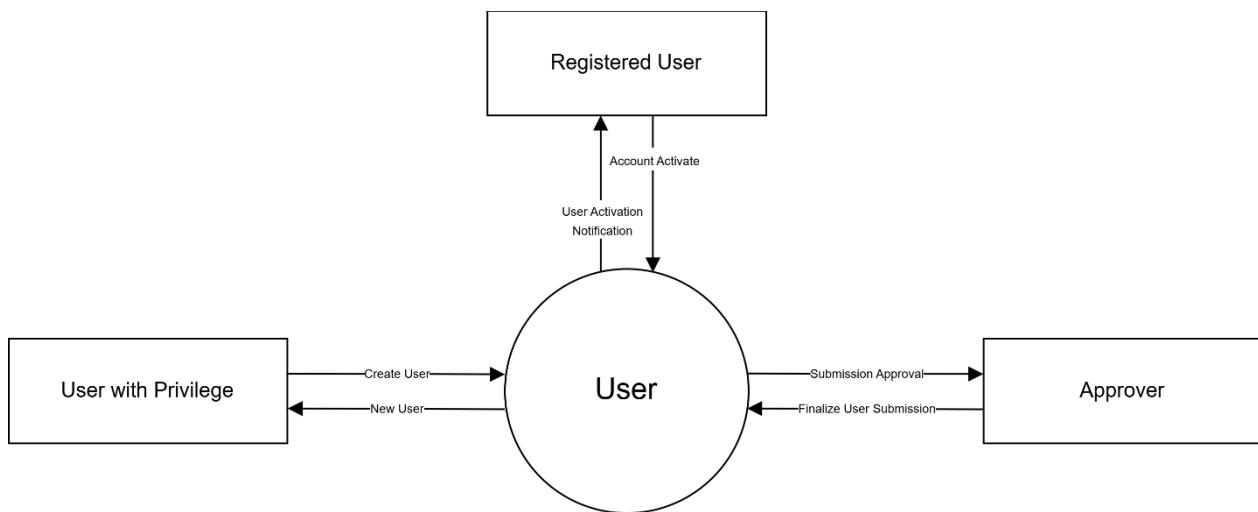


Figure 133. User - Data Flow Diagram Level 0

This diagram illustrates the user management process in the system, showing how data flows between various actors and the central User process. The Registered User begins the process by activating their account after receiving notification. The User with Privilege interacts with the system to create new users and manage user data. Once the user data is prepared, it is submitted for approval.

The Approver then reviews the submission and finalizes it. This process ensures that user registration, activation, and approval are handled systematically, involving appropriate roles to maintain data accuracy, access control, and compliance with organizational standards.

7.11.1.2 User - Data Flow Diagram Level 1

This Data Flow Diagram (DFD) level 1 visualizes the add user process in Tubestream, depicting the flow of data between users, processes, and databases. It offers a more detailed view compared to the Level 0 diagram above. The image below presents the DFD for Tubestream's add user process.

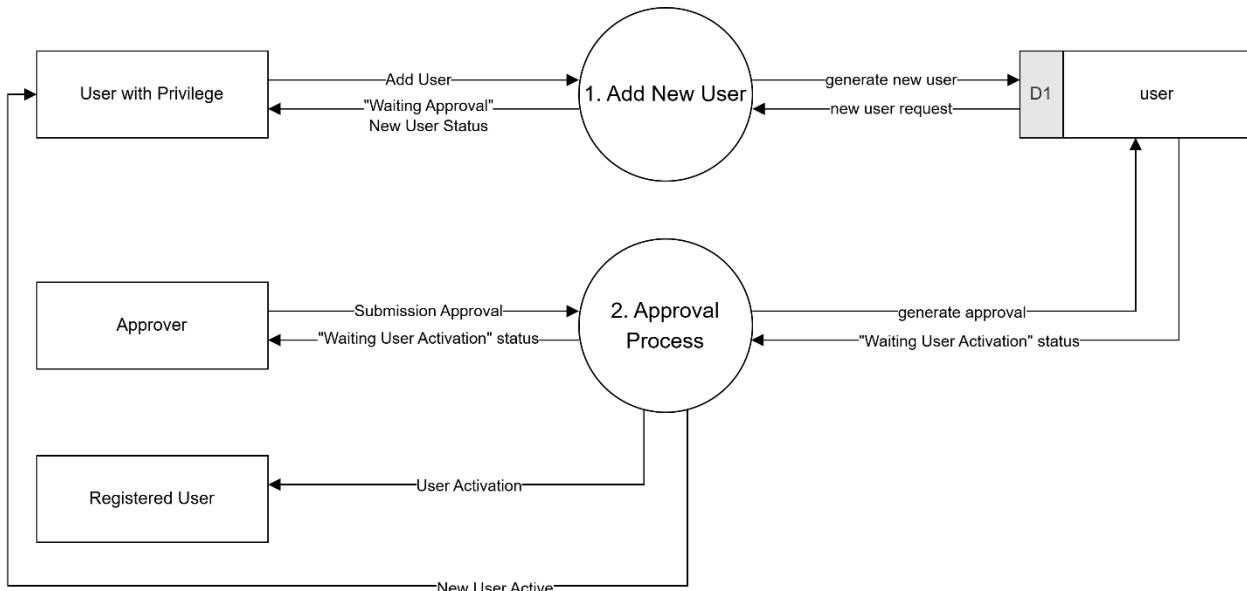


Figure 134. User - Data Flow Diagram Level 1

The key components are explained on the table below.

Table 94. User - Data Flow Diagram level 1 key components

No.	User	Input	Process	Output
1	User with Privilege	User details	<ul style="list-style-type: none"> • Add New User 	New user in user (D1)
2	Approver	Submission approval	Approval Process	Activated user in user (D1)

7.11.2 Privilege

7.11.2.1 Privilege - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Privilege in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the edit privilege process.

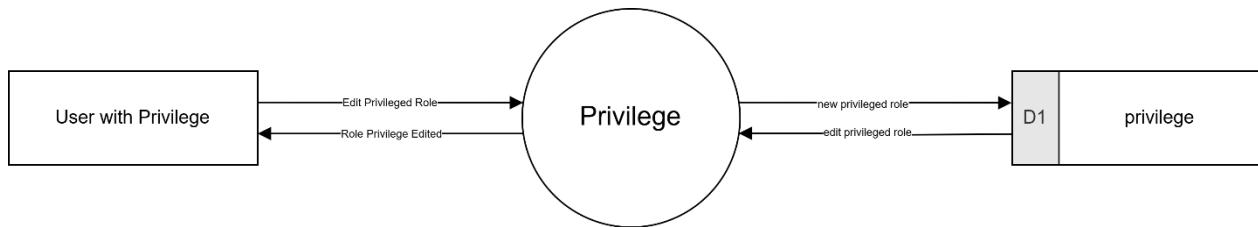


Figure 135. Privilege - Data Flow Diagram Level 0

This diagram illustrates the Privilege management process, showing how a user with privilege can modify roles within the system. The User with Privilege initiates the process by editing a privileged role, and after the system processes the request, confirmation is sent back indicating the role has been updated.

These updates are reflected in the privilege data store, where new or edited roles are stored to ensure that access control and permissions are properly maintained.

7.11.3 Tablet Access

7.11.3.1 Tablet Access - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Tablet Access in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the tablet access process.

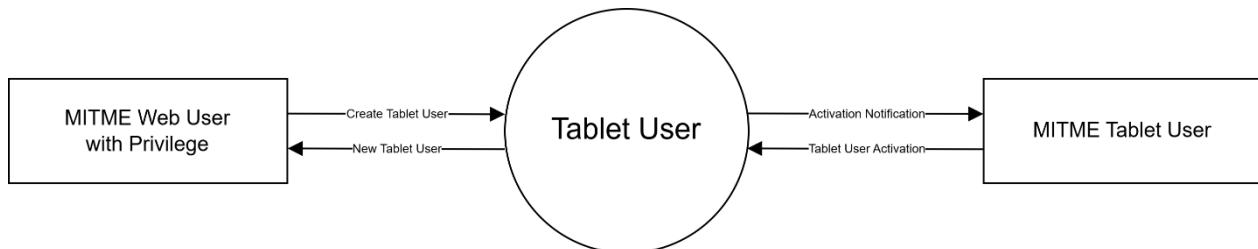


Figure 136. Tablet Access - Data Flow Diagram Level 0

This diagram describes the creation and activation process of a Tablet User within the MITME system. A MITME Web User with Privilege initiates the creation of a new Tablet User. The system sends an activation notification to the MITME Tablet User and awaits their activation. Once the user activates their access, confirmation is sent back, and the process is completed.

This ensures that Tablet Users are securely added and verified within the platform.

7.11.3.2 Tablet Access - Data Flow Diagram Level 1

This Data Flow Diagram (DFD) level 1 visualizes the create tablet access process in Tubestream, depicting the flow of data between users, processes, and databases. It offers a more detailed view compared to the Level 0 diagram above. The image below presents the DFD for Tubestream's tablet access process.

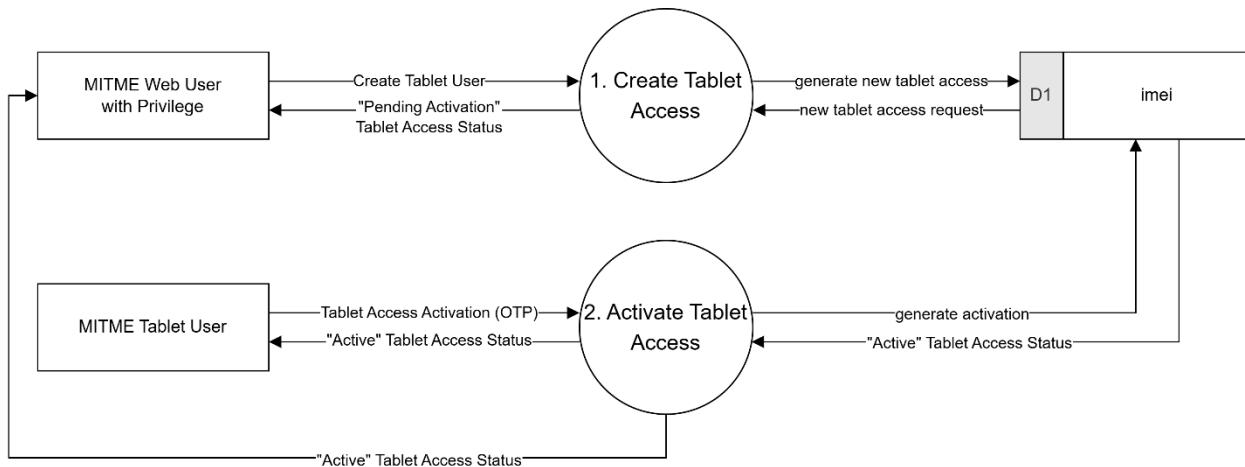


Figure 137. Tablet Access - Data Flow Diagram Level 1

The key components are explained on the table below.

Table 95. Tablet Access - Data Flow Diagram level 1 key components

No.	User	Input	Process	Output
1	MITME Web User	Tablet user info	Create Tablet Access	New access in imei (D1)
2	MITME Tablet User	OTP activation	Activate Tablet Access	Activated status in imei (D1)

7.11.4 Customer > Customer Info

7.11.4.1 Customer > Customer Info - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Customer Info in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the customer process.

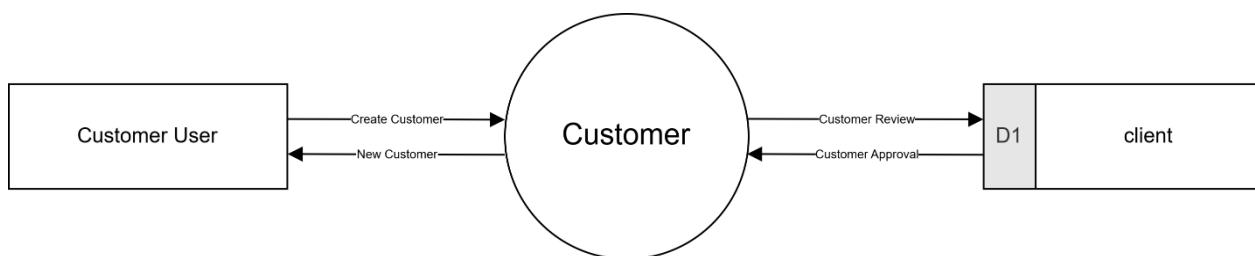


Figure 138. Customer > Customer Info - Data Flow Diagram Level 0

This diagram explains the Customer creation process within the system. A Customer User starts the process by submitting a request to create a new customer. The system then reviews the data and stores the customer details in the client data store after receiving approval. Once the process is complete, confirmation is returned to the Customer User.

This ensures that customer data is systematically verified, approved, and recorded in the database.

7.11.4.2 Customer > Customer Info - Data Flow Diagram Level 1

This Data Flow Diagram (DFD) level 1 visualizes the create customer process in Tubestream, depicting the flow of data between users, processes, and databases. It offers a more detailed view compared to the Level 0 diagram above. The image below presents the DFD for Tubestream's customer info process.

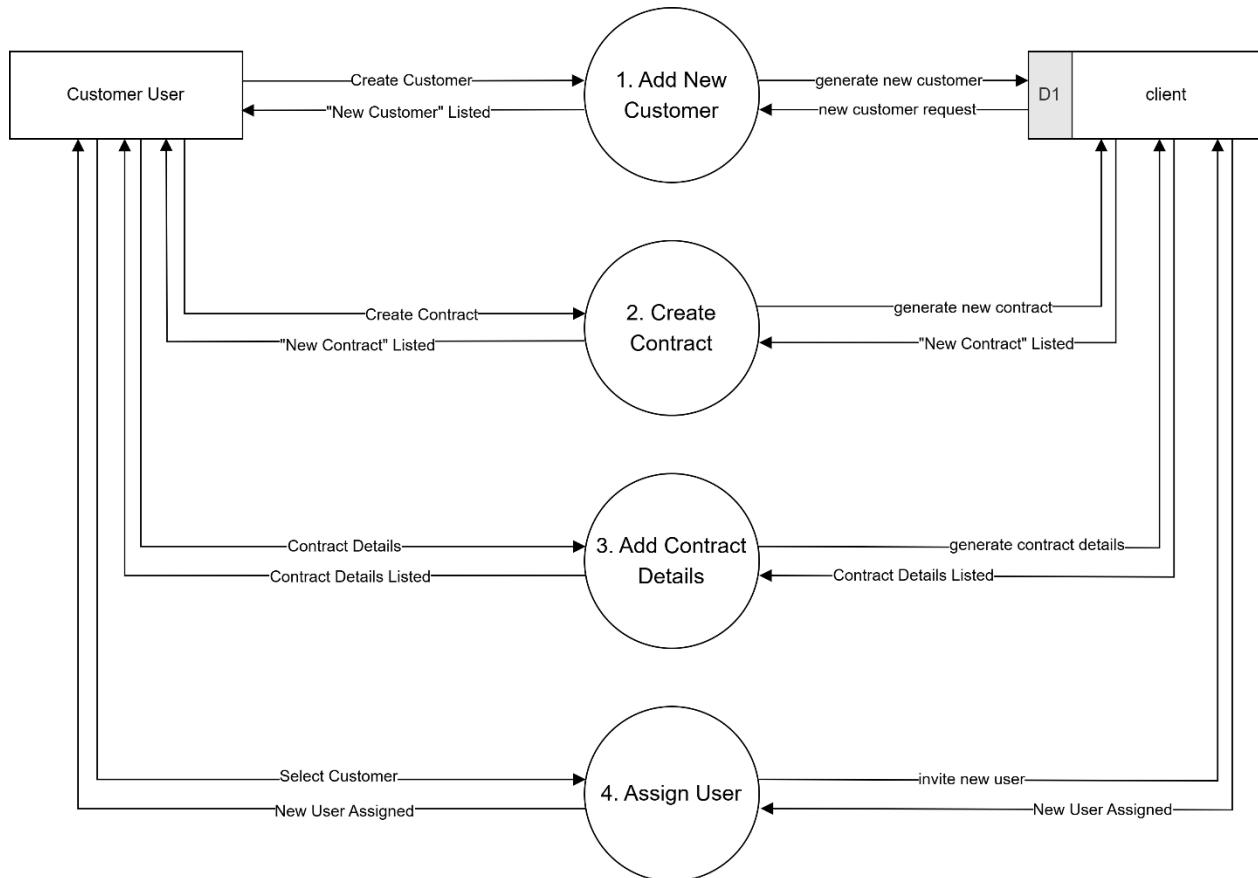


Figure 139. Customer > Customer Info - Data Flow Diagram Level 1

The key components are explained on the table below.

Table 96. Customer > Customer Info - Data Flow Diagram level 1 key components

No.	User	Input	Process	Output
1	Customer User	Customer info	Add New Customer	New customer in client (D1)
2	Customer User	Contract info	Create Contract	New contract in client (D1)
3	Customer User	Contract details	Add Contract Details	Contract detail in client (D1)
4	Customer User	Customer and user assignment	Assign User	Assigned user in client (D1)

7.11.5 Customer > Lot Price

7.11.5.1 Customer > Lot Price - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Lot Price in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the lot price process.

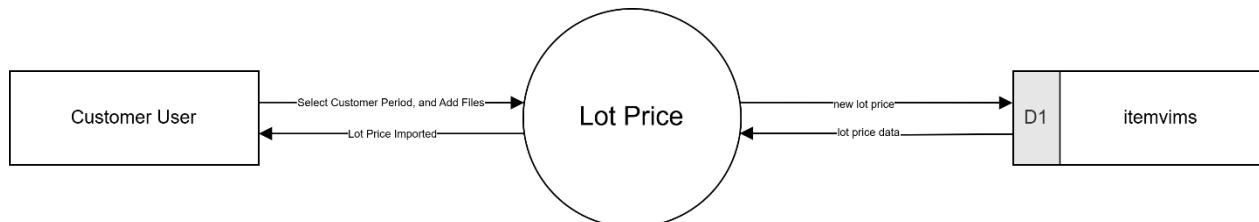


Figure 140. Customer > Lot Price - Data Flow Diagram Level 0

This diagram outlines the management of Lot Price data in the system. A Customer User begins the process by selecting a customer period and uploading related files. The system processes the uploaded data and updates or creates new entries in the itemvims data store. Once the lot price data is successfully handled, the system returns a confirmation to the user.

This process ensures that pricing information is accurately imported and managed.

7.11.6 Customer > Fees

7.11.6.1 Customer > Fees - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Fees in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the creation fee process.

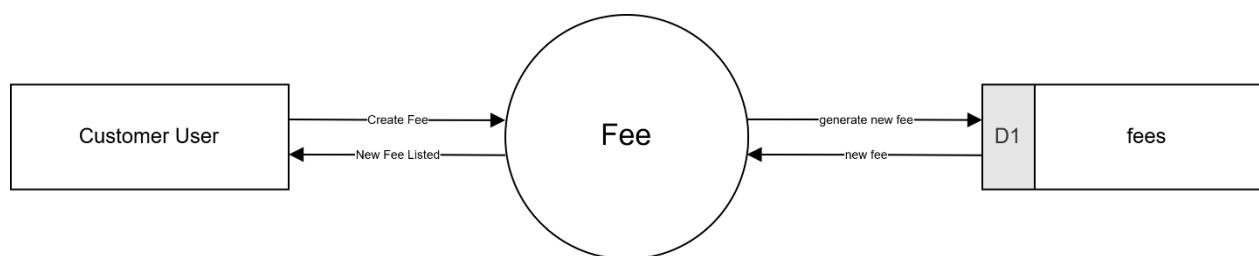


Figure 141. Customer > Fees - Data Flow Diagram Level 0

This diagram illustrates the process for managing fees within the system. A Customer User initiates the process by creating a new entry fee. The system generates the corresponding fee data and stores it in the fees data store. After the data is successfully saved, a notification is sent back to the Customer User confirming that the new fee has been listed.

This ensures the systematic and traceable creation of fee records.

7.11.7 Customer > Customer Item No.

7.11.7.1 Customer > Customer Item No. - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Customer Item No in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the customer item no process.



Figure 142. Customer > Customer Item No. - Data Flow Diagram Level 0

This diagram illustrates the management process for Customer Item Numbers. A Customer User begins by selecting the customer, item, item number, and consignment type. The system then processes this data and updates the item data store by creating or modifying customer item numbers. Once imported, confirmation is returned to the user.

This ensures customer-specific item tracking is accurately configured and stored.

7.11.8 Customer > Collection Point

7.11.8.1 Customer > Collection Point - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Collection Point in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the collection point process.

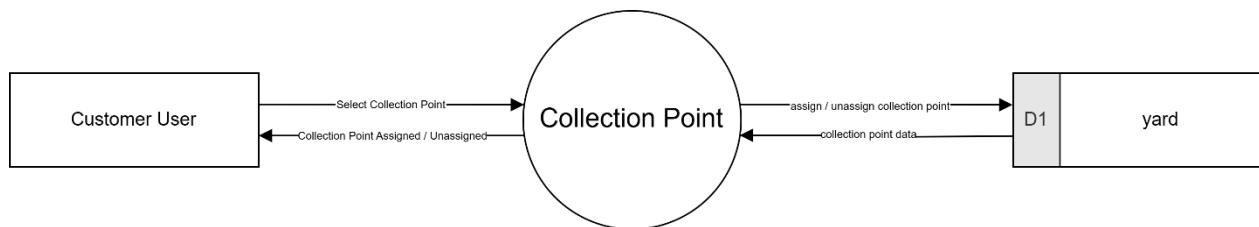


Figure 143. Customer > Collection Point - Data Flow Diagram Level 0

This diagram outlines the process for assigning or unassigning collection points. A Customer User initiates the action by selecting a collection point. The system updates the assignment in the yard data store and provides confirmation once the collection point has been successfully assigned or unassigned.

This ensures effective tracking and location management of collection points tied to customer operations.

7.11.9 Procurement > Contractor

7.11.9.1 Procurement > Contractor - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Contractor in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the contractor process.

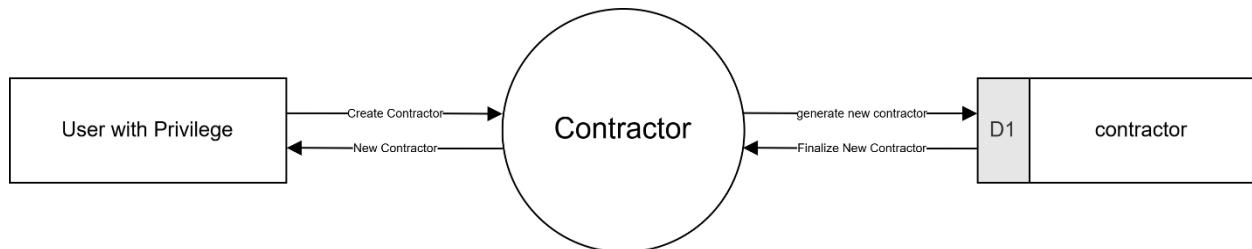


Figure 144. Procurement > Contractor - Data Flow Diagram Level 0

This diagram shows how the system manages contractor data. A User with Privilege initiates the creation of a new contractor. The system processes the request and stores the contractor information in the contractor data store. After finalization, confirmation is sent back to the user.

This process ensures that contractor records are securely created, updated, and maintained.

7.11.10 Procurement > Port

7.11.10.1 Procurement > Port - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Port in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the port process.

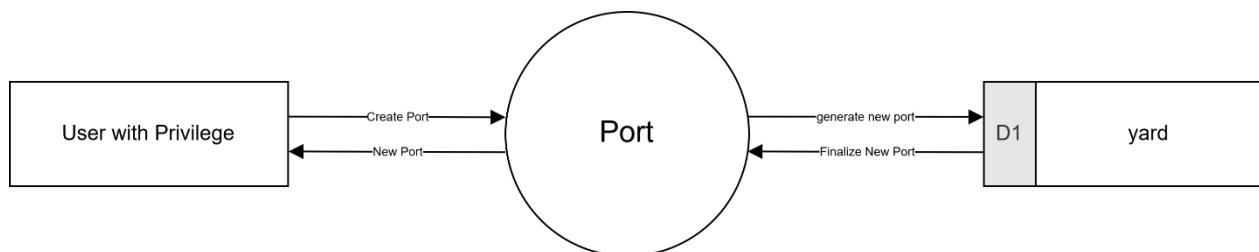


Figure 145. Procurement > Port - Data Flow Diagram Level 0

This diagram describes the creation process for ports within the system. A User with Privilege initiates the process by creating a new port. The system then generates and stores the port data in the yard data store. After successful finalization, the system sends confirmation back to the user.

This process enables accurate and structured management of port information related to yard operations.

7.11.11 Inventory > Item

7.11.11.1 Inventory > Item - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Item Inventory in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the item inventory process.

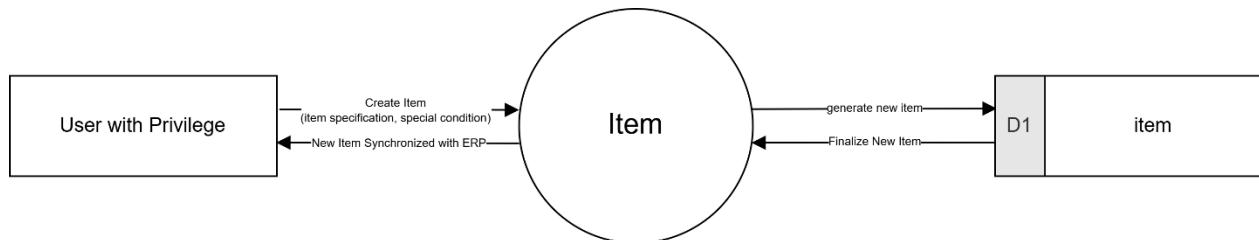


Figure 146. Inventory > Item - Data Flow Diagram Level 0

This diagram illustrates the Item creation process, showing how data flows between authorized users and the system. The process begins when a User with Privilege initiates item creation by inputting item specifications and special conditions. This information is processed and stored in the Item system, which also synchronizes new item data with the ERP system. The Item system then generates and finalizes a new item, sending it to the item database for storage and future retrieval.

The Item creation process supports efficient item management by enabling users with the necessary privileges to define, validate, and finalize new items. It ensures that all required item details are properly recorded and integrated with the enterprise system, maintaining consistency, traceability, and data accuracy throughout the item lifecycle.

7.11.12 Inventory > OD

7.11.12.1 Inventory > OD - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of OD in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the OD inventory process.

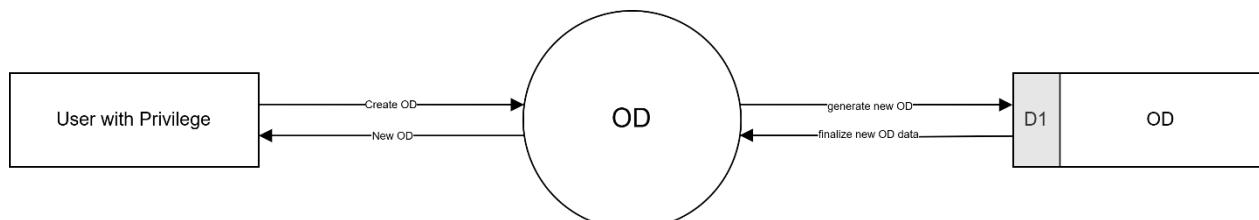


Figure 147. Inventory > OD - Data Flow Diagram Level 0

This diagram illustrates the OD creation process, showing how data flows between a privileged user and the system. The process begins when a User with Privilege creates a new OD, which is processed by the OD system. Once the data is validated and finalized, a new OD entry is generated and stored in the OD database. The system then returns the newly created OD data to the user for confirmation or further actions.

The OD process ensures that order documents are accurately created, reviewed, and stored within the system. It supports streamlined document management by enabling authorized users to initiate and complete OD entries while maintaining data consistency and traceability.

7.11.13 Inventory > Grade

7.11.13.1 Inventory > Grade - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Grade in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the grade inventory process.

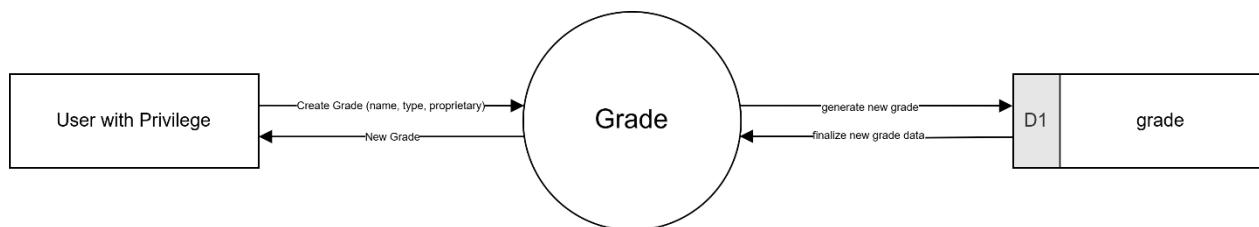


Figure 148. Inventory > Grade - Data Flow Diagram Level 0

This diagram describes the Grade management process, where a User with Privilege initiates the creation of a new grade by providing essential details such as name, type, and proprietary status. The Grade system processes this input, finalizes the data, and generates a new grade, which is stored in the grade database. The completed grade information is then made available to the user.

The Grade process facilitates structured classification of product or material quality by allowing privileged users to define and manage grade attributes efficiently. This helps maintain standardization and accurate referencing across the system.

7.11.14 Inventory > Connection

7.11.14.1 Inventory > Connection - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Connection Inventory in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the connection inventory process.

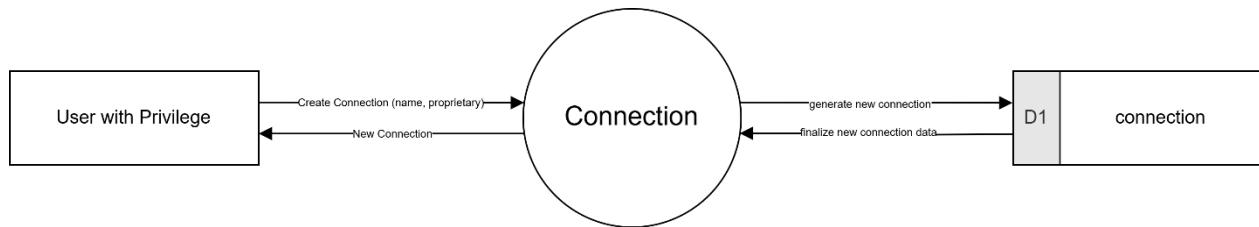


Figure 149. Inventory > Connection - Data Flow Diagram Level 0

This diagram outlines the Connection setup process within the system. A User with Privilege initiates the process by creating a new connection, including details such as

its name and whether it is proprietary. The Connection system processes the input, generates a new connection entry, and stores it in the connection database. The finalized connection data is then returned to the user.

The Connection process supports the establishment and tracking of various system or data relationships, ensuring that all connections are properly defined, stored, and retrievable for operational or integration purposes.

7.11.15 Inventory > Special Condition

7.11.15.1 Inventory > Special Condition - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Special Condition Inventory in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the special condition inventory process.

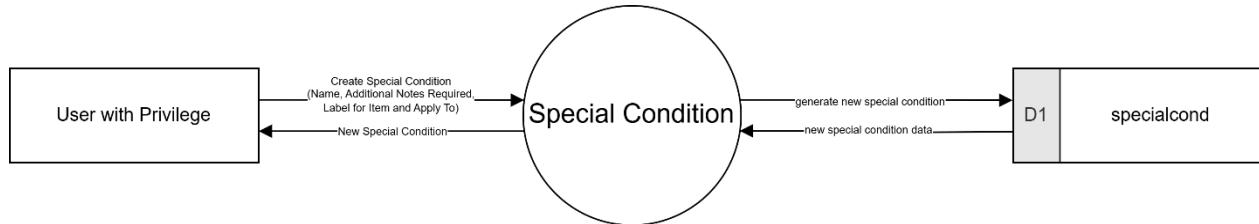


Figure 150. Inventory > Special Condition - Data Flow Diagram Level 0

This diagram shows the creation flow for a Special Condition within the system. A User with Privilege initiates the process by providing key details, including the name, any additional notes required, label for the item, and its application. The Special Condition system processes the input, finalizes the data, and stores the new special condition in the specialcond database. The newly created special condition is then sent back to the user.

The Special Condition process enables the accurate documentation and handling of exceptions or unique requirements related to items. It ensures that special handling instructions are properly captured and integrated, supporting better compliance and operational clarity.

7.11.16 Inventory > Yard

7.11.16.1 Inventory > Yard - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Yard Inventory in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the yard inventory process.

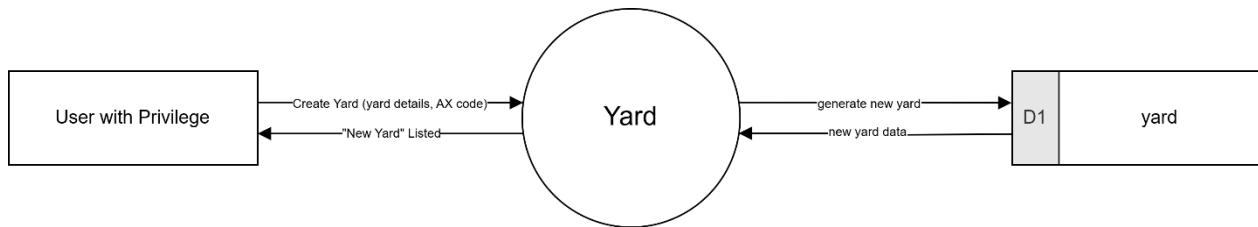


Figure 151. Inventory > Yard - Data Flow Diagram Level 0

This diagram illustrates the Yard management process, detailing how a User with Privilege can create a new yard by inputting relevant yard details, including the AX code. The Yard system processes this input, finalizes the yard data, and stores the new record in the yard database. Once stored, the system returns confirmation of the newly listed yard to the user.

The Yard process supports structured registration and management of yard facilities. It ensures that each yard is accurately documented and available in the system for future operations and referencing.

7.11.16.2 Inventory > Yard - Data Flow Diagram Level 1

This Data Flow Diagram (DFD) level 1 visualizes the yard inventory process in Tubestream, depicting the flow of data between users, processes, and databases. It offers a more detailed view compared to the Level 0 diagram above. The image below presents the DFD for Tubestream's yard inventory process.

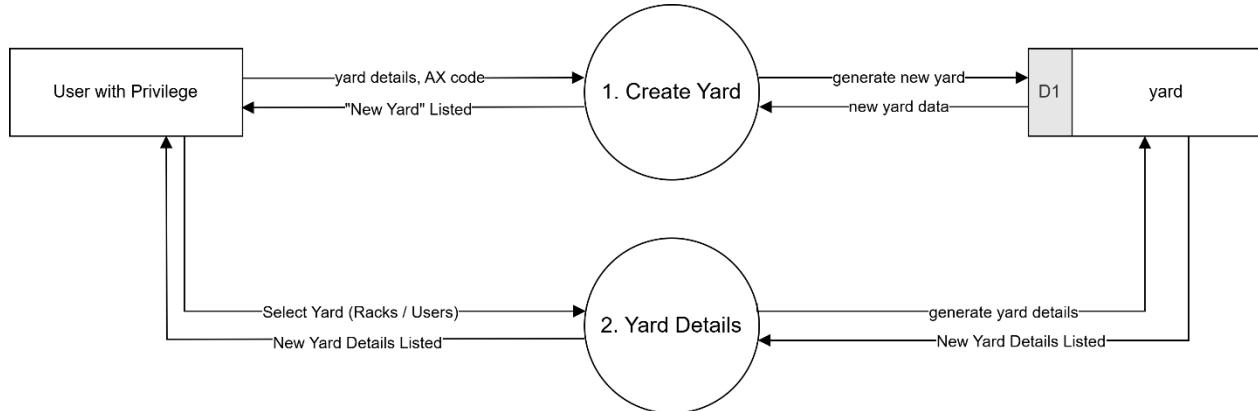


Figure 152. Inventory > Yard - Data Flow Diagram Level 1

The key components are explained on the table below.

Table 97. Inventory > Yard - Data Flow Diagram level 1 key components

No.	User	Input	Process	Output
1	User with Privilege	Yard details, AX code	Create Yard	New yard in yard (D1)
2	User with Privilege	Select racks/users	Yard Details	Yard detail in yard (D1)

7.11.17 Inventory > Package

7.11.17.1 Inventory > Package - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Package Inventory in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the package inventory process.

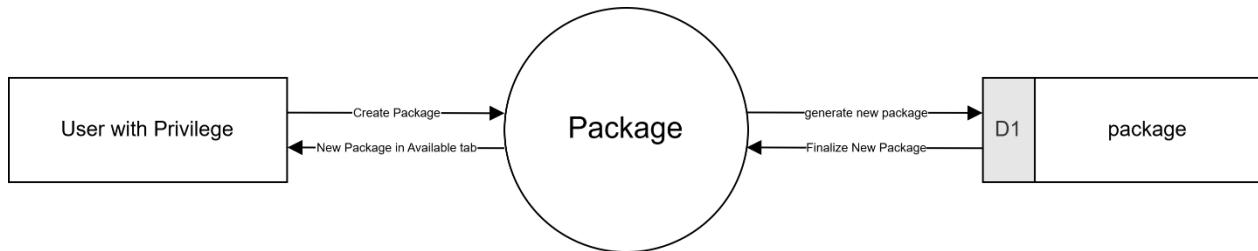


Figure 153. Inventory > Package - Data Flow Diagram Level 0

This diagram explains the Package creation process, where a User with Privilege initiates a new package entry. The Package system processes the creation request, generates and finalizes the new package, and stores it in the package database. Once the entry is successfully stored, the system updates the user by listing the new package under the Available tab.

The Package process facilitates the efficient organization and availability tracking of packages within the system, ensuring clear visibility and access to new package records.

7.11.18 Quality & HSE > Category Customization

7.11.18.1 Quality & HSE > Category Customization - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Category Customization in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the category customization process.

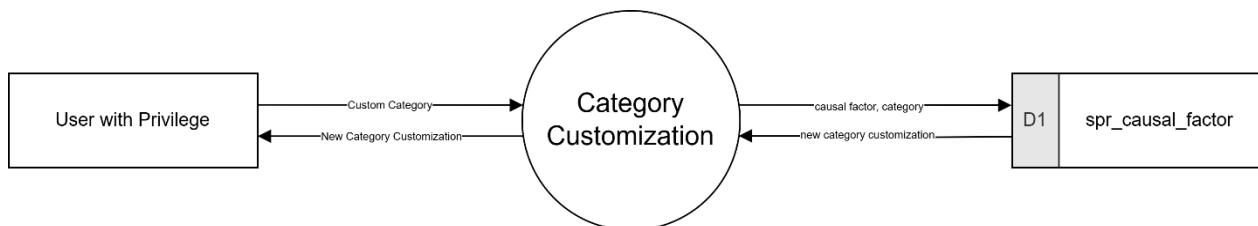


Figure 154. Quality & HSE > Category Customization - Data Flow Diagram Level 0

This diagram illustrates the Category Customization process, where a User with Privilege defines a custom category by submitting details such as causal factors and category names. The Category Customization system processes the request, stores the new configuration in the spr_causal_factor database, and returns the finalized customization back to the user.

The Category Customization process enables personalized categorization based on organizational needs, enhancing flexibility and relevance in reporting and classification activities.

7.12 Support

7.12.1 Support - Data Flow Diagram Level 0

This image represents a Level 0 Data Flow Diagram (DFD) for the main process of Support in Tubestream. It outlines the key interactions between users and the system, showing how data flows between entities and the support process.

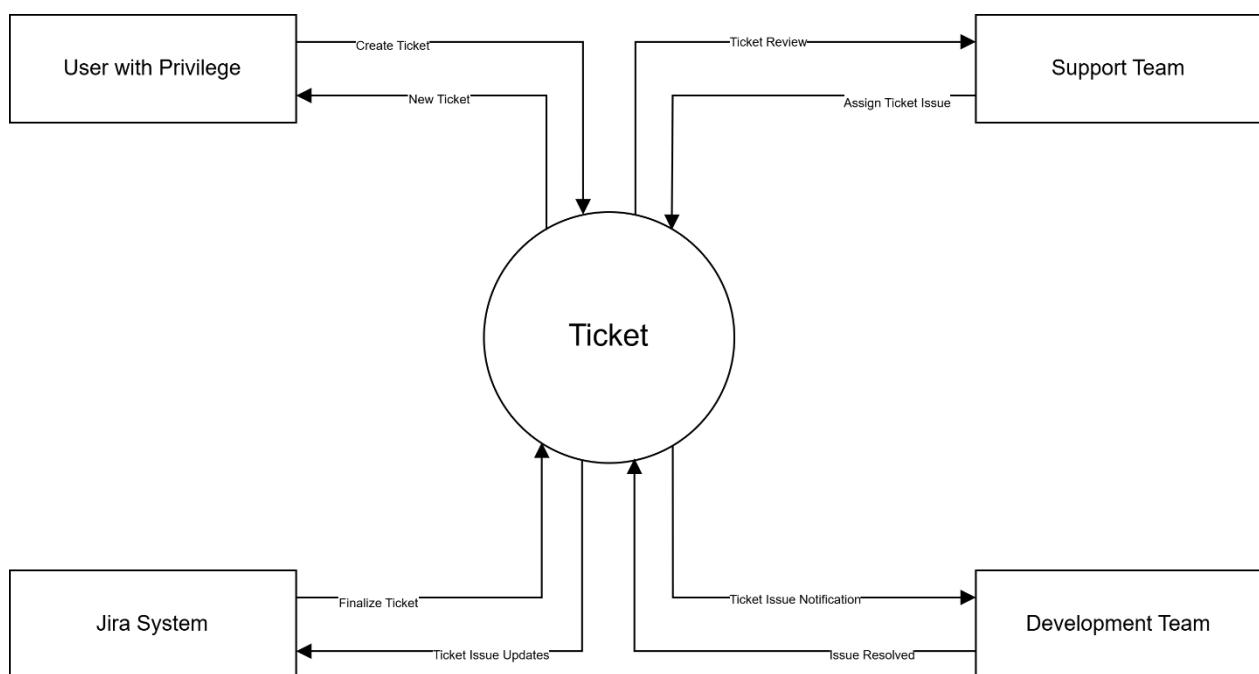


Figure 155. Support - Data Flow Diagram Level 0

This diagram outlines the Ticket management process, starting when a User with Privilege creates a ticket. The ticket is reviewed by the Support Team, who may assign it to the Development Team. Once resolved, the Development Team notifies the system. Ticket updates are continuously tracked and pushed to the Jira System for finalization and reporting.

The Ticket process facilitates streamlined issue tracking and resolution across departments. It ensures timely follow-up and collaboration between users, support staff, developers, and the Jira system, creating a closed-loop workflow for managing technical and operational issues.

7.12.2 Support - Data Flow Diagram Level 1

This Data Flow Diagram (DFD) level 1 visualizes the create ticket process in Tubestream, depicting the flow of data between users, processes, and databases. It offers a more detailed view compared to the Level 0 diagram above. The image below presents the DFD for Tubestream's support process.

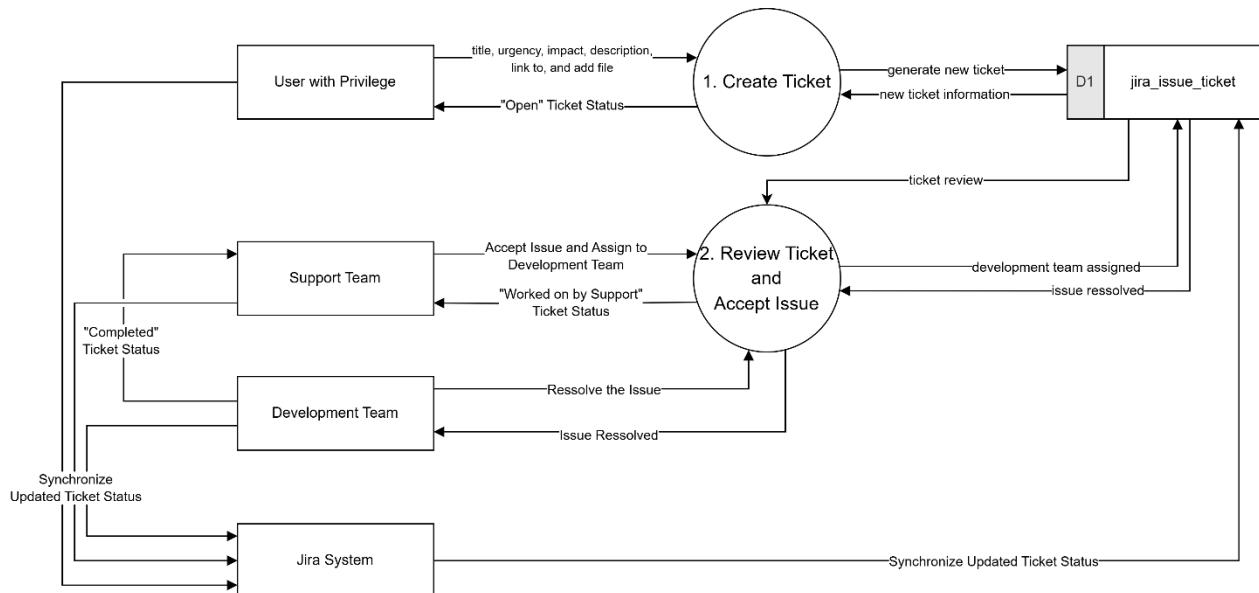


Figure 156. Support - Data Flow Diagram Level 1

The key components are explained on the table below.

Table 98. Support - Data Flow Diagram level 1 key components

No.	User	Input	Process	Output
1	User with Privilege	Ticket info, description	Create Ticket	New ticket in jira_issue_ticket (D1)
2	Support Team	Ticket acceptance	Review Ticket and Accept	Assigned issue in jira_issue_ticket (D1)
3	Development Team	Issue resolution	Resolve Issue	Ticket status updated in jira_issue_ticket (D1)

8. Database Collection

In Tubestream, the database is built using MongoDB, a NoSQL document-based database that stores data in JSON-like BSON format. Instead of traditional tables, MongoDB organizes data into collections containing flexible documents. Below is a list of key collections in Tubestream and their descriptions:

8.1 abbreviation

Stores standardized short forms of terms or phrases used across the system.

Table 99. abbreviations table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
desc	<i>array</i>	Description of abbreviation
name	<i>string</i>	Abbreviation code/name
type	<i>integer</i>	Type of abbreviation

8.2 actionlist

Contains predefined or dynamic actions related to specific workflows or processes.

Table 100. actionlist table

Field	Data Type	Description
_id	<i>ObjectId</i>	Unique identifier
revno	<i>string</i>	Revision number
revised	<i>string</i>	Revision status indicator
fname	<i>string</i>	File name
docno	<i>string</i>	Document number
owner	<i>string</i>	Owner ID
folder	<i>string</i>	Folder ID
desc	<i>string</i>	Description of document
contract	<i>string</i>	Contract ID
docdate	<i>int</i>	Document date (UNIX timestamp)
cout	<i>int, null</i>	Check-out timestamp

Field	Data Type	Description
cin	<i>int, null</i>	Check-in timestamp
owner_rev	<i>string</i>	Owner of revision
lastpostdate	<i>int</i>	Last posted timestamp
lastpostwho	<i>string</i>	Last posted by user

8.3 assign_evaluators

Contains evaluator assignment mappings for processes like MOC or audits, defining who is responsible for evaluations.

Table 101. assign_evaluators table

Field	Data Type	Description
_id	<i>string</i>	Evaluator assignment ID
person	<i>string</i>	ID of evaluator
role	<i>string</i>	Role (e.g., "Evaluator")
department	<i>string</i>	Department of evaluator
status	<i>int</i>	Status of evaluator submission
due_date	<i>int</i>	Evaluation due timestamp
date_submitted	<i>int</i>	When it was submitted
assigned_by	<i>string</i>	Who assigned the evaluator
accepted.user_id	<i>string/null</i>	User ID if accepted
accepted.date_accepted	<i>int/null</i>	Timestamp if accepted
rejected.user_id	<i>string/null</i>	Rejection info
evaluation.<area>.type	<i>string</i>	Area-specific risk type (e.g., schedule, quality)

Field	Data Type	Description
evaluation.<area>.description	<i>string</i>	Area description
evaluation.<area>.risk	<i>int</i>	Risk score
evaluation.status	<i>int</i>	Overall evaluation status
evaluation.action_owners	<i>array of objects</i>	Action items and owners

8.4 auditschedule

Holds scheduled audit activities including dates, scopes, and responsible parties.

Table 102. auditschedule table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
report_start	<i>int</i>	Report start date
report_end	<i>int</i>	Report end date
start	<i>int</i>	Audit start date
end	<i>int</i>	Audit end date
auditor	<i>string</i>	Auditor name
audit_scope	<i>string</i>	Scope of the audit
contractor	<i>string</i>	Contractor ID
location	<i>string</i>	Audit location
mill_id	<i>string</i>	Mill identifier
type	<i>string</i>	Audit type
comment	<i>string</i>	Audit comment
summary	<i>string</i>	Audit summary

Field	Data Type	Description
rating	<i>string, null</i>	Audit rating
title	<i>null</i>	Audit title
auditor_name	<i>null</i>	Auditor's full name
filename	<i>Array, null</i>	Associated file names
report_filename	<i>Array</i>	Report file names
is_from_excel	<i>string</i>	Data source flag (excel/manual)

8.5 ax_soh

Stores stock-on-hand data extracted from AX (ERP) for inventory monitoring.

Table 103. ax_soh table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
item_number	<i>string</i>	Item number
item_name	<i>string</i>	Item name
batch	<i>string</i>	Batch number
vendor_name	<i>string</i>	Vendor name
warehouse	<i>string</i>	Warehouse ID
allocation	<i>string</i>	Allocation name
mill	<i>string</i>	Mill name
po_date	<i>long, int</i>	Purchase order date
date_received	<i>long, int</i>	Date item received
po_status	<i>string</i>	Purchase order status

Field	Data Type	Description
mt	<i>string</i>	Metric tons
qty	<i>string</i>	Quantity
total_value	<i>string</i>	Total value
port	<i>string</i>	Port of arrival
well	<i>string</i>	Well name
delivery_terms	<i>string</i>	Delivery terms
location	<i>string</i>	Location
customer_po	<i>string</i>	Customer purchase order
lastpostdate	<i>int</i>	Last updated timestamp
isimport	<i>bool</i>	Import status
source	<i>string</i>	Source location
month	<i>int</i>	Month of transaction
year	<i>int</i>	Year of transaction
book_value	<i>double</i>	Book value
book_value_mt	<i>double, int, bool</i>	Book value per metric ton

8.6 axlog

Logs transaction or activity events from the AX system for audit purposes.

Table 104. axlog table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
status	<i>string</i>	Import status

Field	Data Type	Description
id_log	<i>string</i>	Log ID
filename	<i>string</i>	File name
last_import	<i>long, int</i>	Last import timestamp
last_missed	<i>null</i>	Last missed timestamp (if any)

8.7 azure_log

Contains logs from Azure-based services or integrations.

Table 105. azure_log table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
whenupdate	<i>int</i>	Last update timestamp
whoupdate	<i>string</i>	Last updated by

8.8 batch_trackability_itemcode_matrix

Maps item codes to batch traceability rules or constraints.

Table 106. batch_trackability_itemcode_matrix table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
item_code	<i>string</i>	Item code
is_tracked	<i>bool</i>	Tracking flag

8.9 batch_trackability_matrix

Defines batch-level traceability configurations for items.

Table 107. batch_trackability_matrix table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
item_type	<i>string</i>	Type of item
customer	<i>string</i>	Customer ID
is_tracked	<i>bool</i>	Tracking flag

8.10 calloff_counter

Stores sequential numbers for call-off processes.

Table 108. calloff_counter table

Field	Data Type	Description
_id	<i>ObjectId</i>	Unique identifier
year	<i>int</i>	Year of call-off
seq	<i>long</i>	Sequence number

8.11 Client

Contains client or customer master data.

Table 109. Client table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
customer_findim	<i>Array</i>	List of financial dimension details linked to the customer
insp	<i>object</i>	Inspection-related information
location	<i>string</i>	Location of the client

Field	Data Type	Description
collection_point	Array	Points where goods/services are collected
credit	object	Credit details including limits, expiry, etc.
region	null, string	Region associated with the client
subcontype	string	Subcontractor type
validate	bool	Whether the client data is validated
mill	Array	Associated mills
hasbu	string	Indicator if the client has business units
contract_code	string	Code referencing the contract
full_name	string	Full legal name of the client
is_lowest	string	Indicator if the client is the lowest in a hierarchy
parent	string	Parent client ID
leadtime	int, object, null	Delivery or response lead time
status	int	Status of the client (e.g. active, inactive)
contact_status	string	Contact relationship status
level	double	Hierarchy level of the client
lastpostdate	null, int	Timestamp of last update
pic	string	Person in charge
deliveryterm	null, string	Delivery terms
mb_gbc	int	Possibly business classification metric
enable_kpi	bool	Whether KPIs are enabled for this client
name	string	Display name of the client

Field	Data Type	Description
bu	Array	Business units associated
tag	Array	Tags or labels
code	string	Internal or reference code
is_contract	bool	Whether the client is under contract
import	bool	Indicates if the client is involved in import
id_erp	string	ERP identifier
millclass	int	Mill classification
tabs	Array	UI or data grouping tabs
address	Array, object	Address(es) of the client
type	string	Type of client (e.g. customer, partner)
is_vims	bool	Indicates use of the VIMS system
limit_data	int, null	Data access or usage limit
cust_ref	string	Customer reference
mb_category	null, string	MB category (e.g. sales class)
lastpostwho	null, string	Last person who updated the record
credit_limit_required	bool	Whether a credit limit is mandatory
legal_name	string	Legal registered name of the client
mb_sc_type	string	Supply chain type under MB classification
avg_price	null, string	Average pricing data
customer_address	Array	Customer-specific address list
shortname	null, string	Short name or alias

Field	Data Type	Description
skip_credit_limit	<i>bool</i>	Whether to bypass credit limit enforcement
mb_inv_type	<i>null, string</i>	Inventory type under MB category
accountno	<i>string</i>	Account number of the client
project_code	<i>string</i>	Project code linked to the client
mb_sc_d365_month	<i>int</i>	MB config for Sales Commitment Calculation (in months)
is_active_project	<i>bool</i>	Whether the project is actively managed

8.12 commitment

Tracks committed stock, services, or deliverables.

Table 110. commitment table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
customer	<i>string</i>	Customer name or ID
pono	<i>string</i>	Purchase order number
contract_code	<i>string</i>	Contract code
pic	<i>string</i>	Person in charge
appv	<i>string</i>	Approver ID
notes	<i>string</i>	Notes
approve	<i>string</i>	Approval status
files	<i>Array</i>	Attached file(s) information
updatedate	<i>int, double</i>	Last update timestamp
updatewho	<i>string</i>	Last updated by

Field	Data Type	Description
approvedate	<i>int, double</i>	Date of approval
item	Array	Items in commitment
closeReason	<i>string</i>	Reason for closing
closedate	<i>int, double</i>	Close date
closewho	<i>string</i>	Closed by
createid	<i>string</i>	Created by
createdate	<i>int, double</i>	Creation date
customer_parent	<i>string</i>	Parent customer
auto_generate	<i>bool</i>	Flag for auto generation
already_approve	<i>bool</i>	Approval status
isAutoApprove	<i>bool</i>	Auto approval flag
item_old	Array	Previous item list
closefiles	Array	Files attached on close
totalPrice	<i>double</i>	Total price
totalMT	<i>double</i>	Total metric tons
cncrReason	<i>null</i>	Cancel reason
canceldate	<i>null</i>	Cancel date
appvrove	<i>string</i>	

8.13 config

General configuration settings are used across the system.

Table 111. config table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
md	<i>string</i>	Email address of responsible
spr_insurance_pic	<i>string</i>	Insurance PIC ID
value_literal	<i>string</i>	Literal value
value	<i>string, int</i>	Config value
updated_at	<i>string, Date, int</i>	Last update timestamp

8.14 configuration

Stores specific configuration details or parameters.

Table 112. configuration table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
type	<i>string</i>	Configuration type
privilege	<i>Array</i>	List of privileges
value_min	<i>int, null</i>	Minimum value
value_max	<i>int, null</i>	Maximum value
month	<i>double</i>	Configuration month
year	<i>double</i>	Configuration year

8.15 connection

Defines system or integration connection parameters.

Table 113. connection table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
name	<i>string</i>	Connection name
code	<i>string</i>	Connection code
type	<i>string</i>	Connection type
prop	<i>string</i>	Proprietary Name
desc	<i>string, null</i>	Description
shareto	Array	Shared with
lastpostdate	<i>int</i>	Last update timestamp
lastpostwho	<i>string</i>	Last updated by

8.16 contract

Holds contract master data and metadata.

Table 114. contract table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
contract_name	<i>string</i>	Contract name
start_date	<i>int</i>	Contract start date
end_date	<i>int</i>	Contract end date
client	<i>string</i>	Client name
commitment_po	<i>string</i>	Commitment purchase order

Field	Data Type	Description
active	<i>string</i>	Contract active status
lastpostdate	<i>int</i>	Last updated date
lastpostwho	<i>string</i>	Last updated by
shareto	<i>Array</i>	Shared with
address	<i>null</i>	Contract address
allowance	<i>null</i>	Allowance value
contact	<i>null</i>	Contact person
contract_code	<i>null</i>	Contract code
currency	<i>null</i>	Currency type
currency_amount	<i>null</i>	Contract amount
customer	<i>string</i>	Customer name
custreff	<i>null</i>	Customer reference
is_draft	<i>bool</i>	Draft flag
pricing_model	<i>null</i>	Pricing model
pricing_option	<i>null</i>	Pricing option
tod	<i>null</i>	Terms of delivery

8.17 contract_doc

Stores documents associated with contracts.

Table 115. contract_doc table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier

Field	Data Type	Description
doc_type	<i>string</i>	Document type
filename	<i>string</i>	File name
id_contract	<i>string</i>	Related contract ID
path	<i>string</i>	File path
status	<i>string</i>	Document status
username	<i>string</i>	User who uploaded document
when	<i>int</i>	Upload timestamp
who	<i>string</i>	Additional person involved

8.18 contractor

Contains contractor company or individual records.

Table 116. contractor table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
name	<i>string</i>	Contractor name
type	<i>Array</i>	Type(s) of contractor
access	<i>string</i>	Access level
active	<i>string</i>	Active status
address	<i>Array</i>	Contractor address(es)
ams_id	<i>string</i>	Contractor ID
category	<i>string</i>	Category
country	<i>string</i>	Country

Field	Data Type	Description
country_states	<i>string</i>	State/Province
created_at	<i>Date, null</i>	Created timestamp
updated_at	<i>Date, null</i>	Updated timestamp
shareto	<i>Array</i>	Shared with
store_stock	<i>string</i>	Store stock flag
code	<i>string</i>	Contractor code
email	<i>string, null</i>	Email address
references_doc	<i>Array</i>	Reference documents
is_managed	<i>bool</i>	Managed contractor flag
credit	<i>object</i>	Credit terms object
stamp	<i>null</i>	Digital stamp/approval
monitored	<i>string</i>	Monitoring flag
max_limit_status	<i>string</i>	Credit limit flag

8.19 counter

Stores system-wide counters or identifiers.

Table 117. counter table

Field	Data Type	Description
_id	<i>ObjectId</i>	Unique identifier
type	<i>string</i>	Counter type
year	<i>int</i>	Year
seq	<i>double, long, int</i>	Sequence value

8.20 country

Master table for country names and codes.

Table 118. country table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
code2	<i>string</i>	ISO 2-letter code
code3	<i>string</i>	ISO 3-letter code
code_phone	<i>string</i>	Phone code
country	<i>string</i>	Country name

8.21 d365_inv_arch

Archived inventory records from Dynamics 365.

Table 119. d365_inv_arch table

Field	Data Type	Description
_id	ObjectId	Unique identifier for the record
aging	string, double	Aging or shelf-life info (string or calculated value)
allocation	string	Allocation category or assignment
arrival	long, int	Arrival timestamp
arrival_text	string	Human-readable arrival info
batch	string	Batch or lot number
book_value	double	Book value of the item
book_value_mt	double	Book value per metric ton
connection	string, null	Connection type or model
connection2	string, null	Secondary connection type

Field	Data Type	Description
grade	null, string	Material or quality grade
id_item	string	ID of the item
item_desc	string	Description of the item
itemcode	string	Item code or SKU
mb	bool	MB classification flag (likely internal business logic)
mill	string, null	Manufacturer or mill of origin
od	string, null	Outer diameter
od2	string, null	Secondary outer diameter
period	string	Reporting or inventory period
pinbox	string, null	Pin or box connection
pinbox2	string, null	Secondary pin or box
product_level	string, null	Product classification level
qty	int	Quantity in stock
qty_m	double	Quantity measured in meters
qty_mt	double	Quantity measured in metric tons
range	null, string	Range classification (e.g., R1, R2, R3)
range_length	null, double	Length of the range
range_unit	string, null	Unit of the range (e.g., ft, m)
sc_desc	string, null	Supply chain or specification description
seq	int	Sequence or sort order
total_value	double	Total value of the stock item

Field	Data Type	Description
type	null, string	Type of item or material
typeYard	string	Yard classification/type
urut	int, long	Record/order number
warehouse	string, null	Warehouse location or code
wh_code	string	Warehouse code
wt	string, null	Weight
wt2	string, null	Secondary Weight

8.22 d365_inv_log

Logs inventory transactions and changes from D365.

Table 120. d365_inv_log table

Field	Data Type	Description
_id	string	Unique identifier
whencreate	int	Record creation timestamp
mb	bool	MB flag (e.g., indicates source or category)
seq	int	Sequence number
whocreate	string	Creator or user who added the record
period	string	Reporting period (e.g., "04-2023")

8.23 d365_obs_arch

Archived records of observations or orders from Dynamics 365.

Table 121. d365_obs_arch table

Field	Data Type	Description
_id	ObjectID	Unique identifier

Field	Data Type	Description
seq	<i>int, double</i>	Sequence/order number
pinbox	<i>string, null</i>	Type of pin or box connection
range_length	<i>double, null</i>	Length of the range
wt	<i>null, string</i>	Weight
sc_desc	<i>string, null</i>	Supply chain description
od2	<i>string, null</i>	Secondary outer diameter
mill	<i>string, null</i>	Manufacturing mill
typeYard	<i>string</i>	Type of yard
value_efy	<i>double</i>	Value at End of Financial Year
connection	<i>string, null</i>	Primary connection type
item_desc	<i>string</i>	Item description
allocation	<i>string</i>	Allocation tag
obsolescence	<i>int, double</i>	Obsolescence rate/flag
qty_mt	<i>double</i>	Quantity in metric tons
forecast	<i>int</i>	Forecasted quantity
connection2	<i>string, null</i>	Secondary connection type
obsolescence_value	<i>double</i>	Obsolescence value
aging	<i>string, double</i>	Aging duration/value
urut	<i>int, long</i>	Internal ordering number
remaining_qty	<i>int</i>	Remaining quantity
id_item	<i>string</i>	Item ID

Field	Data Type	Description
book_value_mt	<i>double</i>	Book value per metric ton
period	<i>string</i>	Reporting or archival period
warehouse	<i>string, null</i>	Warehouse name
od	<i>string, null</i>	Outer diameter
type	<i>string, null</i>	Type of item
batch	<i>string</i>	Batch number
wt2	<i>string, null</i>	Secondary Weight
qty_m	<i>double</i>	Quantity in meters
original_value	<i>int, double</i>	Original value before changes
range	<i>string, null</i>	Item range classification (e.g., R1, R2, R3)
product_level	<i>string, null</i>	Product level/category
itemcode	<i>string</i>	Item code or SKU
qty	<i>int</i>	Quantity
book_value	<i>double</i>	Book value
total_value	<i>double</i>	Total monetary value
grade	<i>string, null</i>	Material or quality grade
wh_code	<i>string</i>	Warehouse code
range_unit	<i>null, string</i>	Unit of the range (e.g., "m", "ft")
arrival_text	<i>string</i>	Readable arrival date
aging_efy	<i>string, double</i>	Aging at End of Financial Year
mb	<i>bool</i>	MB flag (likely used internally)

Field	Data Type	Description
pinbox2	<i>string, null</i>	Secondary pin/box
arrival	<i>long, int</i>	Arrival timestamp

8.24 d365_obs_log

Logs updates and changes in D365 observation records.

Table 122. d365_obs_log table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
period	<i>string</i>	Reporting period
whocreate	<i>string</i>	Name of the user who created the log
whencreate	<i>int</i>	Timestamp when the log was created
mb	<i>bool</i>	MB classification flag
seq	<i>int</i>	Sequence number
current	<i>int</i>	Timestamp or ID for current record cutoff
efy	<i>int</i>	End of fiscal year timestamp

8.25 ddd_logger

Debug and diagnostic logs for internal system monitoring.

Table 123. ddd_logger table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
timestamp	<i>int</i>	Time the error or log was recorded
file	<i>string</i>	File path where the issue occurred
msg	<i>string</i>	Error message

Field	Data Type	Description
trace	<i>string</i>	Stack trace or traceback
class	<i>string</i>	Class name involved in the error
snapshot	<i>object, Array, null</i>	Captured object or environment at time of error (optional(nullable))

8.26 delivery

Details of deliveries including date, location, and items.

Table 124. delivery table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
name	<i>string</i>	Delivery name or label
idClient	<i>null</i>	Linked client ID
address	<i>null</i>	Delivery address

8.27 deliveryterms

Stores information about organizational departments.

Table 125. department table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
name	<i>string</i>	Department name

8.28 department

Master list of delivery terms and their definitions.

Table 126. deliveryterms table

Field	Data Type	Description
<code>_id</code>	<code>string</code>	Unique identifier
<code>name</code>	<code>string</code>	Delivery term name (e.g., FOB)

8.29 doc_format

Templates and settings used for document formatting.

Table 127. doc_format table

Field	Data Type	Description
<code>_id</code>	<code>string</code>	Unique identifier
<code>name</code>	<code>string</code>	Format name or label
<code>doc_type</code>	<code>string</code>	Type of document
<code>format</code>	<code>string</code>	Format string (e.g., "{year}-{seq}")
<code>padding</code>	<code>int</code>	Padding size for number (e.g., 3 → 001)
<code>prefix</code>	<code>string</code>	Prefix used in document numbers
<code>seq</code>	<code>int</code>	Current sequence number
<code>year</code>	<code>string</code>	Year identifier used
<code>lastyear</code>	<code>int, string</code>	Last processed year (can be int or string)

8.30 docs

Storage of document files, metadata, and versions.

Table 128. docs table

Field	Data Type	Description
<code>_id</code>	<code>ObjectID</code>	Unique identifier

Field	Data Type	Description
revno	<i>string</i>	Revision number
end_date	<i>long</i>	Document end date
fname	<i>string</i>	File name
active	<i>string</i>	Document active status
start_date	<i>long</i>	Start date of the document
commitment_po	<i>string</i>	Related commitment PO
doc_type	<i>string</i>	Document type
docno	<i>string</i>	Document number
folder	<i>null</i>	Folder name/reference
shareto	Array	Users shared with
desc	<i>string</i>	Document description
docdate	<i>int</i>	Document date
contract_name	<i>string</i>	Associated contract name
client	<i>string</i>	Associated client
id_contract	<i>string</i>	Related contract ID
lastpostdate	<i>long, int</i>	Last update timestamp
lastpostwho	<i>string</i>	Who last updated the document

8.31 document

Master metadata and properties of all document records.

Table 129. document table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
ref_seq	<i>int</i>	Reference sequence number
doc_type	<i>string</i>	Type of document
vendor_invoice	<i>Array</i>	List of vendor invoices
confirmation_suffix	<i>null</i>	Confirmation suffix
template	<i>null</i>	Document template
items	<i>Array</i>	Items listed in the document
use_item	<i>string</i>	Usage reference for item
support_doc	<i>Array</i>	Supporting documents
pono	<i>string, null</i>	Purchase order number
use_cost_detail	<i>string</i>	Whether cost details are used
customer	<i>string, null</i>	Customer name or ID
attention_of	<i>null, string</i>	Attention of person or dept
issued_by	<i>null</i>	Who issued the document
completed_invoice	<i>Array, object</i>	Completed invoice information
total_vat_amount	<i>object</i>	Total VAT applicable
no_seq	<i>string</i>	Document number sequence
status	<i>string</i>	Document status
vendor_address	<i>string, null</i>	Vendor address

Field	Data Type	Description
report	<i>object, null</i>	Attached report object
id_transfer	<i>string, null, Array</i>	Transfer ID
required_date	<i>null</i>	Required date
cancel	<i>Array, object</i>	Cancellation record
vendor_quotation	<i>string, null</i>	Vendor quotation ID or number
delivery_date	<i>null</i>	Date of delivery
sw	<i>null</i>	Unknown/Software switch flag
notesC	<i>string, null</i>	Customer Notes
type	<i>null</i>	Type of transaction
works	<i>null</i>	Related works/projects
total_total	<i>object</i>	Total amount (including tax, etc.)
customer_ref	<i>null</i>	Customer reference
updateid	<i>null, string</i>	ID of the last update
use_sow	<i>null</i>	Use scope of work
updatewho	<i>string, null</i>	Who updated the document
createwho	<i>string</i>	Who created the document
issued_to	<i>null</i>	Recipient of the document
etd	<i>null</i>	Estimated time of departure
completed_info	<i>null</i>	Completion details
invoice	<i>object, null</i>	Invoice object

Field	Data Type	Description
createdate	<i>int</i>	Creation timestamp
transporter	<i>string</i>	Transporter name
issued_invoice_to	<i>string, null</i>	Invoice recipient
updatedate	<i>int, null</i>	Last updated timestamp
cost_details	Array	Detailed cost breakdown
total_sub_total	object	Subtotal before tax
transports	Array	Transport modes or routes
use_transport	<i>string</i>	Transport method used
no	<i>string</i>	Document number
createid	<i>string</i>	ID of the creator

8.32 ehs

Environmental Health and Safety compliance records.

Table 130. ehs table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
hazard	<i>int</i>	Number of hazards reported
ehs_year	<i>int</i>	Year of EHS data
trcfr	<i>int</i>	TRCFR metric
nearmiss	<i>int</i>	Number of near misses
eniv	<i>int</i>	Environmental incidents
mti	<i>int</i>	MTI metric

Field	Data Type	Description
workhours	<i>int</i>	Total work hours
trcsr	<i>int</i>	TRCSR metric
recordable	<i>int</i>	Recordable incidents
folder	<i>string</i>	Associated folder
incident	<i>Array</i>	List of incidents
ehs_month	<i>int</i>	Month of EHS data
accident	<i>int</i>	Accidents recorded
lastpostdate	<i>int</i>	Last modification timestamp
lti	<i>int</i>	Lost time incidents
lastpostwho	<i>string</i>	User who last updated

8.33 ehs_target

Targets and goals for EHS performance tracking.

Table 131. ehs_target table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
hazard	<i>int</i>	Target number of hazards
ehs_year	<i>int</i>	Target year
trcfr	<i>int</i>	TRCFR target
nearmiss	<i>int</i>	Near miss target
eniv	<i>int</i>	Environmental incident target
mti	<i>int</i>	MTI target

Field	Data Type	Description
workhours	<i>int</i>	Work hours target
trcsr	<i>int</i>	TRCSR target
recordable	<i>int</i>	Recordable incident target
folder	<i>string</i>	Folder or reference path
accident	<i>int</i>	Accident target
lastpostdate	<i>int</i>	Last update
lti	<i>int</i>	LTI target
lastpostwho	<i>string</i>	Last updater

8.34 email_log

Logs of all sent/received emails and statuses.

Table 132. email_log table

Field	Data Type	Description
_id	<i>string</i>	Unique log entry ID
subject	<i>string, null</i>	Email subject
when	<i>Date</i>	When the email was logged
to	<i>string</i>	Recipient
status	<i>bool</i>	Whether the email was sent successfully
err_msg	<i>string</i>	Error message if failed
body	<i>string</i>	Email content

8.35 erp_findim

ERP financial dimension data for reporting.

Table 133. erp_findim table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
status	<i>string</i>	Status of the record
project	<i>string</i>	Related project
delete_date	<i>null</i>	Deletion date if removed
findim	<i>object</i>	Financial dimension object
customer	<i>string</i>	Associated customer

8.36 erp_log

Logs interactions and events with ERP systems.

Table 134. erp_log table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
created_at	<i>Date, int</i>	Creation timestamp
response	<i>object, string, null</i>	Response received from ERP system
status	<i>string, int</i>	Status of the log entry
updated_at	<i>Date, int</i>	Last update timestamp
data_id	<i>string, null</i>	ID of the data entry
updated_at_literal	<i>string</i>	Readable update date
request	<i>object</i>	Request payload
function	<i>string</i>	ERP function or action

8.37 fc_acc_weighted

Forecast data weighted by financial account priorities.

Table 135. fc_acc_weighted table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
customer	<i>string</i>	Customer ID
month	<i>int</i>	Month of forecast
year	<i>int</i>	Year of forecast
date	<i>int</i>	Forecast date
avg	<i>int</i>	Weighted Average Month Value
acc	<i>string</i>	Account reference
whenupdate	<i>int</i>	Last update timestamp
acc_raw	<i>double</i>	Raw account metric

8.38 fc_d365

Forecasting data originating from D365.

Table 136. fc_d365 table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
resolved	<i>bool</i>	Whether the entry is resolved
transaction_date	<i>int</i>	Date of transaction
qty	<i>int, double</i>	Quantity
qty_edit	<i>int</i>	Edited quantity (if applicable)
year	<i>int</i>	Year of transaction

Field	Data Type	Description
month	<i>int</i>	Month of transaction
customer	<i>string</i>	Associated customer
id_item	<i>string</i>	Item ID

8.39 fc_d365_hist

Historical records of D365 forecasts.

Table 137. fc_d365_hist table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
who_update	<i>string</i>	Who performed the update
last_update	<i>int</i>	Last update timestamp

8.40 fc_d365_recap

Summarized recap entries of D365 forecast results.

Table 138. fc_d365_recap table

Field	Data Type	Description
_id	ObjectId	Unique identifier
month	int, double	Transaction month
year	int	Transaction year
hist_month	int	Historical month
hist_year	int	Historical year
transaction_date	int	Date of transaction
qty	int	Quantity
price	double	Unit price

Field	Data Type	Description
tprice	double	Total price
resolved	string	Resolved status
flag	bool	Flag (e.g., for data integrity)
customer	string	Associated customer
id_item	string	Item ID

8.41 fees

Stores configurable fee structures applied to contracts or transactions.

Table 139. fees table

Field	Data Type	Description
_id	<i>ObjectId</i>	Unique identifier
consignment	<i>Array</i>	List of consignment entries
nonconsignment	<i>Array</i>	List of non-consignment entries
customer	<i>string</i>	Customer ID
validfrom	<i>double</i>	Valid from date (timestamp)

8.42 forecast

Contains forecasted demand or inventory needs by period and product.

Table 140. forecast table

Field	Data Type	Description
_id	<i>ObjectId, string</i>	Unique identifier
lastpostdate	<i>int</i>	Timestamp of last post
lastpostwho	<i>string</i>	Who performed the last post
contract	<i>string</i>	Associated contract

Field	Data Type	Description
actual	<i>double</i>	Actual value
forecast	<i>double</i>	Forecasted value
fcast_year	<i>int</i>	Forecast year
fcast_quarter	<i>int</i>	Forecast quarter
bu	<i>string</i>	Business unit
category	<i>string</i>	Forecast category
folder	<i>string</i>	Folder reference

8.43 forecast_acc

Forecast data filtered or grouped by account.

Table 141. forecast_acc table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
acc	<i>int, double</i>	Forecast Accuracy Value
id_item	<i>string</i>	Item ID
customer	<i>string, int</i>	Customer reference

8.44 forecast_acc_hist

Historical forecast accuracy for evaluation over time.

Table 142. forecast_acc_hist table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
usage	<i>int, double</i>	Historical usage
acc	<i>int, double</i>	Forecast Accuracy Value

Field	Data Type	Description
forecast	<i>int</i>	Forecasted quantity
year	<i>int</i>	Forecast year
month	<i>int</i>	Forecast month
customer	<i>string, int</i>	Customer reference
id_item	<i>string</i>	Item ID

8.45 forecast_d365_arch

Archived forecast data from Dynamics 365.

Table 143. forecast_d365_arch table

Field	Data Type	Description
_id	<i>ObjectId</i>	Unique identifier
month	<i>int</i>	Forecast month
year	<i>int</i>	Forecast year
transaction_date	<i>int</i>	Date of transaction
qty	<i>double</i>	Quantity forecasted
amount	<i>double</i>	Forecasted amount
hist_month	<i>int</i>	Historical month
hist_year	<i>int</i>	Historical year
updatewhen	<i>int</i>	Last update timestamp
accountno	<i>string, null</i>	Account number
id_item	<i>string</i>	Item ID

8.46 forecast_mb_hist

Historical forecast data of merchandise balances.

Table 144. forecast_mb_hist table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
type	<i>string</i>	Forecast type
qty	<i>int</i>	Forecast quantity
price	<i>int</i>	Forecast unit price
total_price	<i>int</i>	Total forecast price
transaction_date	<i>int</i>	Date of forecast
start_date	<i>int</i>	Forecast start date
end_date	<i>int</i>	Forecast end date
setup_date	<i>int</i>	Forecast setup timestamp
who_update	<i>string</i>	User who updated the record
month	<i>int</i>	Month of forecast
year	<i>int</i>	Year of forecast
order	<i>int</i>	Forecast order or sequence
group	<i>string</i>	Forecast group
wellname	<i>string, null</i>	Well name
rig	<i>string, null</i>	Rig name or reference
hist_month	<i>int</i>	Historical month
hist_year	<i>int</i>	Historical year
id_item	<i>string</i>	Item ID

Field	Data Type	Description
customer	<i>string</i>	Customer reference

8.47 grade

Classification of item or service quality levels.

Table 145. grade table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
name	<i>string</i>	Grade name
code	<i>string</i>	Grade code
prop	<i>string</i>	Proprietary Name
type	<i>string</i>	Grade type/category

8.48 helps

User help content including guides and FAQs.

Table 146. helps table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
parent_id	<i>string</i>	Parent help entry (for nesting or grouping)
sort	<i>int</i>	Sort order
show	<i>bool</i>	Whether to display this help item
title	<i>string</i>	Title of the help topic
note	<i>string</i>	Additional notes
type	<i>string</i>	Type/category of help entry
urls	<i>null, Array</i>	Related URLs (e.g., resources, videos)

Field	Data Type	Description
video_filename	<i>string</i>	File name of associated video
video2_filename	<i>string</i>	Secondary video file (e.g., alternative format)
content	<i>string</i>	Help content

8.49 imei

IMEI tracking table for traceable mobile assets.

Table 147. imei table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
imei	<i>string</i>	IMEI number (device identifier)
idyard	<i>string</i>	Associated yard or location ID
hash_token	<i>string</i>	Hashed authentication token
token	<i>string</i>	Plain token
otp	<i>null</i>	One-time password (if used)
is_active	<i>string</i>	Activity status (e.g., active/inactive)
description	<i>string</i>	Description of the device or record

8.50 index

Defined index identifiers used across modules.

Table 148. index table

Field	Data Type	Description
_id	<i>ObjectId</i>	Unique identifier
unit	<i>string</i>	Unit of measurement
source	<i>string</i>	Source of index data

Field	Data Type	Description
source_option	<i>string</i>	Source configuration or type
name	<i>string</i>	Name of the index
access	<i>string</i>	Access control or visibility setting
shareto	<i>Array</i>	List of users/groups the index is shared with
mode	<i>string</i>	Operational mode of the index
lastpostdate	<i>int</i>	Last updated timestamp
lastpostwho	<i>string</i>	Last updated by

8.51 indexval

Values associated with defined indexes.

Table 149. indexval table

Field	Data Type	Description
_id	<i>ObjectId</i>	Unique identifier
index	<i>string</i>	Related index name
idx_month	<i>string, int</i>	Index month
idx_year	<i>string, int</i>	Index year
idx_date	<i>int</i>	Date the index was posted
folder	<i>string</i>	Related folder or document group
contract	<i>null</i>	Related contract
change	<i>int</i>	Change value (e.g., % or points)
latestval	<i>string, double</i>	Latest value of the index
islatest	<i>string</i>	Flag to indicate if this is the latest version

Field	Data Type	Description
lastpostdate	<i>int</i>	Last updated timestamp
lastpostwho	<i>string</i>	Who last updated the index

8.52 inquiry

Captures external or internal inquiry submissions, often used for technical, commercial, or operational questions.

Table 150. inquiry table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier of the inquiry
who	<i>string</i>	Name of the person who submitted the inquiry
who_email	<i>string</i>	Email address of the person
when	<i>int</i>	Timestamp of the inquiry submission
bu	<i>string</i>	Business unit making the inquiry
item	<i>object</i>	Item(s) being inquired, grouped by vendor or BU
item.HALLMARK[]	<i>array of objects</i>	List of items requested from HALLMARK
item.HALLMARK[n]._id	<i>string</i>	Item line ID
item.HALLMARK[n].bu	<i>string</i>	BU/Vendor name
item.HALLMARK[n].id_item	<i>string</i>	Internal item ID
item.HALLMARK[n].qty	<i>int</i>	Quantity requested
item.HALLMARK[n].qty_length	<i>double</i>	Total length requested

Field	Data Type	Description
item.HALLMARK[n].mill	<i>object</i>	Dictionary of mill references with index-based keys
item.HALLMARK[n].desc	<i>string</i>	Description of the item
item.HALLMARK[n].code	<i>string</i>	Item code
item.HALLMARK[n].qty_inc	<i>string</i>	Quantity increment or unit
note	<i>object</i>	Notes per BU (e.g., note.HALLMARK)
status	<i>string</i>	Inquiry status (e.g., 'p' for pending)
follow_up_status	<i>string</i>	Status of follow-up (e.g., close)
follow_up_updated_at	<i>int</i>	Timestamp of follow-up status update

8.53 inspection_counter

Counters for generating inspection IDs.

Table 151. inspection_counter table

Field	Data Type	Description
_id	<i>Object/Id</i>	Unique identifier
year	<i>long</i>	Year of inspection count
seq	<i>long</i>	Sequence number (e.g., auto-number)

8.54 inspection_image

Stores images captured during inspections.

Table 152. inspection_image table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier

Field	Data Type	Description
name	<i>string</i>	Image name
id_inspection	<i>string</i>	Related inspection ID
type	<i>string</i>	File/image type
id_files	<i>string</i>	File ID or storage reference
uuid	<i>string</i>	UUID (universally unique identifier)
src	<i>string</i>	Image source path or URL

8.55 **inspection_temp**

Temporary data for ongoing inspection sessions.

Table 153. inspection_temp table

Field	Data Type	Description
length	<i>double</i>	Length of the inspected item
pipeno	<i>string, int</i>	Pipe number
cr	<i>string</i>	Inspection code or remark
notes	<i>string</i>	Inspection notes
_id	<i>string</i>	Document ID
id_inspection	<i>string</i>	Inspection reference ID
files	<i>null, Array</i>	Associated file references
pin	<i>Array</i>	Pin connection data
box	<i>Array</i>	Box connection data
heatno	<i>string</i>	Heat number
body	<i>Array</i>	Pipe body inspection details

8.56 internal_note

Private notes for internal communication.

Table 154. internal_note table

Field	Data Type	Description
updatewhen	<i>int</i>	Last update timestamp
filename	<i>string</i>	File name
notes	<i>string</i>	Note content
type	<i>string</i>	Type/category of note
_id	<i>string</i>	Document ID
path	<i>string</i>	File path
username	<i>string</i>	User who added/updated the note
id_transfer	<i>string</i>	Transfer reference ID
when	<i>int</i>	Creation timestamp
who	<i>string</i>	Creator of the note

8.57 inventory_report

Reports summarizing inventory levels or changes.

Table 155. inventory_report table

Field	Data Type	Description
wt	<i>null, double</i>	Weight
lot_no	<i>string</i>	Lot number
ageing	<i>null, int</i>	Age in days/months
condition	<i>string</i>	Item condition
location	<i>string</i>	Item location

Field	Data Type	Description
item_desc	<i>string</i>	Item description
qr_no	<i>null</i>	Quarantine No
connection	<i>string</i>	Connection type
cust_item_no	<i>string</i>	Customer item number
allocation	<i>string</i>	Allocation status
location_name	<i>string</i>	Human-readable location name
qty_quarantine	<i>null</i>	Quantity in quarantine
id_transfer	<i>null</i>	Transfer ID
rack	<i>string</i>	Storage rack location
id_item	<i>string</i>	Item ID
od	<i>string</i>	Outer diameter
item_type	<i>string</i>	Type of item
contract	<i>string</i>	Related contract
date_created	<i>null</i>	Creation date
owned_name	<i>string</i>	Name of the owner
_id	<i>string</i>	Document ID
grade	<i>string</i>	Grade or quality classification
uom	<i>string</i>	Unit of measure
range	<i>string</i>	Size or range
qty	<i>int</i>	Quantity available
product_level	<i>string</i>	Product level/category

Field	Data Type	Description
reason	<i>null</i>	Reason for status
owned	<i>string</i>	Owner ID or reference

8.58 invoice

Table containing invoice details for transactions.

Table 156. invoice table

Field	Data Type	Description
inv_close	<i>int</i>	Invoice close status (e.g., 0 or 1)
customer	<i>string</i>	Customer ID
source	<i>string</i>	Source system
isimport	<i>bool</i>	Import status
inv_amount	<i>int, double</i>	Invoice amount
inv_due	<i>int</i>	Due date (timestamp)
cust_name	<i>string</i>	Customer name
top	<i>string</i>	Terms of payment
sales_order	<i>string</i>	Sales order reference
inv_settled_curr	<i>double</i>	Settled amount in current currency
folder	<i>string</i>	Folder path
filename	<i>string</i>	Invoice file name
location	<i>string</i>	Invoice location
cust_acc	<i>string</i>	Customer account
tax_invoice	<i>string</i>	Tax invoice number

Field	Data Type	Description
inv_date	<i>int</i>	Invoice date (timestamp)
_id	<i>string</i>	Document ID
inv_curr	<i>string</i>	Currency of invoice
lastpostdate	<i>int</i>	Last posting date
lastpostwho	<i>null</i>	User who last posted
inv_paid	<i>string</i>	Payment status or paid amount

8.59 invoice_mts

Invoices generated for make-to-stock orders.

Table 157. invoice_mts table

Field	Data Type	Description
_id	<i>string</i>	Document ID
no	<i>string</i>	Invoice number
no_gen	<i>int</i>	Generated number (sequence)
id_po	<i>string</i>	Purchase order ID
files	<i>Array</i>	Attached files
items	<i>Array, object</i>	Line items
date	<i>int</i>	Invoice date (timestamp)
notes	<i>string</i>	Notes or remarks

8.60 invoice_pdf

PDF files generated for invoice documents.

Table 158. invoice_pdf table

Field	Data Type	Description
tax_invoice	<i>string, null</i>	Tax invoice number
filename	<i>string</i>	PDF filename
source	<i>string</i>	Source system
_id	<i>string</i>	Document ID
sales_order	<i>string</i>	Sales order reference

8.61 invreport

Detailed inventory reporting data.

Table 159. invreport table

Field	Data Type	Description
yard	<i>string</i>	Yard name
item_code	<i>string</i>	Item code
mill	<i>string</i>	Mill or manufacturer
location	<i>string, null</i>	Storage location
year	<i>string</i>	Year of report
customer	<i>null</i>	Customer name/ID
item_desc	<i>string</i>	Item description
lastinspectdate_raw	<i>string, int</i>	Last inspection date (raw)
cust_item_no	<i>string, null</i>	Customer item number
allocation	<i>string, null</i>	Allocation info

Field	Data Type	Description
hist_month	<i>string</i>	Historical month
hist_year	<i>string</i>	Historical year
aging	<i>int</i>	Item aging (in days/months)
aging_fee_curr	<i>string</i>	Aging fee currency
mgt_fee	<i>double</i>	Management fee
meter	<i>double</i>	Measured length
lastinspectdate	<i>string</i>	Last inspection date (formatted)
id_item	<i>string</i>	Item ID
item_type_raw	<i>string</i>	Raw item type
cpo	<i>string</i>	Batch Number
item_type	<i>string, null</i>	Processed item type
_id	<i>string</i>	Document ID
aging_fee	<i>double</i>	Aging fee
qty	<i>int</i>	Quantity
mgt_fee_curr	<i>string</i>	Management fee currency
ft	<i>double</i>	Length in feet
mt	<i>double</i>	Weight in metric tons
id_customer	<i>string</i>	Customer ID

8.62 item

Master list of all managed items or products.

Table 160. item table

Field	Data Type	Description
range_unit	string	Unit of measurement for item range (e.g., meters, feet).
sc	Array	Special Condition
colorband	string	Visual marking or color code for item identification.
sr_length	double	Sucker Rod Length
connprop	string	Connection properties.
sr_coupling_type	string	Sucker Rod Coupling
range	string	Item range description (e.g., Range 1, 2, 3).
product_level	string	Classification of item level (e.g., raw, semi-finished).
od_mm	double	Outer diameter in millimeters.
is_trackable	bool	Indicates if item is trackable (e.g., via serial or RFID).
_id	string	Unique identifier.
wt2	string	Secondary Weight
active	string	Status flag (e.g., "Y" for active).
sr_length_unit	string	Sucker Rod Length
old_axaptano	string	Legacy AXAPTA item number.
sr_guide	string	Sucker Rod Guide
turnrate_category	string, null	Category of turning rate; nullable.
axaptano	string	Item Code

Field	Data Type	Description
type	<i>string</i>	Item type (e.g., pipe, rod).
type_name	<i>string, null</i>	Human-readable name for the item type.
additional_item	<i>Array</i>	List of related or supplementary items.
gradetype	<i>string, null</i>	Grade category/type.
sr_coupling_material	<i>string</i>	Sucker Rod Coupling
cmdesc	<i>string, null</i>	Commercial or custom description.
assign	<i>object</i>	Assignment info (e.g., location, user).
sctext	<i>Array, object</i>	Special Condition Text
wt2_value	<i>double</i>	Numeric value corresponding to Secondary Weight.
cmweight	<i>string</i>	Cementing Equipment Weight
wt_mm	<i>double</i>	Wall Thickness in millimeters
attachment	<i>Array</i>	List of attachments (e.g., docs, images).
sr_weight	<i>string, double</i>	Weight of Sucker Rod
wt_value	<i>double</i>	Actual weight of item.
pinbox	<i>string</i>	Pin/box connection type.
range_length	<i>double</i>	Length corresponding to the range.
od_inch	<i>int, null, double</i>	Outer diameter in inches.
sr_guide_material	<i>string</i>	Material of the guide.
od2	<i>string</i>	Secondary outer diameter (raw value).
od2_inch	<i>int, double, null</i>	Secondary outer diameter in inches.

Field	Data Type	Description
sr_coupling_grade	<i>string</i>	Sucker Rod Guide
mt	<i>string, double</i>	Mass or metric tons.
scdesc	<i>string, null</i>	Description for sc.
oldno	<i>string</i>	Previous item number.
pinbox2	<i>string</i>	Alternate pin/box connection.
sr_rod_od	<i>string</i>	Outer diameter of the service rod.
is_assembly	<i>bool</i>	Whether the item is an assembly.
sr_guide_tubing_size	<i>string</i>	Tubing size for the guide.
sr_rod_od_inch	<i>double</i>	Rod outer diameter in inches.
sr_tubing_size	<i>string</i>	Size of the tubing.
sr_tubing_size_inch	<i>double</i>	Tubing size in inches.
gradeprop	<i>string</i>	Grade property classification.
grade	<i>string</i>	Material or quality grade.
type_literal	<i>string</i>	Literal string value for the type.
od	<i>string</i>	Outer diameter (string format).
conversion	<i>object</i>	Conversion metrics or factor object.
desc	<i>string</i>	General item description.
class	<i>string</i>	Classification category.
sr_type	<i>string</i>	Sucker Rod Type
sr_material	<i>string</i>	Sucker Rod Material

Field	Data Type	Description
urut	double	Sequence or order number.
sr_guide_qty	int	Sucker Rod Guide
protectormat	string	Protective material used.
sr_coupling_connection	string	Sucker Rod Coupling
connection2	string	Secondary connection info.
connection	string	Main connection type.
sr_guide_type	string	Sucker Rod Guide
map_customer_item_no	Array, object	Customer-item mapping details.
sr_pinbox	string	Pinbox details specific to service rod.
wt	string	Weight in string format.
primary_item	object	Linked or parent item data.

8.63 item_counter

Counter table for generating item-related IDs.

Table 161. item_counter table

Field	Data Type	Description
seq	long	Sequential counter (e.g., for generating item numbers).
type	string	Type/category of the counter.
_id	ObjectId	Unique identifier (MongoDB ObjectId).

8.64 item_type

Categorization of item types by usage or form.

Table 162. item_type table

Field	Data Type	Description
unit	<i>string</i>	Unit of measurement (e.g., ft, m).
default_batch_tracking	<i>bool</i>	Indicates if batch tracking is enabled by default.
pipe_family	<i>string</i>	Pipe family classification.
_id	<i>string</i>	Unique identifier.
name	<i>string</i>	Name of the item type.

8.65 item_type_attribute

Attributes specific to certain item types.

Table 163. item_type_attribute table

Field	Data Type	Description
name	<i>string</i>	Attribute type name.
attribute	<i>Array</i>	List of attributes or values.
order	<i>string</i>	Sort order or display order.
code_prefix	<i>string</i>	Prefix used for code generation.
image	<i>object</i>	Associated image or media metadata.
desc	<i>Array</i>	Description details.
consumable	<i>bool</i>	Indicates if the item is consumable.
default_batch_tracking	<i>bool</i>	Batch tracking default status.
unit	<i>string</i>	Measurement unit.
active	<i>bool</i>	Active/inactive status.

Field	Data Type	Description
is_mt	<i>bool, null</i>	Indicates if related to metric tons.
_id	<i>string</i>	Unique identifier.

8.66 itemcard

Store available stock ledger.

Table 164. itemcard table

Field	Data Type	Description
lot_id	<i>string, null</i>	Lot or batch identifier.
rep	<i>bool</i>	Indicates if the item is a replacement.
appvwhen	<i>int</i>	Approval timestamp.
ft	<i>double</i>	Footage or length in feet.
type_loc	<i>string</i>	Location type (e.g., yard, rack).
idtrans	<i>string</i>	Transaction ID.
condition	<i>string</i>	Condition of the item (e.g., New, Used).
vessel	<i>string, null</i>	Vessel or shipping info.
repair	<i>string</i>	Repair status or notes.
flag	<i>string</i>	Miscellaneous flag.
tag	<i>null</i>	Possibly unused or deprecated.
allocation	<i>string</i>	Well Allocation
appvwho	<i>string, null</i>	Approved by user.
port	<i>string</i>	Port of origin or destination.
is_reserved	<i>bool</i>	Reservation status.

Field	Data Type	Description
source	<i>string</i>	Origin/source system.
id_transfer	<i>string</i>	Transfer ID related to logistics.
oldbatch	<i>string</i>	Previous batch number.
postingdate	<i>Date</i>	Date when item was posted.
appv	<i>string</i>	Approval status.
id_item	<i>string</i>	Linked item ID.
rack	<i>string</i>	Storage rack location.
rev	<i>string, null</i>	Revision info.
m	<i>double</i>	Mass or meters.
type	<i>string</i>	Item type.
locked	<i>string</i>	Customer Allocation
stage	<i>string</i>	Process stage.
updatewho	<i>string, null</i>	User who last updated.
updatewhen	<i>int, null, double</i>	Last updated timestamp.
_id	<i>string</i>	Unique identifier.
batch	<i>string</i>	Current batch number.
description	<i>string</i>	Description of the item.
createdate	<i>int</i>	Creation timestamp.
id_quarantine	<i>string, null</i>	Quarantine tracking ID.
calloff	<i>bool</i>	Indicates call-off status.
qty	<i>int</i>	Quantity of items.

Field	Data Type	Description
notes	<i>string</i>	Notes or comments.
updatedate	<i>int</i>	Last update date.
owned	<i>string</i>	Ownership status or name.
eta	<i>Date</i>	Estimated time of arrival.
version	<i>string</i>	Version number of item/batch.
mt	<i>int, double</i>	Mass in metric tons.
yard	<i>string</i>	Storage yard or area.

8.67 itemcard_balance

Store account balance ledger.

Table 165. itemcard_balance table

Field	Data Type	Description
rep	<i>bool</i>	Indicates whether the item is a replacement.
ft	<i>double</i>	Length in feet or a similar measurement unit.
unit_price	<i>double, string, int, null</i>	Unit price of the item (can be stored as string or number).
owned	<i>string, null</i>	Owner of the item; may refer to a customer or department.
total_price	<i>double, int, null</i>	Total calculated price of the item(s).
notes	<i>string</i>	Additional remarks or notes.
total_priceprice	<i>string</i>	Redundant or legacy field storing total price as string.
_id	<i>string</i>	MongoDB document unique identifier.
whenupdate	<i>double, int</i>	Timestamp of the latest update.

Field	Data Type	Description
files	<i>string, Array, null</i>	File references, such as document paths or metadata.
batch	<i>null, string</i>	Batch identifier for the item.
jan_repair	<i>bool</i>	Whether the item underwent repair in January.
sales_price	<i>double</i>	Selling price of the item.
transaction_date	<i>double, int</i>	Date of the transaction (epoch format).
is_po	<i>bool</i>	Indicates if this is a purchase order item.
curr	<i>string</i>	Currency code (e.g., USD, IDR).
type	<i>string</i>	Type/category of item (e.g., Pipe, Tool).
axaptano	<i>string</i>	ERP or Axapta item number.
batch_price	<i>double</i>	Price associated with a full batch.
bu	<i>string</i>	Business unit or department.
from	<i>string</i>	Origin location or source identifier.
to	<i>string</i>	Destination location or target identifier.
to_name	<i>string</i>	Destination location name.
price	<i>int, double, null</i>	Price per unit or total price depending on context.
invno	<i>string</i>	Invoice number.
wellname	<i>string, null</i>	Name of the well associated with the item.
pono	<i>string</i>	Purchase order number.
from_name	<i>string</i>	Origin location name.
customer	<i>string</i>	Customer name or code.

Field	Data Type	Description
unit	<i>string, double, null</i>	Measurement unit (e.g., pcs, meters).
condition	<i>null</i>	Condition status (data not consistently entered).
year	<i>string, int, double</i>	Year associated with the item (e.g., manufactured year).
mill	<i>string, null</i>	Manufacturer or mill name.
client	<i>string</i>	Client code.
client_name	<i>string</i>	Client full name.
no	<i>string, null</i>	Miscellaneous reference number.
mt	<i>double, int</i>	Weight or mass, likely in metric tons.
usage_type	<i>string</i>	Usage category (e.g., rental, purchase, internal use).
invgroup	<i>string</i>	Inventory grouping or classification.
qty	<i>double, int</i>	Quantity of items.
when_update	<i>double</i>	Duplicate or normalized form of whenupdate.
isimport	<i>bool</i>	Indicates if the item is imported.
lineno	<i>string</i>	Line number on the invoice or order document.
rig	<i>string, double, null</i>	Associated rig name or ID.
whoupdate	<i>string</i>	Username or ID of person who made the last update.
po_type	<i>null</i>	Type of purchase order (data not consistently used).
month	<i>double, string, int</i>	Month associated with the transaction.

Field	Data Type	Description
typetrans	<i>string, null</i>	Type of transaction (e.g., in, out, transfer).
is_opcust	<i>bool, null</i>	Indicates if item is for operational customer.
m	<i>double</i>	Possibly length or weight (context-specific).
id_item	<i>string</i>	Item ID, likely links to item collection.
transaction	<i>null</i>	Placeholder field for transaction reference.
insp_type	<i>string</i>	Type of inspection (e.g., visual, UT).
source	<i>string, null</i>	Source of item (e.g., vendor, warehouse).
id_log	<i>string</i>	Log or audit identifier.
allocation	<i>string, null</i>	Allocation target or purpose.
bu_incoming	<i>string</i>	Business unit receiving the item.
itemdesc	<i>string</i>	Item description.
financial_year	<i>string, int</i>	Financial year the item was recorded.
location	<i>string, null</i>	Storage or operational location.
incoming_price	<i>string</i>	Price at which the item was received.
idtrans	<i>string, null</i>	Transaction ID.

8.68 itemcard_rack

Store physical stock ledger.

Table 166. *itemcard_rack* table

Field	Data Type	Description
condition	<i>string</i>	Condition of the item (e.g., new, used, repaired).
rep	<i>bool</i>	Indicates whether the item is a replacement.

Field	Data Type	Description
idtrans	<i>string</i>	Transaction identifier.
vessel	<i>string, null</i>	Vessel or transport unit.
type_loc	<i>string</i>	Location type (e.g., warehouse, offshore).
appvwhen	<i>int</i>	Time when item was approved (epoch).
repair	<i>bool</i>	Indicates if item is flagged for repair.
ft	<i>double</i>	Length in feet or another unit.
appvwho	<i>string, null</i>	Person who approved the item.
allocation	<i>string</i>	Internal allocation or department usage.
flag	<i>string</i>	Custom flag for special status or condition.
source	<i>string</i>	Origin of the item (e.g., vendor).
port	<i>string</i>	Port or logistics hub.
is_reserved	<i>bool</i>	Indicates if the item is currently reserved.
id_item	<i>string</i>	Reference ID from the item collection.
postingdate	<i>Date</i>	Date item was posted or recorded.
rack	<i>string</i>	Rack location in warehouse.
id_transfer	<i>string</i>	ID for item transfer event.
appv	<i>string</i>	Approval code or status.
oldbatch	<i>string</i>	Previous batch number.
is_damage	<i>bool</i>	Indicates item is damaged.
locked	<i>string</i>	Lock status (e.g., Locked, Unlocked).
type	<i>string</i>	Item type.

Field	Data Type	Description
m	<i>double</i>	Length or other dimension (unit based on context).
rev	<i>null, string</i>	Revision identifier or version.
_id	<i>string</i>	MongoDB document ID.
updatewhen	<i>double, int, null</i>	Last update timestamp.
batch	<i>string</i>	Current batch ID.
updatewho	<i>string</i>	Person who updated the item.
stage	<i>string</i>	Item stage in workflow or process.
updatedate	<i>int</i>	Date item was last updated (epoch).
qty	<i>int</i>	Quantity in stock or transferred.
notes	<i>string</i>	Remarks or comments.
id_quarantine	<i>null, string</i>	Quarantine ID if item is isolated.
createdate	<i>int</i>	Date of item creation/entry (epoch).
description	<i>string</i>	Description of the item.
yard	<i>string</i>	Storage yard or location.
mt	<i>int, double</i>	Weight of item, likely in metric tons.
lot_id	<i>null, string</i>	Lot or grouping ID.
eta	<i>string, Date</i>	Estimated Time of Arrival.
version	<i>string</i>	Item or document version.
owned	<i>string</i>	Owner of the item (internal).

8.69 itemcpo

Store Lot Information, including book value, arrival date, etc.

Table 167. itemcpo table

Field	Data Type	Description
whoreject	<i>string</i>	Condition of the item (e.g., new, used, repaired).
mdr	<i>object, Array</i>	Indicates whether the item is a replacement.
mill	<i>string, null</i>	Transaction identifier.
ismillreject	<i>string</i>	Vessel or transport unit.
rep_mill	<i>bool</i>	Location type (e.g., warehouse, offshore).
updated_at	<i>Date</i>	Time when item was approved (epoch).
vessel	<i>string</i>	Indicates if item is flagged for repair.
ret_rep	<i>bool</i>	Length in feet or another unit.
priceunit	<i>double, string, int, null</i>	Person who approved the item.
is_millcert	<i>bool</i>	Internal allocation or department usage.
appvwho	<i>string</i>	Custom flag for special status or condition.
alias	<i>string</i>	Origin of the item (e.g., vendor).
port	<i>string</i>	Port or logistics hub.
synced_at	<i>Date</i>	Indicates if the item is currently reserved.
status	<i>string</i>	Reference ID from the item collection.
price	<i>string, null, int</i>	Date item was posted or recorded.
ref_number	<i>string</i>	Rack location in warehouse.
appvdate	<i>int</i>	ID for item transfer event.

Field	Data Type	Description
id_item	<i>string</i>	Approval code or status.
book_value_mt	<i>int, double</i>	Previous batch number.
book_value_repaired	<i>bool</i>	Indicates item is damaged.
whocreate	<i>string, null</i>	Lock status (e.g., Locked, Unlocked).
conversion	<i>object, null</i>	Item type.
is_mdr	<i>bool</i>	Length or other dimension (unit based on context).
mandate	<i>string, long, int</i>	Revision identifier or version.
custpo	<i>string, null</i>	MongoDB document ID.
flagRepair	<i>bool</i>	Last update timestamp.
batch	<i>string, null</i>	Current batch ID.
_id	<i>ObjectId, string, null</i>	Person who updated the item.
quarantine	<i>string, Array</i>	Item stage in workflow or process.
whencreate	<i>int</i>	Date item was last updated (epoch).
inspection	<i>Array</i>	Quantity in stock or transferred.
book_value	<i>int, double</i>	Remarks or comments.
old_arrival	<i>int, null, double</i>	Quarantine ID if item is isolated.
arrival	<i>int, null</i>	Date of item creation/entry (epoch).
is_tally	<i>bool</i>	Description of the item.
whenreject	<i>int</i>	Storage yard or location.
pricesource	<i>string, null, int</i>	Weight of item, likely in metric tons.

8.70 iteminspection

Inspection records linked to item reviews.

Table 168. iteminspection table

Field	Data Type	Description
id_item	<i>string</i>	Item ID referencing the main item collection.
batch	<i>string</i>	Batch associated with the inspection.
nextinspectdate	<i>int, double</i>	Next scheduled inspection date (epoch).
id_inspect	<i>string</i>	Unique inspection record ID.
is_new	<i>string</i>	Flag indicating if item is new.
lastinspectdate	<i>double, int, null</i>	Date of last inspection.
owned	<i>string</i>	Owner of the item.
_id	<i>string</i>	MongoDB document ID.
yard	<i>string</i>	Location or storage yard.

8.71 itemtally

Store item metadata per pipe number and heat number.

Table 169. itemtally table

Field	Data Type	Description
inspections	<i>Array</i>	List of inspection records.
expired_date	<i>null</i>	Expiration date (not used or empty).
vendor_order_tally_id	<i>null</i>	Vendor-specific tally order ID.
condition	<i>null, string</i>	Condition of the item.
pipe_no	<i>string</i>	Pipe number or identifier.
mill	<i>string, null</i>	Manufacturing mill name.

Field	Data Type	Description
length_ft	<i>double</i>	Length in feet.
batch	<i>string</i>	Batch identifier.
_id	<i>string, ObjectId</i>	Document ID.
length	<i>double, string, null</i>	Length value (flexibly typed).
lastpostdate	<i>int</i>	Last posting date (epoch).
lastpostwho	<i>string</i>	User who posted the last update.
length_type	<i>string</i>	Type of length measurement (e.g., nominal, actual).
barcode	<i>string, null</i>	Barcode value.
is_expiration_checked	<i>null</i>	Flag if expiration check is enabled.
heat	<i>string</i>	Heat number or batch ID from manufacturing.
status	<i>string, null</i>	Item status (e.g., available, scrapped).
length_m	<i>double</i>	Length in meters.
id_item	<i>string</i>	ID referencing the item.
yard	<i>null, string</i>	Yard or storage location.
serial_number	<i>null</i>	Serial number of the item.
owned	<i>string, null</i>	Owner of the item (e.g., department or client).

8.72 itemtrans

Store all Tubestream transactions such as transfer, inspection, callloff, etc

Table 170. Itemtrans table

Field	Data Type	Description
eta	<i>int, long</i>	Estimated time of arrival
date_bill	<i>int, long, null</i>	Date of billing
cnclReason	<i>string</i>	Reason for cancellation
total_ft	<i>int, double</i>	Total in feet (e.g. length)
production_status	<i>string</i>	Current production status
idtrans	<i>string</i>	Transaction ID
ft	<i>double, int</i>	Quantity in feet
location	<i>null, string</i>	Physical or logical location
dp	<i>string</i>	Down payment or delivery point
isCompleteReceive	<i>string</i>	Status if receiving is completed
notes_from	<i>string</i>	Notes from sender/originator
rejected	<i>bool</i>	Rejected status flag
appvwho	<i>string, null</i>	Who approved the item
allocation	<i>string</i>	Allocation ID or description
flag	<i>string</i>	Status or control flag
incoming	<i>Array</i>	List of incoming items
isCompleteDispatch	<i>string</i>	Status if dispatch is completed
file_pdf	<i>string</i>	Path or name of attached PDF

Field	Data Type	Description
notes_to	<i>string</i>	Notes to recipient
status	<i>string, int</i>	Status code or description
internalNotes	Array	Internal notes list
approvedate	<i>int</i>	Approval date (timestamp)
approvewho	<i>string</i>	Who approved the item
onorder_recv	<i>int</i>	Quantity on order received
screening_number	<i>int</i>	Screening or verification number
invoicedWhen	<i>int</i>	Timestamp when invoiced
truck_company	<i>null</i>	Trucking company used
appv	<i>string</i>	Approval status or stage
total_received	<i>int</i>	Total quantity received
qsfic_files	Array	Files related to QSFIC (audit/logs)
incomingtype	<i>string</i>	Type/category of incoming item
closedate	<i>int</i>	Date the transaction was closed
invoicedwhen	<i>int, double</i>	Timestamp of invoice
itemname	<i>string</i>	Name of item transacted
isInspected	<i>bool</i>	Inspection status
whenclosed	<i>int, double</i>	Timestamp of closing
bl	<i>null</i>	Bill of lading or reference
dm	<i>string</i>	Decision maker or dispatch manager
operator_name	<i>string</i>	Name of machine/human operator

Field	Data Type	Description
rejectwho	<i>string</i>	Who rejected the item
locked	<i>string, null</i>	Lock flag or info
port	<i>string</i>	Port of entry or departure
rig	<i>string</i>	Rig or drilling unit
axdate	<i>int</i>	AX system date
cncldate	<i>int</i>	Date of cancellation
custpo	<i>string, null</i>	Customer PO number
cp	<i>string</i>	Collection Point
qty	<i>string, int</i>	Quantity of item
source_port	<i>null, string</i>	Port of origin/source
pod	<i>string</i>	Proof of delivery or port of discharge
total_mt	<i>int, double</i>	Total metric tons
no	<i>string, null</i>	Document or transaction number
etd_sort	<i>Date</i>	Expected Time of Departure (sortable)
rjtReason	<i>string</i>	Rejection reason
updated_at	<i>Date</i>	Timestamp of last update
condition	<i>string</i>	Condition or quality
cab_number	<i>null</i>	Cab or container number
vessel	<i>null, string</i>	Vessel name
completed	<i>string, null</i>	Completion status
invoiced	<i>string</i>	Invoice status

Field	Data Type	Description
mill	<i>string, null</i>	Mill or factory identifier
dd_files	Array	Due diligence or delivery documents
journaled_date	<i>int</i>	Journal entry timestamp
ecd	<i>int, null</i>	Estimated completion/delivery date
date_ordered	<i>int, null</i>	Order placement timestamp
customer	<i>string</i>	Customer name or ID
type_trf	<i>string</i>	Type of transfer
idYard	<i>string</i>	Yard identifier
pono	<i>string</i>	Purchase order number
closeReason	<i>string</i>	Reason for closing transaction
inv	<i>null, Array, object</i>	Invoice data or references
pic	<i>string</i>	Person in charge
deliveryterm	<i>null, string</i>	Delivery terms (e.g. FOB, CIF)
isDispatch	<i>string</i>	Dispatch status
journaled	<i>string</i>	Journal entry status
isEdit	<i>bool</i>	Editable flag
allocation_customer	<i>null, string</i>	Allocation for customer
allocation_opt	<i>string</i>	Optional allocation details
parent	<i>string</i>	Parent transaction or relation
id_inspection	<i>string</i>	Inspection identifier

Field	Data Type	Description
isReceive	<i>string</i>	Receive status
requester	<i>null, string</i>	Person who requested the item
vendor_account	<i>null, string</i>	Vendor account number
transport	<i>string</i>	Transport method or carrier
updateid	<i>string, null</i>	ID of updater
cancelReason	<i>string</i>	Cancellation reason
vendor_reff	<i>null, string</i>	Vendor reference number
total_balance	<i>int</i>	Remaining balance
qsf	<i>Array, null</i>	Quality summary files
stages	<i>object</i>	Process stages metadata
from	<i>string</i>	Sender or origin
batch	<i>string</i>	Batch identifier
etd	<i>string, int</i>	Estimated departure time
updatewhen	<i>int</i>	When last updated
journaled_who	<i>string</i>	Who made journal entry
createwho	<i>string</i>	Who created the record
allowance	<i>int, double</i>	Allowance (e.g., excess tolerance)
notes	<i>string, null, object</i>	Notes or comments
cc	<i>string</i>	Carbon copy or email CC
book_value	<i>int, double, null</i>	Book value of item

Field	Data Type	Description
received	<i>string</i>	Receive status or notes
createdate	<i>int, null, double</i>	Creation date
nama_loader	<i>string</i>	Name of loader
closedWho	<i>string</i>	Who closed the transaction
additional_transport	Array	Additional transport info
vessel_name	<i>string</i>	Name of vessel
files_backup	Array	Backup files list
owned	<i>string</i>	Owner of item
closefiles	Array	Files related to closing
truckload	Array	Truckload details
receive_no	<i>string</i>	Receive document number
totalprice	<i>double, int</i>	Total transaction price
all_cust_owned	<i>string</i>	All customer ownership flag
cnclfiles	Array	Cancellation files
ponoFiles	Array	PO number-related files
allocationv2	<i>string</i>	Allocation version 2
appvwhen	<i>string, int</i>	Approval timestamp
is_manual_closed	<i>bool</i>	Manually closed flag
trust_limit_balance_snapshot	<i>null</i>	Snapshot of trust limit balance
operatorSign	<i>string, null</i>	Operator's signature

Field	Data Type	Description
closewho	<i>string</i>	Who closed the transaction
closeAttachment	Array	Close-related attachments
photo	Array	Related photos
well	<i>string</i>	Oil well identifier
journalId	<i>string</i>	Journal entry ID
idDispatch	<i>string</i>	Dispatch ID
source	<i>string, null</i>	Data or item source
insp_type	<i>string</i>	Type of inspection
rejectdate	<i>int</i>	Date of rejection
trfType	<i>string</i>	Type of transfer
closed	<i>string</i>	Closed status
id_item	<i>string</i>	Item ID
need_approve	<i>bool</i>	Approval needed flag
coffType	<i>string</i>	Calloff Type
destination	<i>string</i>	Delivery destination
unitprice	<i>int, double, null</i>	Unit price of item
rack	Array	Rack or shelf information
destination_port	<i>string</i>	Port of destination
noDispatch	<i>string</i>	Dispatch number
tax_invoice	<i>object, Array</i>	Tax invoice details

Field	Data Type	Description
is_notified_quality	<i>bool</i>	Quality notification status
notesC	<i>string</i>	Customer Notes
items_raw	Array	Raw item data
mode_delivery	<i>null, string</i>	Mode of delivery (e.g., sea, air)
delivery_term	<i>string, null</i>	Delivery terms
id_loadout_android	<i>string</i>	Android loadout ID
isExistingInv	<i>bool</i>	Existing invoice status
whencancel	<i>int</i>	Cancellation timestamp
bottom_allowance	<i>int, double</i>	Bottom tolerance allowance
notesCr	<i>string</i>	Credit or cancellation notes
incidental	<i>bool</i>	Is incidental cost included
others_files	Array	Other files
updatedate	<i>int, double, null</i>	Last update date
isimport	<i>bool</i>	Import flag
vessel_code	<i>null, string</i>	Vessel code
approve	<i>string</i>	Approval status
createid	<i>string</i>	Creator's ID
company_detail	<i>string</i>	Company details
yard	<i>string</i>	Storage or dockyard
mill_tag	<i>string, null</i>	Mill identifier tag

Field	Data Type	Description
cancelwho	<i>string</i>	Who canceled the transaction
id_quarantine	<i>string</i>	Quarantine ID
date_shipping	<i>long, null, int</i>	Date of shipment
lot_id	<i>null, string</i>	Lot number
tlc_no	<i>string</i>	TLC document number
type_loc	<i>string</i>	Type of location
mt	<i>double, int, string</i>	Metric tons (can vary in format)
items	<i>Array, object</i>	Item details
appvReason	<i>string</i>	Reason for approval
unit	<i>string</i>	Unit of measurement
is_deleted	<i>bool</i>	Deleted flag
invoicedWho	<i>string</i>	Who invoiced
company_create	<i>string</i>	Company that created the record
priceunit	<i>double, null, int</i>	Unit price value
syncStatus	<i>string</i>	Sync status for data
nama_driver	<i>string</i>	Driver's name
isCredit	<i>string</i>	Credit flag
canceledate	<i>int</i>	Date of cancellation
postingdate	<i>int</i>	Posting timestamp

Field	Data Type	Description
book_value_mt	<i>int, double, null</i>	Book value in metric tons
ttd_driver	<i>string, object</i>	Driver's signature
total_joint	<i>int</i>	Total joint units/items
vessel_mmsi	<i>null, string</i>	Vessel MMSI number
cancel	<i>string</i>	Cancel status
id_transfer	<i>string</i>	Transfer ID
tally	<i>string, Array</i>	Tally or log info
to	<i>string</i>	Recipient or destination
ttd_loader	<i>string, object</i>	Loader's signature
custreff	<i>string</i>	Customer reference
croninvoice	<i>string</i>	Invoice batch or cron flag
finalized	<i>string</i>	Finalization status
item	<i>null, Array</i>	Item list or data
type	<i>string</i>	Type of transaction
closedWhen	<i>int</i>	When closed
operatorName	<i>string</i>	Operator's name
trfTypeTo	<i>string, null</i>	Transfer type destination
files	<i>Array</i>	Files attached
_id	<i>string</i>	Document ID (MongoDB)

Field	Data Type	Description
upper_allowance	<i>int, double</i>	Top-level tolerance
invoicedwho	<i>string</i>	Who did the invoicing
updatewho	<i>string</i>	Who updated the record
operator_sign	<i>string, object</i>	Operator's signature info
arrivaldate	<i>int</i>	Arrival date
tlc_files	Array	TLC-related documents
transporter	<i>string</i>	Transport company or person
is_from_tablet	<i>bool</i>	Originated from tablet
carrier	<i>string</i>	Carrier name
oldFileList	<i>string</i>	Legacy or previous file list
total_price	<i>null, double, int</i>	Total price
closeDesc	<i>string</i>	Close description
version	<i>string, int</i>	Version info
qty_onorder	<i>int</i>	Quantity on order
axaptano	<i>string</i>	AXAPTA document number
email_create	<i>string</i>	Creator's email
quarantined_qty	<i>string</i>	Quarantined quantity
total_value	<i>double</i>	Total value of transaction
items_in	Array	Incoming items list

8.73 itemvims

Store all account balance metadata per item including average price, etc.

Table 171. itemvims table

Field	Data Type	Description
is_contractual	<i>bool</i>	Indicates if the item is under contract
client	<i>string</i>	Name of the client
customer	<i>string</i>	Associated customer
market	<i>string</i>	Market classification or segment
ct	<i>string</i>	Casing / Tubing Type
priceunit	<i>string, null</i>	Unit of pricing (e.g., per item, per kg)
probability	<i>string</i>	Probability or likelihood score/description
aging	<i>string, object</i>	Time since creation or update; may include detailed structure
contractual_item	<i>null</i>	Placeholder for future contractual item info
last_avgprice	<i>null, object</i>	Historical average price data
cust_item_desc	<i>null</i>	Customer-specific item description
id_item	<i>string, null</i>	Item identifier
lead	<i>double, int</i>	Lead time (e.g., in days)
bu	<i>string</i>	Business unit
opening_mo	<i>int</i>	Initial stock or value at start of month
axaptano	<i>string</i>	ERP reference number (e.g., from Axapta)

Field	Data Type	Description
conversion	<i>object</i>	Conversion data between units
avgprice	<i>null, object</i>	Current average price data
customer_item_no	<i>string, null</i>	Customer's internal item number
duration	<i>null</i>	Contract or usage duration
active	<i>bool</i>	Whether the item is active
batchprice	<i>object</i>	Pricing info by batch
opening	<i>int</i>	Opening inventory or amount
idop	<i>string</i>	Operator or item ID
consignment_type	<i>string, null</i>	Type of consignment
whoupdate	<i>string</i>	User who last updated the record
_id	<i>string</i>	Unique document ID
whenupdate	<i>int</i>	Timestamp of last update
whencreate	<i>int</i>	Timestamp of creation
fc_acc	<i>object</i>	Forecast Accuracy
opening_year	<i>int</i>	Initial stock or value at the start of year

8.74 ivms

Store information about Intelligent Vehicle Monitoring System.

Table 172. ivms table

Field	Data Type	Description
transporter	<i>string</i>	Transport company or vehicle identifier

Field	Data Type	Description
ivms_month	<i>int</i>	Month of IVMS data
distance	<i>double</i>	Distance traveled (e.g., in km)
occurrence_1	<i>int</i>	Count of first type of IVMS event
fname	<i>object</i>	Possibly filename metadata or data bundle
occurrence_4	<i>int</i>	Count of fourth type of IVMS event
occurrence_5	<i>int</i>	Count of fifth type of IVMS event
ivms_year	<i>int</i>	Year of IVMS data
occurrence_2	<i>int</i>	Count of second type of IVMS event
folder	<i>string</i>	Folder or directory name
contract	<i>null</i>	Placeholder for contract reference
occurrence_3	<i>int</i>	Count of third type of IVMS event
group	<i>string</i>	Group or classification of IVMS record
_id	<i>ObjectId</i>	MongoDB document ID
ivms_time	<i>int</i>	Time duration or timestamp of record
lastpostdate	<i>int</i>	Last posted timestamp
lastpostwho	<i>string</i>	User who last posted the record

8.75 jira_issue_tickets

Jira tickets for managing development or support.

Table 173. jira_issue_tickets

Field	Data Type	Description
updated_at	<i>int</i>	Timestamp of last update
reporter_name	<i>string</i>	Name of the person who reported the issue
created_at	<i>int</i>	Timestamp of ticket creation
status	<i>string</i>	Status of the issue (e.g., Open, Closed)
ticket_number	<i>string</i>	Unique ticket identifier
description	<i>string</i>	Detailed description of the issue
urgency	<i>string</i>	Urgency level (e.g., Low, Medium, High)
title	<i>string</i>	Title or subject of the ticket
impact	<i>string, null</i>	Business or system impact
email	<i>string</i>	Reporter's email address
_id	<i>string</i>	Unique document ID

8.76 jira_log

Log of changes to Jira tickets and status history.

Table 174. jira_log

Field	Data Type	Description
_id	<i>string</i>	Unique identifier for the log document
type	<i>string</i>	Type/category of the JIRA log event

Field	Data Type	Description
date_int	<i>int</i>	Date stored as an integer (likely UNIX timestamp)
date	<i>Date</i>	Human-readable date of the log event
who	<i>string</i>	User who performed the action
payload	<i>Array</i>	List or structure of data related to the log entry

8.77 lessonlearned

Lessons learned and captured from past operations.

Table 175. lessonlearned table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier for the document
name	<i>string</i>	Name/title of the lesson
actions	<i>string</i>	Actions taken in response to the lesson
no	<i>string, int</i>	Reference number or identifier
access	<i>string</i>	Access level or visibility scope
lessonlearned	<i>string</i>	Main content or lesson description
fname	<i>Array</i>	List of attached files or file names
shareto	<i>Array</i>	List of users or departments the lesson is shared with
desc	<i>string</i>	Additional description or context
lastpostdate	<i>int</i>	Timestamp of the last update or posting

Field	Data Type	Description
lastpostwho	<i>string</i>	User who last updated or posted the lesson
classification	<i>string</i>	Category or classification of the lesson
id_spr	<i>string</i>	Reference to another document or identifier (e.g., SPR ID)

8.78 li_currency

Currency master used in 3rd party instruction.

Table 176. li_currency table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier for the currency
name	<i>string</i>	Name of the currency (e.g., USD, EUR)

8.79 li_delivery

Delivery master used in 3rd party instruction.

Table 177. li_delivery table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
name	<i>string</i>	Name of delivery method or type

8.80 li_recipients

Recipient information used in 3rd party instruction.

Table 178. li_recipients table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier

Field	Data Type	Description
name	<i>string</i>	Recipient's name

8.81 li_uom

Units of measure used in 3rd party instruction.

Table 179. li_uom table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
name	<i>string</i>	Unit of Measure name (e.g., KG, PCS)

8.82 loadout_tally

Stores tally sheet data for outbound shipments or yard loadouts.

Table 180. loadout_tally table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
id_item	<i>string</i>	Linked item identifier
id_transfer	<i>string</i>	Transfer document ID
batch	<i>string</i>	Batch identifier
pipeNo	<i>string</i>	Pipe number
heatNo	<i>string</i>	Heat number
length	<i>string</i>	Length of item (typically in meters)
owned	<i>string</i>	Owner information
condition	<i>string</i>	Condition description
createdate	<i>int</i>	Date of creation (UNIX timestamp)
appvwhen	<i>int</i>	Approval timestamp

Field	Data Type	Description
appvwho	<i>string</i>	Approved by (user ID)

8.83 locode

Stores UN/LOCODEs or other standardized location codes.

Table 181. locode table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
locode	<i>string</i>	UN/LOCODE
name	<i>string</i>	Location name
country_code	<i>string</i>	ISO country code
country_name	<i>string</i>	Country name
function	<i>string</i>	LOCODE function code
coordinates	<i>string</i>	Latitude and longitude
status	<i>string</i>	Active/inactive status
sub_div	<i>string</i>	Subdivision code
created_at	<i>string</i>	Creation timestamp
updated_at	<i>string</i>	Update timestamp

8.84 log_api

Logs API request and response data for troubleshooting or analytics.

Table 182. log_api table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
type	<i>string</i>	API request type

Field	Data Type	Description
function	<i>string</i>	Function called
when	<i>int</i>	Timestamp of API call
when_readable	<i>string</i>	Readable date/time
yard	<i>string, null</i>	Associated yard
q	<i>object, Array</i>	Query parameters or body

8.85 logs

General-purpose log table for capturing system events.

Table 183. logs table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
who	<i>string</i>	User who performed the action
when	<i>int</i>	Timestamp
url	<i>string</i>	Accessed URL
controller	<i>string</i>	Controller or module name
action	<i>string</i>	Action performed
folder	<i>string</i>	Folder path (if applicable)
id	<i>string, null</i>	Related record ID
docname	<i>string, null</i>	Document name (if applicable)
module	<i>string</i>	Module or system name

8.86 mdr

Store Manufacturing data record for each batch.

Table 184. mdr table

Field	Data Type	Description
_id	<i>string</i>	Manufacturing Data Record
id_item	<i>string</i>	Item ID
batch	<i>string</i>	Batch number
arrival	<i>int</i>	Arrival timestamp
status	<i>string</i>	Status of MDR
is_mdr	<i>bool</i>	Whether it is an MDR document
is_millcert	<i>bool</i>	Whether it is a mill certificate
is_tally	<i>bool</i>	Whether tally is complete
is_approved	<i>bool</i>	Whether approved
is_complete	<i>bool</i>	Whether complete
appvwho	<i>string, null</i>	Approved by
appvdate	<i>int, null</i>	Approval timestamp
whoreject	<i>string, null</i>	Rejected by
whenreject	<i>int, null</i>	Rejection timestamp
completewho	<i>string, null</i>	Completed by
completedate	<i>int, null</i>	Completion date
complete_comment	<i>string, null</i>	Comment on completion
ref_number	<i>string, null</i>	Reference number
alias	<i>string, null</i>	Alias name

Field	Data Type	Description
mill	<i>string, null</i>	Mill name
ismillreject	<i>string</i>	Mill reject status
percent	<i>string, null</i>	Completion percent
conversion	<i>object</i>	Conversion details
inspection	<i>Array</i>	Inspection data
mdr	<i>Array</i>	MDR document data
production_suffix	<i>null</i>	Suffix if any

8.87 mdr_folder

Organizes MDR documents into folders or logical groupings.

Table 185. mdr_folder table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
code	<i>string</i>	Folder code
name	<i>string</i>	Folder name
vendor_portal_name	<i>string</i>	Name in vendor portal

8.88 merchandise_balance

Contains stock balance data for merchandise items.

Table 186. merchandise_balance table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
id_item	<i>string</i>	Item ID
item_number	<i>string</i>	Item number or SKU

Field	Data Type	Description
qty	<i>int</i>	Quantity in stock
value	<i>double</i>	Total value in currency
value_mt	<i>double</i>	Value in metric tons
book_value	<i>double, int</i>	Booked value in currency
book_value_mt	<i>double, int</i>	Booked value in metric tons
warehouse	<i>string</i>	Warehouse location
customer	<i>string</i>	Customer name or ID
mill	<i>string</i>	Mill source
allocation	<i>string</i>	Allocation category or usage
batch	<i>string</i>	Batch number
source	<i>string</i>	Source of data
aging_val	<i>int</i>	Aging value (e.g., in days)
aging_cat	<i>int</i>	Aging category (e.g., 0-6mo, 6-12mo)
whoupdate	<i>string</i>	Last updated by
whenupdate	<i>int</i>	Last update timestamp

8.89 merchandise_balance_ceiling

Stores maximum allowed stock limits for merchandise.

Table 187. merchandise_balance_ceiling table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
group	<i>string</i>	Group/category name

Field	Data Type	Description
category	Array	Associated item categories
gbc	<i>int</i>	Gross balance ceiling
mbc	<i>int</i>	Merchandise balance ceiling
gbc_mt	<i>int</i>	Gross balance ceiling in metric tons
mbc_mt	<i>int</i>	Merchandise balance ceiling in metric tons

8.90 merchandise_balance_comm

Tracks merchandise balances committed to specific customers or contracts.

Table 188. merchandise_balance_comm table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
id_item	<i>string</i>	Item ID
customer	<i>string</i>	Customer name or ID
mb_cat	<i>string</i>	Merchandise balance category
qty	<i>int</i>	Original quantity
qty_edit	<i>int</i>	Adjusted quantity
factor	<i>double, int</i>	Conversion or scaling factor

8.91 merchandise_balance_comm_raw

Raw data for committed merchandise balances, typically before processing.

Table 189. merchandise_balance_comm_raw table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier

Field	Data Type	Description
id_item	<i>string</i>	Item ID
customer	<i>string</i>	Customer name or ID
mb_cat	<i>string</i>	Merchandise balance category
qty	<i>int</i>	Quantity value (raw/unprocessed)

8.92 merchandise_balance_log

Logs changes or transactions affecting merchandise balances.

Table 190. merchandise_balance_log table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
status	<i>string</i>	Status of the log (e.g., PUBLISHED)
period	<i>string</i>	Reporting period (e.g., 2024-Q1)
whocreate	<i>string</i>	Created by
whopublish	<i>string</i>	Published by
whencreate	<i>int</i>	Created timestamp
publishedat	<i>int</i>	Published timestamp

8.93 merchandise_balance_raw

Unprocessed stock balance data for merchandise items.

Table 191. merchandise_balance_raw table

Field	Data Type	Description
_id	<i>ObjectId</i>	MongoDB document ID
id_item	<i>string</i>	Item ID
item_number	<i>string</i>	Item number or SKU

Field	Data Type	Description
qty	<i>int</i>	Quantity
value	<i>double</i>	Value in currency
value_mt	<i>double</i>	Value in metric tons
book_value	<i>double, int</i>	Book value in currency
book_value_mt	<i>double, int</i>	Book value in metric tons
warehouse	<i>string</i>	Warehouse code
mb_cat	<i>string</i>	Merchandise balance category
transaction_date	<i>int</i>	Transaction timestamp
year	<i>int</i>	Year of transaction
month	<i>int</i>	Month of transaction
mill	<i>string</i>	Mill source
allocation	<i>string</i>	Allocation type
batch	<i>string</i>	Batch number
source	<i>string</i>	Data source system

8.94 merchandise_balance_raw_hist

Historical records of raw merchandise balance data for tracking trends.

Table 192. merchandise_balance_raw_hist table

Field	Data Type	Description
source	<i>string</i>	Source of data
book_value	<i>double, int</i>	Book value in currency
qty	<i>int</i>	Quantity

Field	Data Type	Description
id_item	<i>string</i>	Item ID
book_value_mt	<i>double, int</i>	Book value in metric tons
item_number	<i>string</i>	Item number or SKU
warehouse	<i>string</i>	Warehouse code
mb_cat	<i>string</i>	Merchandise balance category
transaction_date	<i>int</i>	Timestamp of transaction
year	<i>int</i>	Transaction year
month	<i>int</i>	Transaction month
mill	<i>string</i>	Mill source
allocation	<i>string</i>	Allocation type
batch	<i>string</i>	Batch number
_id	<i>ObjectId</i>	Unique document ID
value_mt	<i>double</i>	Value in metric tons
value	<i>double</i>	Value in currency

8.95 merchandise_balance_report

Generates summary or detailed reports on merchandise balances.

Table 193. chandise_balance_report table

Field	Data Type	Description
total_gross_balance	<i>string</i>	Total gross balance
total_mb_ceiling	<i>string</i>	Total merchandise balance ceiling
total_mto	<i>string, double</i>	Total made-to-order items

Field	Data Type	Description
total_pc	<i>string</i>	Total physical count
total_lte_24mo	<i>string</i>	Total inventory ≤ 24 months
total_result	<i>string</i>	Final result or status
total_gross_balance_ceiling	<i>string</i>	Gross balance ceiling total
total_intransit	<i>string</i>	Inventory in transit
period	<i>string</i>	Reporting period
data	Array	Array of report data
total_cos	<i>string</i>	Total cost of sales
total_pi	<i>string</i>	Total pending invoices or PIs
total_sales_commitment	<i>string</i>	Total sales commitments
total_over_36mo	<i>string</i>	Inventory over 36 months
total_reserve_wo	<i>string</i>	Reserved without orders
total_total	<i>string</i>	Grand total
total_mb	<i>string</i>	Total merchandise balance
type	<i>string</i>	Report type (e.g., summary)
total_over_24mo	<i>string</i>	Inventory over 24 months
total_lte_12mo	<i>string</i>	Inventory ≤ 12 months
_id	<i>string</i>	Unique identifier
total_lte_6mo	<i>string</i>	Inventory ≤ 6 months
total_bs_inv	<i>string</i>	Total balance sheet inventory
total_expected_pl	<i>string</i>	Expected profit/loss

8.96 merchandise_balance_sc

Merchandise balance details filtered or grouped by supply chain classification.

Table 194. merchandise_balance_sc table

Field	Data Type	Description
id_item	<i>string</i>	Item ID
sc_usd	<i>double, null</i>	Sales commitment value in USD
sc_mt	<i>double, null</i>	Sales commitment in metric tons
qty	<i>int</i>	Quantity
mb_cat	<i>string</i>	Merchandise balance category
_id	<i>string</i>	Unique document ID

8.97 mill

Stores mill or manufacturer information for product origin tracking.

Table 195. mill table

Field	Data Type	Description
country	<i>string</i>	Country of the mill
detail	<i>object</i>	Additional mill details
country_states	<i>string</i>	Sub-region or state
tag	<i>Array</i>	Tags or labels
access	<i>string</i>	Access permissions
activation_mill	<i>string</i>	Activation status
ams_id	<i>string</i>	AMS system ID
api_text	<i>string</i>	API text configuration

Field	Data Type	Description
store_stock	<i>string</i>	Type of store stock
name	<i>string</i>	Mill name
api	<i>string</i>	API endpoint
city	<i>string</i>	City name
category	<i>string</i>	Mill category or classification
apild	<i>int</i>	API numeric identifier
id	<i>string</i>	Mill ID
shareto	Array	Shared access list
created_at	Date	Creation timestamp
updated_at	Date	Last updated timestamp
state	<i>string</i>	State/Province
grab_status	<i>string</i>	Data grabbing status
type	Array	Mill types or tags
_id	<i>string</i>	Document ID
api_sel	<i>string</i>	Selected API option
active	<i>string</i>	Active/inactive status
tier	<i>string</i>	Tier level

8.98 milladvisor

Logs or recommendations from mill advisors on production or quality.

Table 196. milladvisor table

Field	Data Type	Description
detail	<i>object</i>	Additional advisor details
country	<i>string</i>	Country
tag	<i>Array</i>	Tags or labels
country_states	<i>string</i>	Sub-region or state
access	<i>string</i>	Access level
activation_mill	<i>string</i>	Activation status
ams_id	<i>string</i>	AMS system ID
api_text	<i>string</i>	API text configuration
store_stock	<i>string</i>	Store stock setting
name	<i>string</i>	Advisor name
api	<i>string</i>	API endpoint
city	<i>string</i>	City
id	<i>string</i>	Advisor or mill ID
apild	<i>int</i>	API ID
shareto	<i>Array</i>	Shared access targets
category	<i>string</i>	Category classification
created_at	<i>Date</i>	Creation timestamp
state	<i>string</i>	State/Province
updated_at	<i>Date</i>	Last modification date

Field	Data Type	Description
grab_status	<i>string</i>	Crawler status
type	<i>Array</i>	Advisor/mill types
_id	<i>string</i>	Document ID
api_sel	<i>string</i>	API selection
active	<i>string</i>	Active/inactive flag
tier	<i>string</i>	Tier grouping

8.99 moc

Stores Management of Change (MOC) request records, including workflow stages, assigned users, and history approval.

Table 197. moc table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier for the MOC request
accepted.user_id	<i>string</i>	ID of the user who accepted the MOC
accepted.date_accepted	<i>int</i>	Timestamp when the MOC was accepted
assign_acknowledger	<i>array</i>	List of users assigned as acknowledgers
assign_approver	<i>array</i>	List of users assigned as approvers
assign_evaluators	<i>array of objects</i>	List of evaluator assignments, see nested breakdown below

Field	Data Type	Description
attachments	<i>array</i>	Files attached to the MOC
category	<i>string</i>	Category ID reference
createdate	<i>int</i>	MOC creation timestamp
createwho	<i>string</i>	ID of the user who created the MOC
date_completed	<i>int</i>	Timestamp of MOC completion
date_opened_required.open	<i>int</i>	MOC opening date (timestamp)
date_opened_required.required	<i>int</i>	Required deadline for opening MOC
description_of_category	<i>string</i>	Detailed description of the category
level_of_risk	<i>string</i>	Risk level ID reference
moc_manager	<i>string</i>	ID of the MOC manager
moc_summary._id	<i>string</i>	ID of the summary (unused/empty)
moc_summary.description	<i>string</i>	Summary description
moc_summary.attachments	<i>array</i>	Summary attachments
moc_summary.submitter.user_id	<i>string/null</i>	ID of the summary submitter
moc_summary.submitter.date_submitted	<i>int/null</i>	Submission timestamp

Field	Data Type	Description
no	<i>string</i>	MOC number/identifier
potential_lost_cost.estimate_total_cost	<i>double</i>	Estimated total cost
potential_lost_cost.currency	<i>string</i>	Currency code
potential_lost_cost.actual_costs	<i>array</i>	List of actual costs (empty)
potential_lost_cost.actual_total_cost	<i>double</i>	Actual total cost
potential_lost_cost.percent_of_estimated_cost	<i>double</i>	Percentage of estimated cost
progress	<i>int</i>	Current progress status
reason_of_change	<i>string</i>	Reason for the MOC
rejected.user_id	<i>string/null</i>	User ID who rejected (if any)
rejected.reason	<i>string/null</i>	Reason for rejection
rejected.date_rejected	<i>int/null</i>	Rejection timestamp
rejected.attachments	<i>array</i>	Rejection attachments
scope_of_change	<i>string</i>	Detailed scope (internal/external)
status	<i>int</i>	MOC status code
title	<i>string</i>	Title of the MOC
type_of_change	<i>string</i>	Type of change ID reference

Field	Data Type	Description
updatedate	<i>int</i>	Last update timestamp
accessibility	<i>string</i>	Accessibility status (e.g., "closed")

8.100 moc_dropdown

Reference list for dropdown values used within the MOC interface, supporting consistent field selection (e.g., types, categories).

Table 198. moc_dropdown table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier for the risk level entry
name	<i>string</i>	Name of the risk level (e.g., "Low")
type	<i>string</i>	Type classification (e.g., "moc_level_of_risk")

8.101 notification

Stores user or system-triggered notification records.

Table 199. notification table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
userId	<i>string, null</i>	User ID
user	<i>string, null</i>	Username or user reference
read	<i>bool, Array</i>	Read status flag or history
hide	<i>Array</i>	Visibility status or hidden from list
message	<i>string</i>	Notification message content

Field	Data Type	Description
access	<i>string</i>	Access level or permission context
timestamp	<i>int</i>	Notification timestamp
url	<i>string, null</i>	Target URL when notification is clicked
shareTo	<i>string, null, Array</i>	Users or groups notification is shared to
no	<i>string, null</i>	Notification number/reference

8.102 od

Store master Outer Diameter for each Item.

Table 200. od table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
name	<i>string</i>	Name of the OD (outer diameter)

8.103 openorder_vessel_lotid

Stores data of open orders linked to vessel shipments and lot IDs.

Table 201. od table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
vessel_name	<i>string</i>	Name of the vessel

8.104 ordervims

Store account balance data.

Table 202. ordervims table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
ro_notes	<i>string, bool</i>	Remarks or notes
setup_date	<i>int</i>	Setup or creation date
cr_cust	<i>string, double, int</i>	Credit Customer Owned
op_cust	<i>double, int</i>	Opening Balance Customer Owned
year	<i>double, int</i>	Year
remarks	<i>string</i>	General remarks
client	<i>string</i>	Client name
ao	<i>double, int</i>	Approved Order
customer	<i>string, null</i>	Customer name or ID
commit_price	<i>double, int</i>	Committed price
au_total	<i>double, int</i>	Actual Usage Total
eom	<i>double, int</i>	End of month value
au_cust	<i>double, int</i>	Actual Usage Customer Owned
fc	<i>double, int, string</i>	Forecast commitment
op_total	<i>double, int</i>	Opening Balance Total
au_cust_dmg	<i>int</i>	Actual Usage Customer Owned Damaged

Field	Data Type	Description
is_legacy	<i>bool</i>	Whether record is from legacy system
ro_date	<i>int</i>	Recommended Order Date
price	<i>double, int</i>	Price
signoff_date	<i>double, int</i>	Signoff date
cr_cust_dmg	<i>int</i>	CR damaged quantity for customer
id_item	<i>string, null</i>	Item ID
ro_date2	<i>string, bool</i>	Alternative RO date
bu	<i>string</i>	Business unit
is_op	<i>bool</i>	Opening Balance
op	<i>double, int</i>	OP (operational performance) value
axaptano	<i>string</i>	Axapta item number
is_opcust	<i>bool</i>	OP for customer flag
eom_cust	<i>double, int</i>	End-of-month value for customer
transaction_date	<i>double, int, bool</i>	Date of transaction
month	<i>double, int</i>	Transaction month
au	<i>double, string, int</i>	Actual Usage
cr	<i>double, int</i>	Credit
cr_total	<i>double, int</i>	Credit Total
sales_price	<i>double</i>	Sales price
batch_price	<i>double</i>	Price per batch

Field	Data Type	Description
eom_cust_dmg	<i>int</i>	EOM damage for customer
whenupdate	<i>int</i>	Last update timestamp
op_cust_dmg	<i>int</i>	Opening Balance Customer Owned Damage
commit_mt	<i>double, int</i>	Committed metric tons
qty	<i>int</i>	Quantity ordered
avg_price	<i>double, int</i>	Average price
ao_total	<i>double, int</i>	AO total value
total_price	<i>double, int</i>	Total price
is_opcust_dmg	<i>bool</i>	OP customer damage flag
commit_qty	<i>double, int</i>	Committed quantity
eom_total	<i>double, int</i>	Total EOM quantity
signoff_price	<i>double, int</i>	Signoff price
credit_price	<i>double</i>	Price on credit
ro	<i>int</i>	Recommended Order

8.105 pipesales_bu_setup

Stores business unit configurations specific to the Pipesales platform.

Table 203. pipesales_bu_setup table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
pipesales_id	<i>int</i>	Pipe sales BU ID
pipesales_name	<i>string</i>	Name of the pipe sales BU

Field	Data Type	Description
pipesales_contact_no	<i>string</i>	Contact number
pipesales_email	<i>string</i>	Contact email address
pipesales_seller_id	<i>int</i>	Seller ID
pipesales_seller_name	<i>string, null</i>	Name of the seller
tubestream_id	<i>string, null</i>	Associated Tubestream ID
tubestream_seller_parent_id	<i>string, null</i>	Parent seller ID from Tubestream
tubestream_seller_parent_name	<i>string, null</i>	Parent seller name from Tubestream
pipesales_role	<i>string</i>	Role in pipe sales
pipesales_country_id	<i>int, null</i>	Country ID
pipesales_country_name	<i>string, null</i>	Country name
type	<i>string</i>	Type or category
tubestream_seller_id	<i>Array, null</i>	List of seller IDs from Tubestream
is_default	<i>bool</i>	Indicates if this is the default BU

8.106 pipesales_global_setup

Stores global configuration settings for the Pipesales environment.

Table 204. pipesales_global_setup table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
pipesales_bu_name	<i>string, null</i>	Name of the pipe sales business unit
pipesales_country_id	<i>int</i>	Numeric ID for the country

Field	Data Type	Description
pipesales_alias	<i>string</i>	Alias name used in pipe sales
pipesales_owner	<i>string</i>	Owner or account manager for the BU
pipesales_label	<i>int</i>	Label identifier or code
pipesales_product_type	<i>string</i>	Type of product sold (e.g., tubing, casing)
pipesales_standard	<i>string</i>	Standard specification (e.g., API, ISO)
pipesales_name	<i>string</i>	Full name of the pipe sales entity
name	<i>string</i>	Alternate name or display label
pipesales_bu_id	<i>null, int</i>	ID of the associated business unit
tubestream_id	<i>string</i>	Identifier linking to Tubestream system
pipesales_product_type_id	<i>int</i>	Numeric ID for the product type
pipesales_license_number	<i>string, null</i>	License number for sales (if applicable)
pipesales_type	<i>string</i>	Type or category of pipe sales setup
pipesales_od	<i>string</i>	Outer diameter in raw string format (e.g., "4.5")
pipesales_country_name	<i>string</i>	Country name
pipesales_od_inch	<i>string</i>	Outer diameter in inches
type	<i>string</i>	General type (can refer to business type or pipe type)
pipesales_min_yl_imperial	<i>int</i>	Minimum yield limit in imperial units (e.g., PSI)

Field	Data Type	Description
pipesales_od_mm	<i>double</i>	Outer diameter in millimeters
pipesales_id	<i>int</i>	Pipe sales ID (primary key)
pipesales_min_yl_metric	<i>int</i>	Minimum yield limit in metric units (e.g., MPa)

8.107 pipesales_term_condition

Contains predefined terms and conditions for Pipesales contracts or orders.

Table 205. pipesales_term_condition table

Field	Data Type	Description
updatewhen	<i>int</i>	Timestamp of the last update
name	<i>string</i>	Name or title of the term/condition
_id	<i>string</i>	Unique identifier
active	<i>bool</i>	Whether the term is currently active
is_default	<i>bool</i>	Marks if this is the default condition
updatewho	<i>string</i>	User who last updated the record
attachment	<i>object</i>	Attached file or document
updateid	<i>string</i>	Update transaction ID

8.108 pl_reject_reason

Stores reasons for rejection in packing list or logistics processes.

Table 206. pl_reject_reason table

Field	Data Type	Description
name	<i>string</i>	Name of the rejection reason
_id	<i>string</i>	Unique identifier

8.109 po_mts

Store Purchase order to MTS (Machine Shop).

Table 207. po_mts table

Field	Data Type	Description
no	string	PO number
closefiles	Array	Files related to PO closure
status	string	Status of the purchase order
cancelfiles	Array	Files related to cancellation
notes	string	Notes or comments
closeReason	string	Reason for closing the PO
_id	string	Unique identifier
no_gen	int, double	Auto-generated sequence number
items	Array	Items in the purchase order
delivery_terms	string	Delivery terms (e.g., FOB, CIF)
vendor	string	Vendor name or ID
location	string	Delivery location
order_date	double, int	Order date (timestamp)
cancelReason	string	Reason for cancellation

8.110 port

Master data for ports used in logistics or shipment planning.

Table 208. port table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier

Field	Data Type	Description
type	<i>string, int</i>	Type of privilege or code

8.111 privcode

Stores privilege codes used in role-based access control.

Table 209. privcode table

Field	Data Type	Description
type	<i>string, int</i>	Type of privilege or code
code	<i>string</i>	Code identifier
name	<i>string</i>	Name of the privilege
_id	<i>string</i>	Unique identifier
group	<i>string</i>	Grouping label or category

8.112 privilege

Maps roles to system privileges or access levels.

Table 210. privilege table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
role_id	<i>string</i>	Linked role ID
menu_id	<i>string</i>	Linked menu ID
access	<i>string</i>	Access level (e.g., read, write)

8.113 product_level

Defines hierarchy or categorization levels for products.

Table 211. product_level table

Field	Data Type	Description
name	<i>string</i>	Name of the level
_id	<i>string</i>	Unique identifier

8.114 protectormat

Stores protective material data used in packaging or transport.

Table 212. protectormat table

Field	Data Type	Description
name	<i>string</i>	Protector material name
_id	<i>string</i>	Unique identifier

8.115 quality

Records quality inspection data or product quality grades.

Table 213. quality table

Field	Data Type	Description
revno	<i>string</i>	Revision number
fname	<i>string</i>	File name
access	<i>string</i>	Access rights or level
docno	<i>string</i>	Document number
name	<i>string</i>	Document name
qualification	<i>string</i>	Type or level of qualification
company	<i>string</i>	Company name
folder	<i>string</i>	Folder location

Field	Data Type	Description
shareto	Array	Users or roles with access
desc	string	Description or summary
docdate	int	Document date
mill	string	Mill or production location
_id	string	Unique identifier
lastpostwho	string	Last user to update
lastpostdate	int	Timestamp of last update

8.116 quarantine

Stores inventory items that are placed under quarantine due to inspection failure or pending approval.

Table 214. quarantine table

Field	Data Type	Description
id_item	string	ID of the quarantined item
batch	string	Batch number
_id	string	Unique identifier
yard	Array	List of associated yards
owner	string	Owner of the item
when	int	Quarantine timestamp
reason	string	Reason for quarantine

8.117 range

Defines allowable or standard value ranges used in inspection or classification.

Table 215. range table

Field	Data Type	Description
name	<i>string</i>	Name of the range
_id	<i>string</i>	Unique identifier
length	<i>double</i>	Minimum length
length_max	<i>double</i>	Maximum length

8.118 recap_listing_ledger

Store Items that ready to be listed as Pipesales Listing.

Table 216. recap_listing_ledger table

Field	Data Type	Description
yard	<i>string</i>	Yard name
map_reseller_agreement	<i>bool</i>	Indicates if mapped to reseller agreement
openlisting_qty	<i>int</i>	Quantity in open listing
wt	<i>double</i>	Weight
condition	<i>string</i>	Item condition
item_desc	<i>string</i>	Item description
connection	<i>string</i>	Connection type
openlistingqty_mt	<i>int</i>	Open listing in metric tons
map_pipesales_grade	<i>bool</i>	Mapped to pipe sales grade
openlistingqty_ft	<i>int</i>	Open listing in feet
openlistingqty_m	<i>int</i>	Open listing in meters

Field	Data Type	Description
id_item	<i>string</i>	Item ID
ledger_mt	<i>double, int</i>	Ledger value in metric tons
m	<i>int, double</i>	Meters
open_listing	Array	Open listing records
map_pipesales_od_mm	<i>bool</i>	OD mm mapping flag
yardname	<i>string</i>	Name of the yard
od	<i>string</i>	Outer diameter
item_type	<i>string</i>	Type of item
item_no	<i>string</i>	Item number
country	<i>string</i>	Country of origin
ledger_m	<i>double, int</i>	Ledger value in meters
batch	<i>string</i>	Batch identifier
ledger_ft	<i>double, int</i>	Ledger value in feet
mill_id	<i>string, null</i>	Mill ID (nullable)
_id	<i>ObjectId</i>	Unique identifier
mill_name	<i>string, null</i>	Name of the mill
grade	<i>string</i>	Grade of the item
qty	<i>int</i>	Quantity
map_pipesales_od	<i>bool</i>	Mapped to pipe sales OD
map_pipesales_connection	<i>bool</i>	Mapped to pipe sales connection

Field	Data Type	Description
map_to_pipesales	<i>bool</i>	General mapping flag to pipe sales system
owner	<i>string</i>	Owner of the item
id	<i>string</i>	Custom identifier
ft	<i>int, double</i>	Feet
mt	<i>int, double</i>	Metric tons
owned	<i>string</i>	Ownership information
ledger_qty	<i>int</i>	Total ledger quantity

8.119 recap_metadata

Metadata describing recap listings such as timestamps, filters, or owners.

Table 217. recap_metadata table

Field	Data Type	Description
renew_period	<i>int</i>	Metadata renewal period in days/months
_id	<i>string</i>	Unique identifier
timestamp	<i>int</i>	Timestamp of the record

8.120 region

Stores geographic region data used in classification or analysis.

Table 218. region table

Field	Data Type	Description
code	<i>string</i>	Region code
country	<i>string</i>	Associated country
name	<i>string</i>	Region name

Field	Data Type	Description
<code>_id</code>	<code>string</code>	Unique identifier

8.121 revision_document

Tracks document revisions and version history.

Table 219. revision_document table

Field	Data Type	Description
transaction_id	<code>string</code>	ID of the original transaction or document
rev	<code>string</code>	Revision identifier/version
snapshot	<code>object</code>	Snapshot of the document before revision
<code>_id</code>	<code>string</code>	Unique identifier
no	<code>string</code>	Document number
doc_type	<code>string</code>	Type of document
revision_date	<code>int</code>	Revision timestamp

8.122 role

Stores user roles for system access and functionality management.

Table 220. role table

Field	Data Type	Description
def_url	<code>string</code>	Default URL or route for the role
is_visible	<code>bool</code>	Visibility of the role
column_order	<code>int, double</code>	Order of role in listings or views
name	<code>string</code>	Name of the role
<code>_id</code>	<code>string</code>	Unique identifier

8.123 salesdata

Contains transaction or revenue data used for sales analysis.

Table 221. salesdata table

Field	Data Type	Description
client_name	<i>string</i>	Client name
client	<i>string</i>	Client ID
mill	<i>null, string</i>	Mill name or code
year	<i>int</i>	Year
financial_year	<i>int</i>	Financial year
location	<i>null, string</i>	Sales location
itemdesc	<i>string</i>	Item description
customer	<i>null, string</i>	Customer name
bu_incoming	<i>string</i>	Incoming business unit
incoming_price	<i>string</i>	Price of incoming item
allocation	<i>null, string</i>	Allocation information
wellname	<i>null, string</i>	Well name
id_log	<i>string</i>	Log ID
source	<i>null, string</i>	Source of data
invno	<i>string</i>	Invoice number
price	<i>double</i>	Unit price
id_item	<i>string</i>	Item ID
bu	<i>null, string</i>	Business unit
curr	<i>string</i>	Currency

Field	Data Type	Description
type	<i>string</i>	Type of transaction or sale
axaptano	<i>string</i>	Axapta item number
transaction_date	<i>int</i>	Date of transaction
month	<i>int</i>	Month of transaction
sales_price	<i>double</i>	Sales price
jan_repair	<i>bool</i>	January repair flag
batch	<i>null, string</i>	Batch number
whenupdate	<i>int</i>	Last update timestamp
_id	<i>string</i>	Unique identifier
files	<i>string</i>	Attached file names
isimport	<i>bool</i>	Import flag
rig	<i>null, string</i>	Rig name or ID
whoupdate	<i>string</i>	Who updated the record
notes	<i>string</i>	Notes or remarks
qty	<i>double, int</i>	Quantity sold
total_price	<i>double, int</i>	Total price
unit_price	<i>int, double</i>	Price per unit
invgroup	<i>string</i>	Invoice group/category
mt	<i>int, double</i>	Metric tons
usage_type	<i>string</i>	Type of usage

8.124 scope_work

Defines the scope of work entries for projects or inspections.

Table 222. scope_work table

Field	Data Type	Description
doc_type	<i>string</i>	Document type
order	<i>int</i>	Sequence or priority
version	<i>double</i>	Version number
works	<i>Array</i>	List of work tasks or steps
subscope	<i>string</i>	Subscope or category
subscope_key	<i>string</i>	Unique key for subscope
_id	<i>string</i>	Unique identifier

8.125 scope_work_template

Stores templates for common scope of work setups.

Table 223. scope_work_template table

Field	Data Type	Description
createid	<i>string</i>	User ID who created the template
type	<i>string</i>	Type of template
updatewhen	<i>int</i>	Last update timestamp
createwhen	<i>int</i>	Creation timestamp
works	<i>Array</i>	Tasks or work details
updateid	<i>string</i>	ID of the last update
createwho	<i>string</i>	Creator's name or ID
default_access	<i>null, object, Array</i>	Access control configuration

Field	Data Type	Description
updatewho	<i>string</i>	Who last updated the template
level	<i>null</i>	Scope level or depth
name	<i>string</i>	Template name
_id	<i>string</i>	Unique identifier
default	<i>bool</i>	Marks if this is the default template

8.126 signoff

Records digital signoffs and approvals for various processes.

Table 224. signoff table

Field	Data Type	Description
no	<i>string</i>	Sign-off number or identifier
setup_date	<i>int, double</i>	Setup date
client	<i>string</i>	Client ID
year	<i>int, double</i>	Year
signoff_tubing	<i>null, Array</i>	Tubing sign-off data
ao	<i>int, double</i>	AO metric
unit	<i>string</i>	Measurement unit
customer	<i>string</i>	Customer ID
eom	<i>int, double</i>	End-of-month metric
signoff_files	<i>null, Array</i>	Attached sign-off documents
prev_signoff_date	<i>int</i>	Previous sign-off date
is_legacy	<i>bool</i>	Legacy record flag

Field	Data Type	Description
signoff_date	<i>int, double</i>	Sign-off date
id_item	<i>string</i>	Item ID
bu	<i>string</i>	Business unit
op	<i>int, double</i>	OP metric
transaction_date	<i>int, double</i>	Transaction date
month	<i>int, double</i>	Transaction month
batch_price	<i>int, double</i>	Price per batch
_id	<i>string</i>	Unique identifier
whoupdate	<i>string</i>	Who updated the record
whenupdate	<i>int, double</i>	Last update timestamp
dummy	<i>bool</i>	Dummy flag
signoff_casing	<i>null, Array</i>	Casing sign-off data
qty	<i>int, double</i>	Quantity
notes	<i>string</i>	Notes or remarks
total_price	<i>int, double</i>	Total price
is_santos_new	<i>bool</i>	Flag for new Santos client type
signoff_price	<i>int, double</i>	Price at sign-off
usage_type	<i>string</i>	Usage type (operational, rental, etc.)
rep	<i>bool</i>	Repair flag

8.127 specialcond

Stores special conditions applicable to contracts or transactions.

Table 225. specialcond table

Field	Data Type	Description
order	<i>double, int</i>	Order or priority
protector	<i>string</i>	Protector type or code
name	<i>string</i>	Condition name
_id	<i>string</i>	Unique identifier
applyto	<i>Array</i>	List of applicable item types
prefix	<i>string, null</i>	Prefix used in code generation
freetext	<i>string</i>	Free text field or description

8.128 spr

Store Supplier Performance Report.

Table 226. spr table

Field	Data Type	Description
md_appv	<i>string</i>	Approval flag by managing director
finaldraft	<i>object</i>	Final draft details
ehs_type	<i>Array</i>	Types of Environmental, Health, and Safety categories
folder	<i>string</i>	Folder or reference ID
reason	<i>Array</i>	Reasons associated with SPR
creator	<i>string</i>	Creator's username or ID
pdfhistinitial	<i>Array</i>	Initial PDF history
cust_review	<i>string</i>	Customer review status

Field	Data Type	Description
createdate	<i>int</i>	Creation timestamp
srcpt_invite	Array	Source recipient invitations
reply	Array	Replies to the SPR
flaginsurpic	<i>string</i>	Flag for insurance PIC
claim	<i>string</i>	Claim status or number
md_whoappv	<i>string</i>	Managing director who approved
_id	<i>string</i>	Unique identifier
subcon	<i>string</i>	Subcontractor ID
lastpostwho	<i>string</i>	Last user who posted
createwho	<i>string</i>	User who created the record
finalrevcount	<i>int</i>	Final revision count
pdfhistfinal	Array	Final PDF history
flagmdappv_who	<i>string</i>	Flag of MD approver
bu	<i>string</i>	Business unit
cpo	<i>string</i>	CPO number or reference
flagpdffinal	<i>bool</i>	Flag indicating final PDF presence
supplier_lbl	<i>string</i>	Supplier label or display name
action	<i>object</i>	Action or response object
md_user	<i>string</i>	Managing director user ID
prcpt_invite	Array	Process recipient invite list
initial	<i>object</i>	Initial review object

Field	Data Type	Description
flagpdfinitial	<i>bool</i>	Flag for initial PDF upload
subcon_lbl	<i>string</i>	Subcontractor label
scrpt_invite	<i>Array</i>	Script recipient invitations
customer	<i>string</i>	Customer ID
lastpostdate	<i>int</i>	Timestamp of last post
flagmdappv	<i>string</i>	MD approval status
year	<i>int</i>	Year of creation/transaction
final	<i>object</i>	Final review object
md_whenappv	<i>int</i>	When MD approved (timestamp)
no	<i>string</i>	SPR number/reference
isehs	<i>string</i>	Is EHS related (flag)
finalrev	<i>Array</i>	Final review entries
supplier	<i>string</i>	Supplier ID
desc	<i>string</i>	Description
customer_lbl	<i>string</i>	Customer label or display name
month	<i>int</i>	Month of creation/transaction
lastfinaldate	<i>int</i>	Timestamp of last final review
cust_reff	<i>string</i>	Customer reference
title	<i>string</i>	SPR title
customer_user	<i>string</i>	Customer user ID
lastfinalwho	<i>string</i>	User who finalized last

Field	Data Type	Description
prcpt	<i>string, Array</i>	Process recipients
srcpt	<i>Array</i>	Source recipients
pdfinitial	<i>string</i>	Path or ID of initial PDF
md_notes	<i>string</i>	MD notes or comments
closedate	<i>int</i>	Closure timestamp
status	<i>double, string, int</i>	Status (numeric or labeled)
md_review	<i>string</i>	Managing director review comment/status
pdffinal	<i>string</i>	Path or ID of final PDF
cust_appv	<i>string</i>	Customer approval status
flaginitial	<i>string</i>	Initial flag
flagmdappv_when	<i>int, double</i>	When MD approved (timestamp or date)
isfinal	<i>string</i>	Is final status flag
isdraft	<i>string</i>	Is draft flag
causal_factor	<i>string, null</i>	Causal factor description
insurpic	<i>string</i>	Insurance PIC (person in charge)

8.129 spr_causal_factor

Logs root causes or contributing factors for reported safety issues.

Table 227. spr_causal_factor table

Field	Data Type	Description
categories	<i>Array</i>	Categories of causal factor

Field	Data Type	Description
name	string	Name of the causal factor
_id	string	Unique ID
level_type	string	Type level (hierarchy)

8.130 spr_causal_factor_old

Historical or legacy version of causal factor records.

Table 228. spr_causal_factor_old table

Field	Data Type	Description
categories	Array	Categories of causal factor
name	string	Name of the causal factor
_id	string	Unique ID
level_type	string	Type level (hierarchy)

8.131 spr_hse_option

Stores selectable options for HSE-related fields or categories.

Table 229. spr_hse_option table

Field	Data Type	Description
type	string	HSE type/category
name	string	Name of the option
_id	string	Unique identifier

8.132 spr_notify

Configuration for notification triggers related to safety performance.

Table 230. spr_hse_option table

Field	Data Type	Description
email	string	Notification email address
_id	string	Unique identifier

8.133 spr_reason

Stores reasons behind safety events or non-conformities.

Table 231. spr_reason table

Field	Data Type	Description
email	string	Reason name
_id	string	Unique identifier

8.134 spr_reason_old

Historical or deprecated records of reasons in earlier SPR versions.

Table 232. spr_reason_old table

Field	Data Type	Description
email	string	Reason name
_id	string	Unique identifier

8.135 spr_user_role

Maps users to their roles in SPR processes.

Table 233. spr_user_role table

Field	Data Type	Description
_id	string	Unique ID
role	string	User role

Field	Data Type	Description
user_id	string	Linked user ID

8.136 sprv2

Supplier Performance Report, store NCR data

Table 234. sprv2 table

Field	Data Type	Description
requirement	<i>string</i>	Requirement details
completed	<i>object, null</i>	Completed data or status
mill	<i>string</i>	Mill ID or name
hse	<i>object</i>	Health, Safety, Environment info
items	<i>Array</i>	List of items/issues
level	<i>string</i>	Severity or importance level
vendor_id	<i>string</i>	Vendor ID
rejected	<i>null</i>	Rejection reason or object (if any)
classification	<i>string</i>	Type or classification
initial_action_comment	<i>object</i>	Initial action comments
final_action_comment	<i>object</i>	Final action comments
attachments	<i>Array</i>	Attached files
acknowledgers	<i>Array</i>	Users who acknowledged
status	<i>int</i>	Status code
finalReview	<i>bool</i>	Final review flag
customer_reference_no	<i>string</i>	Customer reference number

Field	Data Type	Description
progress	<i>int</i>	Completion progress
accepted	<i>object, null</i>	Acceptance data
category_id	<i>string</i>	Category ID
customer_id	<i>string</i>	Customer ID
assign_role_task	<i>Array</i>	Assigned roles or tasks
potential_lost_cost	<i>object</i>	Potential cost loss (estimation)
createname	<i>string</i>	Name of creator
potential_lost_costs	<i>Array</i>	List of estimated losses
ncr_manager_name	<i>string</i>	NCR manager's name
title	<i>string</i>	Title of the case/report
hse_consequence	<i>Array</i>	HSE impact details
createwho	<i>string</i>	Creator user ID
date_opened_required	<i>object</i>	Required opening date
updatewho	<i>string</i>	Last updater ID
vendor_name	<i>string</i>	Vendor name
_id	<i>string</i>	Unique ID
createdate	<i>int</i>	Creation timestamp
sub_category	<i>Array</i>	Sub-categories
isRowOpen	<i>bool</i>	UI control flag
customer_name	<i>string</i>	Customer name
department_id	<i>string</i>	Department ID

Field	Data Type	Description
problem_description	<i>string</i>	Problem statement
category_name	<i>string</i>	Category name
updatedate	<i>int</i>	Last updated timestamp
updatename	<i>string</i>	Name of last updater
department_name	<i>string</i>	Department name
category	<i>string</i>	Category ID or label
ncr_manager	<i>string</i>	NCR manager user ID
version	<i>string</i>	Version string
no	<i>string</i>	Reference number

8.137 sr_material

Sucker Rod material master

Table 235. sr_material table

Field	Data Type	Description
name	<i>string</i>	Sucker Rod
_id	<i>string</i>	Unique identifier
guide	<i>bool</i>	Is this a guide material?

8.138 sr_od

Sucker Rod OD.

Table 236. sr_od table

Field	Data Type	Description
name	<i>string</i>	Sucker Rod
_id	<i>string</i>	Unique identifier

Field	Data Type	Description
tubing_guide	<i>bool</i>	Is tubing guide?

8.139 sr_type

Sucker Rod Type.

Table 237. sr_type table

Field	Data Type	Description
guide	<i>bool</i>	Sucker Rod
_id	<i>string</i>	Unique identifier
name	<i>string</i>	Type short name
full_name	<i>string</i>	Type full name

8.140 storage_files

Stores uploaded files and digital assets linked to system entries.

Table 238. storage_files table

Field	Data Type	Description
exclude_ext	<i>null</i>	Excluded extensions (if any)
status	<i>string</i>	File status
filename	<i>string</i>	Name of the file
disk	<i>string</i>	Disk location or label
_id	<i>string</i>	Unique identifier
path	<i>string</i>	File storage path
username	<i>string</i>	Username of uploader
when	<i>int</i>	Upload timestamp
is_directory	<i>bool</i>	Is this entry a directory?

Field	Data Type	Description
who	<i>string</i>	User who uploaded

8.141 tally_history

Stores historical records of tally entries.

Table 239. tally_history table

Field	Data Type	Description
status	<i>string</i>	Status of the tally
createdate	<i>int</i>	Creation timestamp
id_item	<i>string</i>	Item ID
id_itemtally	<i>string</i>	Tally item ID
transaction_no	<i>string</i>	Transaction number
yard	<i>string</i>	Storage yard
owned	<i>string</i>	Ownership status
id_transaction	<i>string</i>	Transaction ID
condition	<i>string</i>	Item condition
appvwhen	<i>int</i>	Approval timestamp
type	<i>string</i>	Tally type
appvwho	<i>string</i>	Approver user ID
batch	<i>string</i>	Batch number
_id	<i>string</i>	Unique identifier
id_subtransaction	<i>null</i>	Sub-transaction ID

8.142 tenderdoc

Stores documents and related metadata submitted or generated as part of the tender or bidding process.

Table 240. tenderdoc table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier for the document
name	<i>string</i>	Name of the tender document
desc	<i>string</i>	Description of the document
modified	<i>int</i>	Last modified timestamp
access	<i>string</i>	Access level (e.g., All, Restricted)
filename	<i>string</i>	Name of the uploaded file
to	<i>array</i>	List of recipients or business units the document is shared with
tags	<i>array</i>	List of tags or categories associated with the document
owner	<i>string</i>	Owner of the document
posted	<i>int</i>	Posting timestamp
by	<i>string</i>	Name of the person who posted the document
s3key	<i>string</i>	Storage path or key on S3 or cloud storage

8.143 todo

Tracks user or system-assigned tasks and their statuses.

Table 241. todo table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier

8.144 tpicomp

Stores third-party inspection company information.

Table 242. tpicomp table

Field	Data Type	Description
lastpostdate	<i>long</i>	Last post timestamp
desc	<i>string</i>	Description
_id	<i>ObjectId</i>	Unique identifier
name	<i>string</i>	Name
lastpostwho	<i>string</i>	Who last posted

8.145 transaction_logs

Logs system transactions for auditing or analytics.

Table 243. transaction_logs table

Field	Data Type	Description
what	<i>string</i>	Description of the action
error	<i>bool</i>	Indicates if there was an error
tablet_info	<i>object</i>	Device/tablet metadata
subIdTrans	<i>string</i>	Sub-transaction ID
when	<i>int</i>	Timestamp
idtrans	<i>null, string</i>	Transaction ID
desc	<i>string</i>	Description or notes
created_at	<i>Date</i>	Record creation date
updated_at	<i>Date</i>	Last update date
snapshot	<i>null, object</i>	Snapshot of data before change

Field	Data Type	Description
type	<i>string</i>	Type of transaction
_id	<i>string</i>	Unique identifier
is_erp_log	<i>bool</i>	Whether linked to ERP
who	<i>null, string</i>	Who performed the action

8.146 transfer_activity

Records asset or item transfers between locations or entities.

Table 244. transfer_activity table

Field	Data Type	Description
createdate	<i>int, null</i>	Creation date
status	<i>string</i>	Status of the transfer
closedate	<i>int</i>	Date the transfer was closed
_id	<i>string</i>	Unique identifier
no	<i>string</i>	Transfer activity number
items_raw	<i>Array</i>	Raw item details

8.147 transfer_counter

Tracks transfer transactions' sequential identifiers.

Table 245. transfer_counter table

Field	Data Type	Description
year	<i>long</i>	Year of the record
_id	<i>ObjectId</i>	Unique identifier
seq	<i>long</i>	Sequence number

8.148 transfer_receipt

Stores receipt confirmations of transferred items.

Table 246. transfer_receipt table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
last_receipt	<i>int, null</i>	Last receipt number
journal_no	<i>Array</i>	Journal numbers
first_receipt	<i>int, null</i>	First receipt number

8.149 turnrate_hist

Historical turn rate data used in inventory analysis.

Table 247. turnrate_hist table

Field	Data Type	Description
item_desc	<i>string, null</i>	Item description
opening_balance	<i>int, double</i>	Opening balance
id_item	<i>string</i>	Item ID
price	<i>double</i>	Unit price
financial_year	<i>int</i>	Financial year
year	<i>int</i>	Year
transaction_date	<i>int</i>	Transaction date
month	<i>int</i>	Month
sales	<i>double</i>	Sales volume
item_code	<i>string, null</i>	Item code
turnrate_category	<i>string, null</i>	Turnrate category

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
expected_close	<i>double</i>	Expected closing balance
actual_close	<i>double</i>	Actual closing balance
customer	<i>string</i>	Customer

8.150 turnrate_price

Stores turn rate price calculations or estimates.

Table 248. turnrate_price table

Field	Data Type	Description
customer	<i>string</i>	Customer name or ID
month	<i>int</i>	Month
unit_price	<i>string</i>	Unit price
year	<i>int</i>	Year
transaction_date	<i>int</i>	Transaction date
_id	<i>string</i>	Unique identifier
whenupdate	<i>int</i>	Timestamp of update
id_item	<i>string</i>	Item ID
price	<i>int, double</i>	Price

8.151 user_activity

Logs user activity across the system for audit or analytics purposes.

Table 249. user_activity table

Field	Data Type	Description
email	<i>string</i>	Email address of the user
time	<i>int</i>	Activity timestamp
who	<i>string</i>	Username or user ID
ip	<i>string</i>	IP address
postingdate	<i>Date</i>	Date the activity was posted
url	<i>string, object</i>	Accessed URL or endpoint
_id	<i>string</i>	Unique identifier

8.152 user_config

Stores user-specific configuration or preference settings.

Table 250. user_config table

Field	Data Type	Description
widget_vendor_order	<i>Array</i>	Vendor order widget preferences
vendor_order_list_config	<i>object, Array</i>	Configuration for vendor order listing
_id	<i>string</i>	Unique identifier
email	<i>string</i>	User's email address

8.153 user_log

Records user login attempts, session history, or changes made.

Table 251. user_log table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
when	<i>int</i>	Log timestamp
uri	<i>string</i>	Accessed URI path
id_user	<i>string</i>	User ID

8.154 userlog

Additional or legacy user log data capturing system interactions.

Table 252. userlog table

Field	Data Type	Description
file	<i>string</i>	File name or attachment
what	<i>string</i>	Action or description
fname	<i>string</i>	Full name of the user
notes	<i>string, bool</i>	Notes or remarks
ip	<i>string</i>	IP address
menu	<i>string</i>	Accessed menu name
folder	<i>string, long, null</i>	Folder name or ID
when	<i>int, long</i>	Timestamp
idyard	<i>string</i>	Yard ID
mo	<i>int</i>	Month
year	<i>int</i>	Year

Field	Data Type	Description
imei	<i>string</i>	Device IMEI
client	<i>string, null</i>	Client reference
_id	<i>ObjectID, string</i>	Unique identifier
tb	<i>string, null</i>	Unknown field (likely table/flag)
who	<i>string, null</i>	Username or user ID
path	<i>string</i>	Accessed file or route path

8.155 users

Stores user account information including credentials, roles, and contact details.

Table 253. users table

Field	Data Type	Description
is_approved_before	<i>string, null</i>	Approval status flag
tablet_token	<i>string</i>	Tablet session token
base	<i>Array</i>	Base locations or roles
delete_date	<i>null</i>	Deletion date
portal_guide	<i>Array</i>	Portal walkthrough flags
tokenexp	<i>int</i>	Token expiry timestamp
invitation_history	<i>Array, null</i>	Invitation history records
approval_date	<i>int</i>	Date when user was approved
invite_date	<i>int</i>	Invitation date
yard_list	<i>Array</i>	List of yards assigned
customer	<i>string, null</i>	Customer associated

Field	Data Type	Description
customer_widget	Array	Widget configuration for customer
email	string	Email address
invite_by	string	User who sent the invitation
salt	string, null	Password salt
lastemail	int	Last email sent timestamp
add_by	string, null	Who added the user
whenlocked	null	Lock timestamp
privilege	string	User privilege role
id_yard	string, null	Assigned yard ID
colorbox	string	UI color theme
done_survey	int, double	Survey completion flag
approval	string, null	Approval status
name	string	Full name
contact_no	string, null	Contact number
mute_notification	Array, null	Disabled notifications
code	string, null	Employee or system code
sales_location	string	Sales office location
type	string	User type (admin, customer, etc.)
config_ui	Array	UI configuration settings
country_id	string	Country ID
mute_notification_email	Array, null	Disabled email notifications

Field	Data Type	Description
session_id	<i>string, null</i>	Session token or ID
contract	Array	Linked contracts
vendor	Array	Linked vendor data
active	<i>string</i>	Activity status
disable_kpi	<i>bool</i>	KPI disabled flag
_id	<i>string</i>	Unique identifier
add_date	<i>int, null</i>	Date added to system
sname	<i>string</i>	Short name or nickname
failedcounter	<i>int</i>	Login failure count
fname	<i>string</i>	First name
pref	<i>object</i>	Preference object
approval_reason	<i>string</i>	Reason for approval or rejection
role	Array	Role IDs or definitions
sso_token	<i>string, null</i>	SSO token
ncr_widget	Array	NCR dashboard widgets
employee_code	<i>string</i>	Internal employee code
token	<i>string</i>	Auth token
widget	<i>null, Array</i>	Dashboard widget configuration

8.156 vendor_order

Stores order data related to external vendors or suppliers.

Table 254. vendor_order table

Field	Data Type	Description
mill_tag	<i>string</i>	Mill tag reference
mdr	<i>Array</i>	MDR documents or references
product_type	<i>string</i>	Type of product ordered
mill	<i>string, null</i>	Mill name or ID
location	<i>null</i>	Location (nullable)
yard	<i>null</i>	Yard location (nullable)
item_desc	<i>string</i>	Item description
pono	<i>string</i>	Purchase order number
shipments	<i>null</i>	Shipment details (nullable)
date_ordered	<i>int</i>	Order date (timestamp)
progress_cargo_ready_date	<i>int</i>	Progress cargo ready date
source	<i>string</i>	Source of the order
status	<i>string</i>	Status of the vendor order
destination	<i>string</i>	Destination of the goods
destination_port	<i>null</i>	Destination port (nullable)
id_item	<i>string</i>	Item ID
act_cargo_ready_date	<i>null</i>	Actual cargo ready date
delivery_details	<i>Array</i>	Delivery detail entries
item	<i>object</i>	Detailed item info

Field	Data Type	Description
item_type	<i>string</i>	Type/category of item
_id	<i>string</i>	Unique identifier
createdate	<i>int</i>	Record creation timestamp
delivery_term	<i>string</i>	Delivery term (e.g., FOB, CIF)
manufacturing_schedule	<i>Array</i>	List of manufacturing schedules
source_port	<i>null</i>	Source port
qty	<i>int</i>	Quantity ordered
updatedate	<i>int</i>	Last update timestamp
ft	<i>double</i>	Quantity in feet
eta	<i>int</i>	Estimated time of arrival
mt	<i>double</i>	Quantity in metric tons

8.157 vessel

Master table for vessels used in logistics and shipping.

Table 255. vessel table

Field	Data Type	Description
updated_at	<i>Date</i>	Last update timestamp
synced_at	<i>Date</i>	Last sync time
_id	<i>string</i>	Unique identifier
name	<i>string</i>	Vessel name
code	<i>string</i>	Vessel code
mmsi	<i>string, null</i>	Maritime Mobile Service Identity

8.158 vessel_tracking

Stores tracking information for vessel movements and statuses.

Table 256. vessel_tracking table

Field	Data Type	Description
position	<i>object</i>	Vessel's current position
vessel_id	<i>int</i>	ID of the vessel
cn_name	<i>string</i>	Vessel Chinese name
status	<i>string</i>	Vessel status
position_received	<i>string</i>	Position data received timestamp
cn_iso2	<i>string</i>	ISO2 country code
voyage	<i>object</i>	Voyage information
static_received	<i>string</i>	Static data received timestamp
mmsi_number	<i>int</i>	MMSI number
name	<i>string, null</i>	Vessel name
openorder_lotid	<i>string</i>	Open order lot ID
next_port_date	<i>string, null</i>	Next port ETA
next_port	<i>null, object</i>	Next port information
vt_verbose	<i>string</i>	Verbose tracking info
mmsi_str	<i>string</i>	MMSI number as string
callsign	<i>string, null</i>	Vessel call sign
imo_number	<i>null, int</i>	IMO number
updatewhen	<i>int</i>	Last update timestamp
_id	<i>string</i>	Unique identifier

Field	Data Type	Description
last_port	<i>null, object</i>	Last visited port
length	<i>null, int</i>	Vessel length
schedules	Array	Travel schedule
last_update	<i>int</i>	Most recent update

8.159 vims_avgprice

Store average price per item and customer.

Table 257. vims_avgprice table

Field	Data Type	Description
month	<i>int, double</i>	Forecast month
year	<i>int, double</i>	Forecast year
avg_price	<i>int, double</i>	Average price
client	<i>string, null</i>	Client name or ID
ledger	<i>null, Array</i>	Ledger details
prev_price	<i>double</i>	Previous period's price
bu	<i>string</i>	Business unit
_id	<i>ObjectId</i>	Unique identifier
id_item	<i>string, null</i>	Item ID
lastupdate	<i>int, double</i>	Last update timestamp
eom	<i>int, double</i>	End of month quantity or marker
customer	<i>string</i>	Customer name

8.160 vims_batchprice

Store batch price value per item and customer.

Table 258. vims_batchprice table

Field	Data Type	Description
period_end	<i>int</i>	End of the pricing period
id_item	<i>string</i>	Item ID
period	<i>int</i>	Period identifier
quarter	<i>int</i>	Quarter of the year
price	<i>double</i>	Batch price
bu	<i>string</i>	Business unit
currency	<i>string</i>	Currency code
year	<i>int</i>	Year
unit	<i>string, null</i>	Measurement unit
client	<i>string</i>	Client name or ID
_id	<i>string</i>	Unique identifier
whenupdate	<i>int</i>	Last update timestamp
whoupdate	<i>string</i>	Who performed the update
customer	<i>string</i>	Customer name

8.161 vims_breakdown

Store commitment stock breakdown per customer

Table 259. vims_breakdown table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier

Field	Data Type	Description
aging	<i>object</i>	Aging breakdown object
setup_date	<i>int</i>	Date when breakdown was setup
year	<i>int</i>	Forecast or reporting year
transaction_date	<i>int</i>	Transaction timestamp
month	<i>int</i>	Forecast or reporting month
hist_time	<i>int</i>	Historical time marker
customer	<i>int, string</i>	Customer ID or code
id_item	<i>string</i>	Item ID
hist_year	<i>int</i>	Historical year
hist_month	<i>int</i>	Historical month

8.162 vims_forecast

Store forecast value entered by user.

Table 260. vims_forecast table

Field	Data Type	Description
_id	<i>ObjectId, string</i>	Unique identifier of the document
year	<i>int</i>	Year of the forecast
transaction_date	<i>int, long</i>	Timestamp of the transaction
month	<i>int</i>	Month of the forecast
repair	<i>bool</i>	Indicates if it is a repair forecast
commitment	<i>string</i>	Forecast commitment reference
setup_date	<i>int</i>	Date setup was initiated (timestamp)

Field	Data Type	Description
client	<i>string</i>	Client name or ID
type	<i>string</i>	Type of transaction or forecast
group	<i>string</i>	Group classification
hist_year	<i>int</i>	Historical year reference
hist_month	<i>int</i>	Historical month reference
rig	<i>string, null</i>	Rig involved
wellname	<i>string, null</i>	Well name
customer	<i>string</i>	Customer name or ID
end_date	<i>int</i>	Forecast end date (timestamp)
qty	<i>int</i>	Quantity forecasted
order	<i>int</i>	Order number or index
total_price	<i>int</i>	Total price estimate
who_update	<i>string</i>	User who last updated
start_date	<i>int, long</i>	Start date of the forecast (timestamp)
id_item	<i>string</i>	Item ID being forecasted
bu	<i>string</i>	Business Unit
price	<i>int</i>	Price per unit

8.163 vims_forecast_hist

Store historical value of forecast.

Table 261. vims_forecast_hist table

Field	Data Type	Description
_id	<i>ObjectId</i>	Unique identifier
id_item	<i>string, null</i>	Item ID
group	<i>string, null</i>	Group classification
new	<i>string</i>	New forecast or status indicator
month	<i>int</i>	Month
transaction_date	<i>bool, int</i>	Transaction timestamp or indicator
year	<i>int</i>	Year
import	<i>bool</i>	Import flag
customer	<i>string</i>	Customer name
hist_month	<i>int, double</i>	Historical month
hist_year	<i>int</i>	Historical year
rig	<i>string</i>	Rig involved
wellname	<i>string</i>	Well name
status	<i>string</i>	Forecast status
qty	<i>int</i>	Quantity

8.164 vims_pdf

Store signoff pdf.

Table 262. vims_pdf table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
notes	<i>string</i>	Notes associated with the PDF
anydate	<i>string</i>	Date field (possibly ISO date)

8.165 vims_recomorder

Store Recommended Order creating in account balance.

Table 263. vims_recomorder table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
batch_price	<i>double</i>	Price for the batch
month	<i>int</i>	Month of recommendation
id_item	<i>string</i>	Item ID
ro	<i>int</i>	Recommendation order number
bu	<i>string</i>	Business Unit
ro_date2	<i>bool, string, null</i>	Secondary RO date
qty	<i>int</i>	Quantity
ro_date	<i>int</i>	Recommendation order date (timestamp)
customer	<i>string</i>	Customer name
ct	<i>string</i>	Possibly contract type
ro_notes	<i>string, bool, null</i>	Notes on RO

Field	Data Type	Description
transaction_date	<i>int</i>	Transaction date (timestamp)
year	<i>int</i>	Year of recommendation

8.166 well_design

It contains engineering and design details for wells (e.g., structure, depth, equipment).

Table 264. well_design table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
well_number	<i>int</i>	Well number
item	<i>Array</i>	Items related to the well
customer	<i>string</i>	Customer
well_type	<i>string</i>	Type of well

8.167 yard

Stores yard master data, including layout or location names.

Table 265. yard table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
enable_trust_limit	<i>bool</i>	Whether trust limit is enabled
zip	<i>string, null</i>	Zip/postal code
drill_location	<i>string, null</i>	Drilling location
region	<i>string, null</i>	Region
tallyreq	<i>string</i>	Tally requirement flag
phone	<i>string, null</i>	Phone number

Field	Data Type	Description
updated_at	Date	Last update timestamp
erp_code	string	ERP system code
import	bool, null	Import flag
email	string, null	Contact email
locode	string	Location code
max_limit_usd	string, null	Maximum limit in USD
trust_limit_expiry	int, null	Trust limit expiry date
city	string, null	City name
rack	object	Rack configuration
code	string	Yard code
name	string	Yard name
bu	null, Array	Business Units
tally_req	null	Tally requirement (possibly deprecated)
whocreate	string	Creator user
type	Array	Yard types
address	string, null	Street address
insurance_usd	double	Insurance amount in USD
max_limit_mt	string, null	Maximum limit in metric tons
active	string	Active status
no_inspect	bool	No inspection flag

Field	Data Type	Description
country	<i>string</i>	Country
access	<i>null</i>	Access control settings
sub	<i>object, null</i>	Sub-location or subordinate settings
whencreate	<i>int</i>	Creation timestamp
is_managed	<i>bool</i>	Is the yard actively managed
resource_capacity_mt	<i>string</i>	Resource capacity in metric tons
landed_cost_mt	<i>double, null</i>	Landed cost per metric ton
category	<i>null</i>	Yard category
contact	<i>string, null</i>	Contact person

8.168 yard_balance_report

Reports show stock balances by yard location and item.

Table 266. yard_balance_report table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
condition	<i>string</i>	Item condition
owner	<i>string</i>	Owner of the items
year	<i>int</i>	Year
ft	<i>int, double</i>	Quantity in feet
id_item	<i>string</i>	Item ID
mt	<i>int, double</i>	Quantity in metric tons
dont_show	<i>bool</i>	Visibility flag

Field	Data Type	Description
location_name	<i>string</i>	Yard location name
usd	<i>int, double</i>	Value in USD
related_location_name	<i>string</i>	Related yard location
qty	<i>int</i>	Quantity of items
transaction_id	<i>string</i>	Transaction ID
ownership	<i>string</i>	Ownership status
type_transfer	<i>string</i>	Type of transfer
batch	<i>string</i>	Batch reference
transaction_code	<i>string</i>	Transaction code
week_of_year	<i>int</i>	Week of the year
related_location	<i>string</i>	Related yard location ID
type	<i>string</i>	Type of item or report
subtype	<i>string</i>	Subtype of item
location	<i>string</i>	Yard location
date	<i>int</i>	Date (timestamp)

8.169 **yard_opening_balance_report**

Records opening stock balances at the start of a period for each yard.

Table 267. well yard_opening_balance_report table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
qty	<i>int</i>	Quantity

Field	Data Type	Description
year	<i>int</i>	Year of report
usd	<i>int, double</i>	Value in USD
yard	<i>string</i>	Yard ID
mt	<i>int, double</i>	Quantity in metric tons
week	<i>int</i>	Week number
ownership	<i>string</i>	Ownership type/status
_id	<i>string</i>	Unique identifier
qty	<i>int</i>	Quantity
year	<i>int</i>	Year of report
usd	<i>int, double</i>	Value in USD
yard	<i>string</i>	Yard ID
mt	<i>int, double</i>	Quantity in metric tons
week	<i>int</i>	Week number
ownership	<i>string</i>	Ownership type/status

8.170 yard_supervisor

Stores information about supervisors assigned to yard locations or shifts.

Table 268. yard_supervisor table

Field	Data Type	Description
_id	<i>string</i>	Unique identifier
yard	<i>string</i>	Yard ID
supervisors	<i>Array</i>	List of supervisors assigned