**SAP Automated Sales Workflow System**

A major project report submitted in partial fulfillment of the requirement for the award of degree of

**Bachelor of Technology**

in

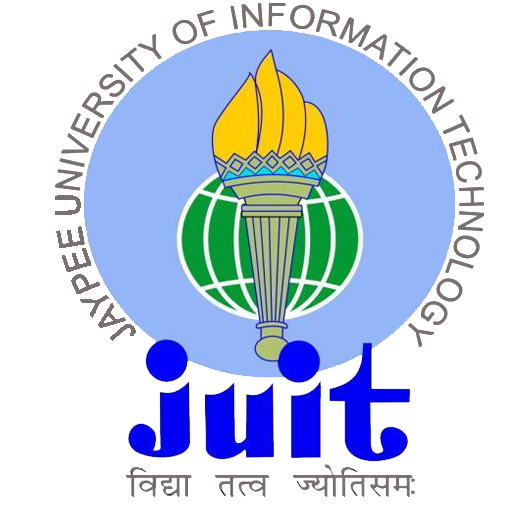
**Computer Science & Engineering**

*Submitted by*

**Tanisha Chaubey (211312)**

*Under the guidance & supervision of*

**Dr. Anita**



**Department of Computer Science & Engineering and**

**Information Technology**

**Jaypee University of Information Technology, Waknaghat, Solan - 173234 (India)**

**May 2025**

**Supervisor’s Certificate**

This is to certify that the major project report entitled **‘SAP Automated Sales Workflow System’**, submitted in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering**, in the Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Waknaghat, is a bonafide project work carried out by Tanisha Chaubey(211312) under my supervision during the period from July 2024 to May 2025.

I have personally supervised the research work and confirm that it meets the standards required for submission. The project work has been conducted in accordance with ethical guidelines, and the matter embodied in the report has not been submitted elsewhere for the award of any other degree or diploma.

Date: 09-05-2025

Place: Waknaghat

(Supervisor Signature) (External Supervisor Signature)

Supervisor Name: Dr. Anita Supervisor Name: Prakadeesh Anghappan

Designation: Assistant Professor (SG) Designation: Lead Software Engineer

Department: CSE & IT Department: Customer Success Team

**Candidate’s Declaration**

We hereby declare that the work presented in this major project report entitled **‘SAP Automated Sales Workflow System’**, submitted in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering**, in the Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Waknaghat, is an authentic record of my own work carried out during the period from July 2024 to May 2025 under the supervision of **Dr. Anita**.

I further declare that the matter embodied in this report has not been submitted for the award of any other degree or diploma at any other university or institution.

(Student Signature)

Name: Tanisha Chaubey

Roll No.: 211312

Date: 09-05-2025

This is to certify that the above statement made by the candidates is true to the best of my knowledge.

Date: 09-05-2025

Place: Waknaghat

(Supervisor Signature) (External Supervisor Signature)

Supervisor Name: Dr. Anita Supervisor Name: Prakadeesh Anghappan

Designation: Assistant Professor (SG) Designation: Lead Software Engineer

Department: CSE & IT Department: Customer Success Team

**ACKNOWLEDGEMENT**

Firstly, I express my heartiest thanks and gratefulness to almighty God for his divine blessing in making it possible to complete the project work successfully.

I am grateful and wish my profound indebtedness to Supervisor Dr. Anita, Assistant Professor (SG) Department of CSE Jaypee University of Information Technology, Waknaghat. The deep knowledge & keen interest of my supervisor in the field of “SAP” and “ABAP” helped me to carry out this project. Her endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts and correcting them at all stages have made it possible to complete this project.

I would like to express my heartiest gratitude to Dr. Anita, Department of CSE, for her kind help in finishing my project.

I would also generously welcome each one of those individuals who have helped us straightforwardly or in a roundabout way in making this project a win. In this unique situation, We might want to thank the various staff individuals, both educating and non-instructing, who have developed their convenient help and facilitated our undertaking.

Finally, I must acknowledge with due respect the constant support and patience of my  parents.

Tanisha Chaubey

(211312)

**TABLE OF CONTENTS**

**SUPERVISOR’S CERTIFICATE………………………………………………………...i**

**CANDIDATE'S DECLARATION………………………………………………………..ii**

**ACKNOWLEDGEMENT………………………………………………………………....iii**

**LIST OF TABLES…………………………………………………………………………vi**

**LIST OF FIGURES………………………………………………………………………..vii**

**LIST OF ABBREVIATIONS……………………………………………………………..ix**

**ABSTRACT………………………………………………………………………………...x**

**1. INTRODUCTION**

1.1 Introduction……………………………………………………………………...1

1.2 Problem Statement………………………………………………………………3

1.3 Objective………………………………………………………………………...4

1.4 Significance and Motivation…………………………………………………….5

1.5 Organization of project report…………………………………………………...7

**2. LITERATURE SURVEY**

2.1 Overview of relevant literature …………………………………………………8 2.2 Key gaps in the literature……………………………………………………….13

**3. SYSTEM DEVELOPMENT**

3.1 Requirements and Analysis…………………………………………………14

3.1.1 Functional Requirement…………………………………………….14

3.1.2 Non-Functional Requirement……………………………………….17

3.1.3 User Types…………………………………………………………..18

3.1.4 Custom Tables Used…………………………………………………19

3.2 Project Design and Architecture…………………………………………….20

3.3 Data Preparations……………………………………………………………24

3.4 Implementation………………………………………………………………25

3.4.1 Tools and Technologies used………………………………………..27

3.4.2 Project Code Snippets……………………………………………….29

3.5 Key Challenges……………………………………………………………..35

**4. TESTING**

4.1 Testing Strategy ……………………………………………………………37

4.2 Test Cases and Outcomes ………………………………………………….41

**5. RESULTS AND EVALUATION**

5.1 Results ………………………………………………………………………48

**6. CONCLUSIONS AND FUTURE SCOPE**

6.1 Conclusion …………………………………………………………………55

6.2 Future Scope ……………………………………………………………….56

**REFERENCES…………………………………………………………………………….**58

**APPENDIX…………………………………………………………………………………61**

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| **S. No.** | **Name of the table** | **Page No.** |
| 1 | Table 2.1 : Literature Review | 9 |
| 2 | Table 3.1.4: Custom Table Names | 19 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **S.No.** | **Name of the Figure** | **Page No.** |
| 1 | Figure 3.2.1: Flow Chart of the Project | 22 |
| 2 | Figure 3.2.2: Use Case Diagram | 22 |
| 3 | Figure 3.2.3: Sequence Diagram | 23 |
| 4 | Figure 3.4.2.1: Data Declaration For Screen 1. | 29 |
| 5 | Figure 3.4.2.2: Data Declaration For Screen 3. | 30 |
| 6 | Figure 3.4.2.3: Data Declaration For Screen 4. | 30 |
| 7 | Figure 3.4.2.4: PBO (Process Before Output) Module for Screen 1. | 31 |
| 8 | Figure 3.4.2.5: PAI (Process After Input) Module for Screen 1. | 31 |
| 9 | Figure 3.4.2.6: Form-Routines Module for Build\_FieldCatalog. | 32 |
| 10 | Figure 3.4.2.7: Form-Routines Module for Build\_Display\_Table. | 32 |
| 11 | Figure 3.4.2.8: Form-Routines Module for ALV Display. | 33 |
| 12 | Figure 3.4.2.9: Form-Routines Module for Fetching Data. | 33 |
| 13 | Figure 3.4.2.10: Form-Routines Module for Auto Populate the customer ID. | 34 |
| 14 | Figure 3.4.2.11: Form-Routines Module for User\_Admin\_Login | 34 |
| 15 | Figure 5.1.1: Module Pool Editor Screen | 49 |
| 16 | Figure 5.1.2: SmartSale Login Screen | 49 |
| 17 | Figure 5.1.3: Register Screen | 50 |
| 18 | Figure 5.1.4: Customer Screen Showing Orders | 50 |
| 19 | Figure 5.1.5: Update Profile | 51 |
| 20 | Figure 5.1.6: Admin screen with Open Orders | 51 |
| 21 | Figure 5.1.7: Admin Screen with Cancelled Orders | 52 |
| 22 | Figure 5.1.8: Admin Screen with Completed Orders | 52 |
| 23 | Figure 5.1.9 :Workflows using T-code SWDD | 53 |
| 24 | Figure 5.1.10: Smartforms Creation | 53 |
| 25 | Figure 5.1.11: Smart forms Quotation Details | 54 |
| 26 | Figure 5.1.12: Smart forms Invoice Details | 54 |

**LIST OF ABBREVIATIONS**

|  |  |
| --- | --- |
| SAP | System Application Product |
| ABAP | Advanced Business Application Programming |
| ERP | Enterprise Resource Planning |
| ECC | ERP Central Component |
| HANA | High Performance Analytic Appliance |
| ODATA | Operational Data |
| CDS | Core Data Services |
| HCM | Human Capital Management |
| BAPI | Business Application Programming Interface |
| BADI | Business Add-Ins |
| PAI | Process After Input |
| PBO | Process Before output |
| OOP | Object Oriented Programming |
| MM | Material Management |
| GUI | Graphic User Interface |
| RFC | Remote Function Call |
| ALV | ABAP List Viewer |
| CRM | Customer Relationship Management |
| SD | Sales Distribution |
| BDC | Batch Data Communication |
| LSMW | Legacy System Migration Workbench |

**ABSTRACT**

This project presents the design and development of a custom SAP application for SmartSale Ltd., an automobile manufacturing company aiming to standardize and streamline its Sales and Distribution (SD) processes. Leveraging SAP ABAP and module pool programming, the solution provides a fully functional GUI-based portal to manage customer inquiries, quotations, order tracking, and fulfillment workflows. Key components include custom login and registration interfaces, ALV-based dashboards for both customers and admins, and seamless integration with standard SAP SD transactions.

Advanced SAP technologies such as ABAP Object-Oriented Programming (OOP), screen management using PBO and PAI events, and modularization through function modules were used to ensure maintainability and scalability. Email alerts and approval processes were enabled by SAP Business Workflow, while Smartforms took care of development of expert quotes and invoices. Real-time testing scenarios were reflected without manual table maintenance by migrating data through using BDC methods.

The system has also on-the-admin side controls for the processing of price adjustments, order fulfillment, issuing invoices, checking stock, and customer actions such as order cancellation requests and making inquiries. With these features, customers and the administrative team can team in real time which helps in maintaining a total sales loop.

Apart from providing thorough training on how to integrate SAP development applications, this project also proves a reality of implementation of SAP SD functionalities. As a scalable reference, this framework allows similar businesses to automate and optimize their sales lifecycles.

**CHAPTER 1: INTRODUCTION**

* 1. **INTRODUCTION**

A wide array of organizations use SAP, an advanced enterprise resource planning (ERP), to optimally manage their basic business processes. It contains several connected modules including: Sales and Distribution (SD), Production Planning (PP), Human Capital Management (HCM), Materials Management (MM) and Finance (FI). By every module focusing on its section of the business, they are meant to work together and have a unified flow to the overall firm.

The Sales and Distribution (SD) module takes care of such important tasks such as client-interaction, orders management, coordinating deliveries and processing bills. By providing the backbone to the order-to-cash process, it ensures that its client requests come to fruition with the right amount of accuracy and time. Producers of sales activities and direct customer contact process gain a lot from possessing SD module’s capability to ensure operation order and effectiveness. Comprehensive integration with important modules such as; MM (material management) and FI (finance and accounting) improves its ability to monitor each stage of business operations.

Companies tend to find it challenging to fully integrate the standard SAP SD module with the existing workflows or user interfaces although the module presents strong capabilities. In order to enhance existing core capabilities, businesses are usually in need to design custom SAP applications. Developer-built, these customized solutions are embedded into ABAP & module pool development framework in order to support bespoke interfaces, custom business rules and a smooth user experience that enables process control and efficiency. Additionally, this approach enables automation and sophisticated reporting as well as specialized validation requirements not possible by standard SAP.

To load is to create a customized SAP module pool application that streamlines and normalizes an assortment of sales and distribution processes. Common SAP tools are used, and the system also has the ability to track orders, manage quotations, provide for customer logins, and provide for queries and invoicing. This means that administrators and customers can enjoy professional and interactive context because of SAP workflows, email notifications, Smartforms, and ALV grids. In living up to SAP’s essence, the platform completes features that are missing and facilitates smooth order flowing from request to completion.

The project shows how implementing customized solutions can help reduce manual work, increase business agility at the Sales and Distribution domain level by integrating standard SAP design with customized development.

Besides, enterprises are introducing flexible and instinctive systems in accordance with the users’ needs due to the increasing demands of automation and digital transformation. It is possible for developers to design screen-based user interfaces that ensure both security and user interactivity through SAP’s custom module pool programming. With the aid of buttons, dropdown elements, tables, and tabs, these custom screens become more easily obtained, enabling administrators and customers to perform sophisticated functions easily and with no reliance on backend transactions.

Our project aims at showing that it is possible to incorporate all sales stages into a thoroughly customized SAP platform without losing the ability for seamless linkage to SAP’s business rules and standard components. By leveraging SAP ABAP tools, workflow automation, and smart form generation, the project not only enhances operational efficiency but also provides a learning platform to explore the practical implementation of core SAP development concepts. This initiative bridges the gap between theoretical SAP knowledge and its real-world application, making it highly relevant for both academic and professional growth.

* 1. **PROBLEM STATEMENT**

SmartSale Ltd., a renowned automobile manufacturing company that specializes in the complete lifecycle of vehicle production—from part procurement and manufacturing to final assembly and sales—identified the need to streamline and standardize its Sales & Distribution (SD) operations. Currently, much of the SD workflow is either manually handled or fragmented across different modules, leading to inefficiencies, delays in customer response, and limited visibility into order statuses. These challenges often result in miscommunication, redundant tasks, and poor customer satisfaction.

To address these issues, the company proposed the development of a bespoke SAP application using custom ABAP programming. The solution would centralize critical business operations by leveraging SAP’s module pool programming to create a user-friendly interface that integrates directly with SAP’s backend. Essential master data such as customer credentials and product information will be managed using dedicated database tables. These tables will store sensitive and operational data in a structured and secure manner—for example, the Customer Table will manage IDs, names, passwords, and security credentials, while the Product/Material Table will track inventory with product IDs, names, prices, and available stock.

The application will be initiated via a custom transaction code and feature a secure login mechanism for both customers and administrators. A dedicated registration process will ensure that only verified users are onboarded, with automatic Customer ID generation and security question validation incorporated to maintain data integrity. After successful login, customers will be directed to a personalized dashboard displaying their open orders using ALV Grid presentation. The interface will be dynamic, allowing customers to interact with selected orders using dedicated GUI buttons—for creating new inquiries, requesting follow-ups, canceling orders, and accepting or rejecting quotations.

From the administrative perspective, the application will offer robust tools to manage incoming orders more effectively. Admin users will be able to view and filter orders based on their status—Open, Cancelled, or Completed—through a dropdown-based navigation system. For open orders, administrators will perform stock validations, issue quotations using a dynamic popup window for price entry, and once approved by customers, fulfill the orders and generate invoices using Smartforms. Additionally, the application will automate communications by sending email notifications at every crucial event—ensuring timely updates between the customer and the business team.

A key motivation behind this project lies in handling real-world complexities such as multi-product orders, partial stock availability, and the need for timely administrative decisions. The current lack of automation and system-driven decision support was causing avoidable delays and creating friction between departments. By creating a centralized, workflow-enabled platform with automated emails and screen-driven processing, this project aims to remove those bottlenecks and provide a scalable solution tailored to business needs.

Ultimately, this solution serves as a practical demonstration of how standard SAP capabilities can be extended with custom development to build a more intelligent, transparent, and responsive SD environment. It not only simplifies daily operations but also enhances user experience, security, and business performance.

* 1. **OBJECTIVES**

The major goal of this project is to design and implement a customized SAP application that simplifies and automates the Sales and Distribution (SD) process for enterprises by integrating module pool programming, database design, workflow automation, and SAP Smartforms. The application is designed to improve system communication through email notifications, correct deficiencies in manual order handling, and increase visibility for administrators and customers. Specific goals are as follows:

* **Automating the Sales and Distribution Workflow**: Develop a structured and intuitive module pool interface to streamline customer interactions such as inquiry creation, order cancellation, quotation handling, and follow-ups. These processes are integrated with backend validations to ensure smooth and accurate transaction flow.
* **Designing a Custom Transaction with Secure Access**: Implement a transaction code that acts as a centralized entry point into the system, offering secure login for both customers and admin users. This also includes a custom registration interface with auto-generated customer IDs and robust password and security question validation mechanisms.
* **Enhancing Real-Time Communication and Notifications**: Integrate automated email notifications that are triggered at key decision points (such as inquiry creation, quotation sent, cancellation request, etc.) to ensure timely and traceable communication between customers and the administrative team.
* **Integrating SAP Standard Components and Output Forms**: Utilize SAP’s standard Inquiry screen for creating sales documents and employ Smartforms to generate customer invoices upon successful order fulfillment, ensuring compliance with SAP’s output management standards.
* **Managing Data Through Custom Tables and ALV Grids**: Structure customer and product information using custom DDIC tables and present open order details through interactive ALV grids. This allows users to filter, select, and process data efficiently while maintaining data integrity and SAP UI standards.
  1. **SIGNIFICANCE AND MOTIVATION OF THE PROJECT WORK**

This project has been a valuable opportunity to go beyond textbook learning and understand how business challenges are solved using SAP technologies in a real-world scenario. SmartSale Ltd.'s requirement to streamline their Sales & Distribution process offered the perfect context to explore how customized SAP solutions can bring structure, automation, and clarity to business workflows.

One of the key motivations behind choosing this project was to gain practical exposure to SAP ABAP programming and understand how custom applications are developed on top of the SAP framework. While theoretical knowledge gives a foundation, working on this end-to-end implementation allowed me to connect the dots between back-end logic, database design (using DDIC), and front-end user experience through module pool programming. It also introduced me to SAP’s development lifecycle, right from requirement analysis and screen design to testing and deployment.

A major part of the learning came from simulating business transactions such as customer registration, inquiry creation, order follow-ups, and quotation processing. It therefore became easier to grasp the movement of data through the system and how every process is connected to the others. When I realized how to link a customer inquiry to an order and monitor the latter with follow-ups and delivery, my vision of SAP SD’s contribution to business processes has become more defined.

Having SAP Workflow to dispatch email notifications whenever customers exhibit relevant behaviors was a major component of the project. Especially rewarding was this project outcome for the impact it had on popularizing event-driven automation in business applications. Tradition of workflow abilities transformed the system’s look and made it more responsive and intuitive, fitting the real-life needs businesses set to their digital platforms perfectly.

I also learned a lot working with Smartforms, particularly on generating quotes and invoices, and that provided me with insight into SAP’s abilities in document management. By developing structured output forms, and by including them in email notifications, I found it easier to comprehend SAP’s processes for managing customer communication.

Eventually, this experience has increased my appreciation of the particular preaching of SAP in terms of the conduct of business activities. It also reinforced the importance of clean design, code modularity, and following naming conventions—skills that are essential in any enterprise-level development role. More than just coding, this experience was about thinking like a consultant—understanding business needs and translating them into reliable, maintainable software.

* 1. **ORGANISATION OF PROJECT REPORT**

**Chapter 1: Introduction**

This chapter sets the stage for the project by explaining the background, problem statement, and the motivation behind the work. It outlines the main goals and objectives of the project, the business scenario of SmartSale Ltd., and gives a high-level overview of what the project aims to achieve. It also includes the significance of the project and how the report is organized.

**Chapter 2: Literature Survey**

This chapter provides a review of the existing literature and tools related to SAP ABAP, the SD module, workflows, and Smartforms. It highlights current solutions and practices in the domain and identifies key gaps that this project aims to address. The review helped in understanding industry standards and selecting the most suitable approaches for implementation.

**Chapter 3: System Development**

This chapter covers the core development of the system. It starts with gathering and analyzing the functional and technical requirements, followed by the design of the application architecture. It details the module pool programming, database structures, and the workflow logic used. The data preparation using BDC and other migration techniques is also discussed here. Challenges faced during development and how they were resolved are included in this chapter.

**Chapter 4: Testing**

This chapter focuses on the testing phase of the project. It describes the testing strategies used to verify that each component of the system functions correctly. To demonstrate the system’s accuracy and dependability, the system includes various test scenarios, likely results, and actual results.

**Chapter 5: Results and Evaluation**

The results achieved using the applied system are presented in this chapter. This chapter describes the resulting outputs of the project as well as the major functionalities introduced and their success at increasing smart sale Ldt’s sales and distribution operations.

**Chapter 6: Conclusions and Future Scope**

In the final chapter, lessons learned from the project are described, supplemented by comments on the methodology of the project development. It also investigates opportunities for future improvement and describes how the system can be improved in future versions, such as those with real-time dashboards, advanced analytics, or as extension of solution to other SAP modules.

**CHAPTER 2: LITERATURE SURVEY**

**2.1 OVERVIEW OF RELEVANT LITERATURE**

As SAP ABAP is a critical component in the customization of SAP ERP systems, the area has attracted researchers from academic disciplines and those in the industrial domain. For a good reference book on the ABAP language and its details on procedural and object-oriented programming, the readers can consult books like that of Horst Keller and Sascha Krüger – ABAP Objects. Basic advice in such resources allows people to get to know how ABAP is used in improving and personalizing SAP applications for enterprise setting.

Numerous research papers examined how ABAP assists in attaining the alignment between technical implementations and processes of businesses in information systems and software engineering. Detailed information regarding how ABAP interacts with NetWeaver platform can be found within SAP NetWeaver AS ABAP System Administration publication and numerous SAP Press books which emphasize the importance of ABAP in maintaining modular design, providing effective maintenance of the system, and increasing the efficiency of the system. The books frequently discuss good debugging, tuning performance, as well as managing SAP code.

Research has investigated issues and approaches tied to the ABAP programming context. There are numerous articles that discuss topics such as the transition from the classical to modern ABAP-syntaxis, implementing of the S/4HANA, and the management of technical debt in legacy ABAP-systems. Scholarly works highlight the need to update the ABAP development through new discourses such as the ABAP RESTful Application Programming Model (RAP) and Core Data Services (CDS).

There are several scholarly articles that are dedicated to describing the ways in which ABAP can be unified with other technological systems. The documentation on this side of ABAP’s interfaces to APIs, OData services, SAP Fiori has grown with the rising interest of the community towards Hybrid system landscapes. Official publications from SAP guide developers to create sustainable, platform independent applications which not only improve user-friendliness but also data access by combining ABAP with newer technologies.

ABAP literature is supplemented by training and certification material that places developers through project scenarios that inevitably resemble its use in practice. OpenSAP courses material and SAP Learning Hub materials are usually supplied in order to help developers combine their theoretical knowledge with a practical situation. These resources are strong at teaching developers how ABAP responds to SAP’s changing terrains and assist their preparations for SAP certification exams.

In other words, resources for SAP ABAP include practitioner guides, scholarly research, and technical manuals. From elementary coding practices all the way to complex integration and modernization methods, the sum of literature available provides a full picture of ABAP applications in SAP environments. By expanding SAP to the new technologies of S/4HANA and cloud-based offerings, we will see continued contributions to available resources on ABAP.

**Table 2.1 : Literature review**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S. No.** | **Paper Title [Cite]** | **Journal/**  **Conference (Year)** | **Tools/ Techniques/ Dataset** | **Results** | **Limitations** |
| **1.** | ERP Logs and Its Use for Process Mining Student Learning Purposes | Innovative Technologies and Learning Conference (ICITL 2024), Springer | Educational use of ERP logs; process mining techniques | Explored the use of ERP logs in teaching process mining, enhancing student understanding through practical application. | Focused on educational settings; may not directly translate to industrial applications. |
| **2.** | Opportunities and Challenges in Data Analysis Using SAP: A Review of ERP Software Performance | International Journal of Management Information Systems and Data Science, 2024 | Literature review, comparative analysis | Identified strengths and challenges of SAP in data analysis, emphasizing real-time processing and scalability. | Broad review; may lack in-depth technical analysis. |
| **3.** | A Generic Approach to Extract Object-Centric Event Data from Databases Supporting SAP ERP | Journal of Intelligent Information Systems, 2023 | Object-centric event data extraction, SAP ERP databases | Presented a method for extracting object-centric event data to support process mining. | Requires further validation across diverse SAP ERP implementations. |
| **4.** | How Do SAP Developers Use Knowledge Resources Such as the SAP Community? | R&D Management Journal, 2023 | Grounded theory methodology, developer interviews | Explored how SAP developers utilize knowledge resources, highlighting the role of communities in problem-solving. | Based on interviews; may not capture all usage patterns. |
| **5.** | Analyzing Interconnected Processes: Using Object-Centric Process Mining to Analyze Procurement Processes | International Journal of Data Science and Analytics, 2023 | Object-centric process mining; SAP ERP data | Demonstrated the application of object-centric process mining to analyze procurement processes within SAP ERP systems, enabling the identification of complex interdependencies. | Prototype limited by in-memory approach and customization options; scalability concerns noted. |
| **6.** | Multi-Objective, Multi-Project Scheduling Solver Implementation Using SAP ABAP Language | Production Systems and Information Engineering, 2022 | ABAP-based scheduling solver, multi-project scheduling | Developed a solver for multi-project scheduling using ABAP, demonstrating its applicability in project management. | Specific to scheduling; may not address other project management aspects. |
| **7.** | SAP ERP Implementation for a Manufacturing Company | ERPCorp Case Study, 2021 | SAP ERP modules (FI/CO, MM, COPA); discrete manufacturing processes | Detailed an end-to-end SAP ERP implementation in a manufacturing firm, highlighting integration of various modules and improved operational efficiency. | Case-specific insights; generalizability to other industries may be limited. |
| **8.** | Telco SAP ERP Implementation: A Swenta Global Success Story | Swenta Global Case Study, 2021 | SAP ERP; telecommunications industry | Presented a successful SAP ERP implementation in a telecom company, resulting in enhanced operational efficiency and real-time reporting capabilities. | Specific to telecom industry; broader applicability requires further study. |
| **9.** | ERP Post Implementation Review with Process Mining: A Case of Procurement Process | Procedia Computer Science, 2017 | Process mining techniques; procurement process data | Applied process mining to review ERP post-implementation, identifying inefficiencies in the procurement process. | Case-specific findings; may not be applicable to other processes or organizations. |
| **10** | Strategic Technology Management in Practice: Dynamic SAP-LAP Analysis of an Auto Component Manufacturing Firm in India | Global Journal of Flexible Systems Management, 2010 | SAP-LAP framework, case study methodology | Applied SAP-LAP analysis to assess strategic technology management in an Indian manufacturing firm. | Single case study; findings may not be generalizable. |
| **11** | Teaching ERP Programming Using SAP ABAP/4 | IGI Global Book Chapter, 2007 | Educational pedagogy, ABAP/4 programming | Proposed a teaching methodology for ERP programming using SAP ABAP/4. | Educational focus; not centered on industrial application. |
| **12** | Preprocessing Support for Large Scale Process Mining of SAP Transactions | Business Process Management Workshops (BPM 2007), Springer | ERP log analysis system; ProM process mining tool | Developed a system to define and transform SAP transaction data for effective process mining using ProM. | Focused on preprocessing; does not address the entire process mining pipeline. |

**2.2 KEY GAPS IN THE LITERATURE**

With an increasing pool of research on SAP ERP systems, real-world implementation of tailored SAP solutions that follow closely to real-time sales and distribution is still lagging behind. While new research casts light on improvements and automation, a lot of studies are theoretical or only partial in what they discuss, and do not show the role of SAP’s inherent tools such as module pool programming, ABAP Objects, and integrated workflows in real application in achieving full functionality. For this reason, the knowledge which arises from research does not always translate to actual returns in the operations of manufacturing companies.

In addition, there is a noticeable lack of emphasis on the integration of multiple SAP technologies within a single, cohesive solution. Existing literature often focuses on isolated use cases—like scheduling, procurement, or process mining—but does not adequately address projects that demand user interaction through custom GUIs, database validations, automated email notifications, and Smartforms all working in tandem. The need for studies that explore how these elements function together within a modular SAP application remains largely unmet.

Another area where literature falls short is the role of customization in enhancing user experience and internal efficiency. Many implementations documented in past research rely heavily on SAP standard modules and overlook the need for personalized interfaces, access-based dashboards, or role-specific functionalities that modern businesses often demand. This is especially relevant for mid-size firms like SmartSale Ltd., where flexibility and rapid system adaptability are crucial for staying competitive.

Furthermore, the educational studies that do exist, while beneficial in theory, do not fully prepare developers for the complexities of real-world development. There is little emphasis on debugging practices, performance optimization, modularization techniques, or the challenges faced during workflow integration. These are precisely the areas that developers struggle with during enterprise deployment but are often ignored in both academic and industrial case studies.

Lastly, although object-centric process mining has been explored in a few recent papers, its application within core SAP modules like SD (Sales and Distribution) or MM (Materials Management) has not been studied extensively. Most of the use cases focus on procurement or financial data analysis, leaving a gap in understanding how sales inquiries, quotations, cancellations, and order fulfillments can be modeled and improved using these newer process analysis techniques.

In conclusion, the existing literature lacks holistic, full-stack SAP solutions that bridge user interface design, backend processing, and process automation—all of which are essential for a dynamic sales and distribution environment. This project attempts to address these deficiencies by developing a comprehensive custom SAP module for SmartSale Ltd., combining technical depth with business relevance and process transparency.

**CHAPTER 3: SYSTEM DEVELOPMENT**

**3.1 Requirements and Analysis**

Before initiating the development of any SAP-based solution, it is crucial to perform a thorough requirement analysis to ensure the system is aligned with business objectives and end-user expectations. In this project, the objective was to build a custom SAP application for SmartSale Ltd. to streamline and digitize their Sales and Distribution processes. The analysis phase focused on identifying both the functional and non-functional needs, understanding the users involved, and defining the necessary data models through custom tables.

**3.1.1 Functional Requirements**

* **LOGIN FUNCTIONALITY:** The login screen is designed with usability and security in mind. It acts as the entry point for two types of users: customers and admins. While the admin uses a static login ("MyAdmin"/"Admin@123"), customer logins are validated against entries in the ZCUSTOMER table. This table contains all the necessary fields such as customer ID, name, password, security question, and answer. These values are checked during login to ensure that only authorized individuals can access the system.

To further secure access, field-level validations are applied. This includes mandatory fields for username and password, checks for empty or incorrect entries, and limited login attempts to prevent brute-force attacks. The password is masked during entry to protect sensitive information, and proper messages are shown in case of failed login attempts. Additionally, a clickable "Register Now" text is provided for new users, which acts as a navigation link to the registration screen.

This separation of admin and customer access allows the system to load different functionalities based on the user role. The admin has access to process-level data and back-end order management, while customers are limited to their own order and quotation management. By clearly dividing the flow from the login screen itself, the user experience is streamlined, and access rights are strictly maintained.

* **CUSTOMER REGISTRATION :** The registration functionality is designed to be simple yet secure. New users are required to provide basic details such as name, password, security question, and answer. The system auto-generates the customer ID by reading the last entry in the ZCUSTOMER table and incrementing it. This ensures that customer IDs are always unique and in a sequential order.

Input validation is a key focus in this module. Passwords are checked for minimum length and complexity, while the 'Re-enter Password' field ensures that users do not accidentally save incorrect credentials. The use of a security question and answer adds an extra layer of recovery in case of forgotten passwords. All successfully validated data is inserted into the ZCUSTOMER table using modularized function modules for reusability and consistency.

After successful registration, a confirmation message is shown and the user is redirected to the login screen. Behind the scenes, the custom table is updated, and the workflow is ready to allow the new customer access to the ordering features. This process ensures that only valid and securely registered users are allowed into the system.

* **ORDER MANAGEMENT:** Once logged in, customers land on a personalized screen that displays their open orders using an ALV Grid. This grid shows order ID, date, and current status, making it easy for users to view and track their orders. The customer can select one or more orders from the list and perform actions using the GUI buttons provided below the grid.

The available buttons include "Create Inquiry," "Follow-up," "Cancel," and "Accept/Reject." Each button is associated with a specific backend process. For example, clicking "Create Inquiry" navigates the user to the standard SAP Inquiry screen and sends an email notification to the admin team upon successful creation. The "Follow-up" and "Cancel" buttons trigger workflows and email messages to inform the admin of the requested actions, ensuring that communication remains active throughout the order lifecycle.

The "Accept/Reject" functionality becomes active when a quotation has been sent for an order. Customers can respond to quotations, and their choices are recorded in the ZQUOTATION table. This structured approach to order handling ensures that all customer interactions are captured, processed, and visible to the admin for further action. It also introduces real-time interactivity to the custom-built SD module.

* **ADMIN TASKS :** The admin panel is equipped with a hierarchical dropdown filter that allows the admin to view orders based on their current status: Open, Cancelled, or Completed. This classification helps in prioritizing tasks such as checking stock availability, sending quotations, or fulfilling accepted orders. Administrative efficiency is retained because the selected order details are properly given in an orderly look. Administrators can check a ZPRODUCTS table to ensure that there are stocks for materials placed in open orders. In case inventory levels allow that, the administrator can generate a quote by entering a recommended price in the showed popup. Once submitted, the customer receives e-mail with quotation attached. Once accepted, if the quotation is then accepted, the administrator can proceed on the fulfillment of the order and altering the status of the order as finished in the ZORDERS table.

In the final step, after order completion, an invoice is generated using Smartforms and sent to the customer via email. Cancelled orders can also be revisited, where the admin has the option to send a revised quotation, depending on the case. This workflow ensures that all stages of the sales cycle — from inquiry to invoicing — are handled smoothly and systematically within the SAP environment.

* **ERROR HANDLING :** Effective error handling is crucial to ensure smooth system operation and user satisfaction. The system includes built-in validations and responses to common failure scenarios.

For instance, if a customer places an order exceeding available stock, the system checks inventory levels and displays a warning message if the quantity is insufficient. This prevents failed orders and informs the user immediately.

In cases where automated email notifications fail due to workflow issues, the system logs the error and alerts the admin for manual follow-up, ensuring no communication is missed.

Additionally, if a customer tries to accept or reject a quotation without a valid email or linked quotation, the system stops the status update and notifies the user with a clear message. These checks help maintain data accuracy and process integrity throughout the application.

**3.1.2 Non-Functional Requirements**

* **SECURITY:** Security is one of the most critical non-functional aspects of the system. The entry of the passwords should be masked and the use of encryption should be applied to secure the information of the clients when the information is stored. This ensures a privacy of personal details across all instances of user sessions. For the purpose of safeguarding the application from SQL injection attacks, validation controls have to be incorporated. An effective role-enabled permission model should be implemented with good access control. By doing this, the only people that will be getting access in accessing the sensitive piece of back-end materials and the admin control are the authenticated admin users.
* **DATA INTEGRITY:** Consistency and dependability can only be sustained if the integrity of data is consistently maintained within the application. Every data entered by any user in the database has to pass through rigorous validations before being stored in the database. All the fields in question must be checked to ascertain the existence of all required data, that it is in the needed form and that it has been put within the required character limits. Further, in order for maintain the referential integrity, is necessary that relationships between tables such as ZORDERS, ZINQUIRY and ZQUOTATION are appropriately established. in other words, data in a single table needs to fit in with consistent related data across other tables. In cases that demand presence of a lot of data entry or checks and if manual updates are used, the use of LSMW or BDC improves accuracy of data by counterbalancing human error risks.
* **USABILITY:** Usability plays a key role in ensuring that the application is user-friendly and accessible. The interface should be built following SAP GUI best practices, with consistent layouts, clear field labels, and logical screen flows. UI components like dropdown lists, popup windows, and ALV Grids should be incorporated to make navigation simple and interactive. These elements not only improve visual clarity but also streamline the user experience. Further, a properly crafted system should provide users with useful confirmations and complex error messages, with which users will be able to identify and correct input errors easily and confidently.

**3.1.3 User Types**

The design of the system is simple; it is easy to identify the two leading user roles, namely Customer and Admin and each has features that are tailored appropriately to their roles in sales and distribution.

**CUSTOMERS :** There are key front-end features of the system, which form the main platform for customer interaction. Starts by inputting their details in a guided sign-up screen, then doining a log-in within their own login details. After login, clients are able to view their open orders, contact for the product questions, and select approval or rejection of given bids. Customers may also request to withdraw an order or check out any forthcoming order they may have. The fact that this system is user-friendly, means that customers through it will be able to trace the progress of their orders from the point of initiation to the end making the process transparent to the customers and also guaranteeing control over transactions.

**ADMINS :** Administrator, unlike users, take care of the ins-and-outs of the system. They are all given the responsibility of reviewing customers’ requests, checking product stock and quoting, finalizing orders that have been accepted. Besides processing cancellations, admins take care of invoice generation through Smartforms and update correct order statuses. To increase the accuracy of the oversight and the decision-making process, the intended configuration features detailed controls and layouts such as hierarchical lists and status-based order filters.

One of the main characteristics of this system is the provision of role-based access control within it. This strategy reduces the chance of user mistakes or untoward actions, data confidentiality, and system security are also kept under wraps. Customer access is handled dynamically and is retrieved when run time from ZCUSTOMER table while static pre-defined admin credentials (MyAdmin / Admin@123) is provided to administrators during implementation and testing phases.

By assigning clearly defined roles, the system promotes efficiency, minimizes confusion, and enforces a logical separation of duties that aligns with real-world business workflows.

**3.1.4 Custom Tables Used**

The following custom DDIC tables were designed and implemented to handle data management throughout the project:

**Table 3.1.4 Custom Table Names**

|  |  |
| --- | --- |
| **Table Name** | **Description** |
| ZCUSTOMER | Stores customer-related data such as customer ID, name, encrypted password, security question, and answer for login and authentication. |
| ZPRODUCTS | Maintains the list of all available products including product ID, name, unit price, and current stock quantity for availability checks. |
| ZORDERS | Captures customer order headers including order ID, order date, status (open, completed, canceled), and a link to the customer who placed it. |
| ZINQUIRY | |  | | --- | |  |  |  | | --- | | Logs customer inquiries related to product orders and maintains reference to the selected products and customer details. | |
| ZQUOTATION | Manages quotations provided by the admin, linked to specific orders. It includes quotation amount and the status (Accepted or Rejected) by the customer. |

These tables are essential to support the flow of login, order creation, inquiry management, quotation handling, and order completion in a structured and integrated manner within the SAP environment.

**3.2 PROJECT DESIGN AND ARCHITECTURE**

The core objective of the "SmartSale Sales & Distribution Portal" project is to create a robust and standardized application to streamline the end-to-end sales and distribution process within the company. This custom SAP solution is developed using ABAP module pool programming and integrates seamlessly with SAP’s standard SD functionalities. The architecture ensures secure, modular, and scalable handling of customers, orders, quotations, and invoicing, while also allowing for real-time communication between customers and the admin team.

This section outlines the system's design strategies, focusing on modular design, database-driven operations, and user role-based navigation for customers and administrators.

1. **Modular User Interface and Navigation:**

* The application is divided into independent screens for login, customer registration, profile update, customer dashboard, and admin dashboard.
* Each screen is connected through a custom transaction code, and navigation is handled using CALL SCREEN logic, ensuring easy maintenance and enhancement.
* Role-based access control directs customers and admins to their respective screens after login verification.

1. **Data Management via Custom Tables:**

* Core tables like ZCUSTOMER, ZPRODUCT, ZORDER, and ZORDER\_ITEMS are used to handle customer details, product stock, order tracking, and quotation/invoice details.
* All insertions and updates to these tables are done through ABAP logic triggered by button actions in the screen flow, ensuring data consistency and avoiding direct table modifications.
* The customer ID is auto-generated based on the latest entry, ensuring uniqueness and avoiding manual errors.

1. **Workflow and Email Integration:**

* Email notifications are triggered automatically based on specific actions such as inquiry creation, order cancellation, or quotation updates.
* SAP Workflow or function modules are used to trigger these communications in real-time, ensuring immediate admin/customer notification.
* Admin actions like sending quotations or confirming order completion also result in emails being sent, keeping the customer informed at every stage.

1. **Smartforms for Invoicing:**

* Upon successful order completion, a detailed invoice is generated using SAP Smartforms and is sent to the customer's registered email address.
* The Smartform includes dynamic data pulled from the order and customer tables, providing a professional and accurate document for transaction records.

1. **ALV and GUI Standards:**

* Customer dashboards use ALV grids to present orders in a structured format, enabling actions like selecting multiple orders and triggering batch operations.
* All custom screens follow SAP GUI standards, including proper field alignment, logical tab order, and consistent button styling for enhanced usability.

The project design ensures a modular, scalable, and user-friendly SAP application tailored to SmartSale Ltd.’s sales and distribution needs. By integrating custom screens, workflow-based communication, and standard SAP functionalities, the architecture supports seamless order management and real-time interaction. This structured approach lays a strong foundation for future enhancements and enterprise-wide adoption.

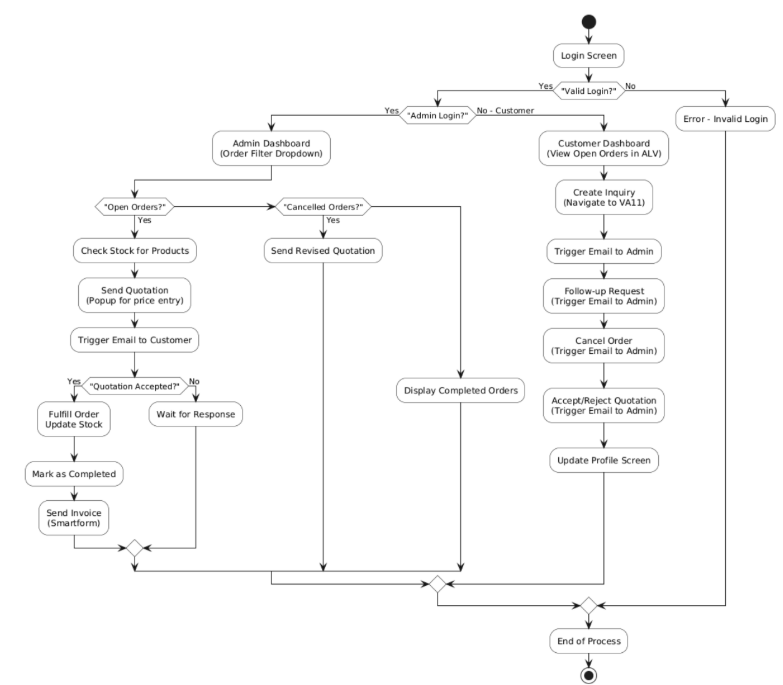


Figure 3.2.1 Flowchart of the Project

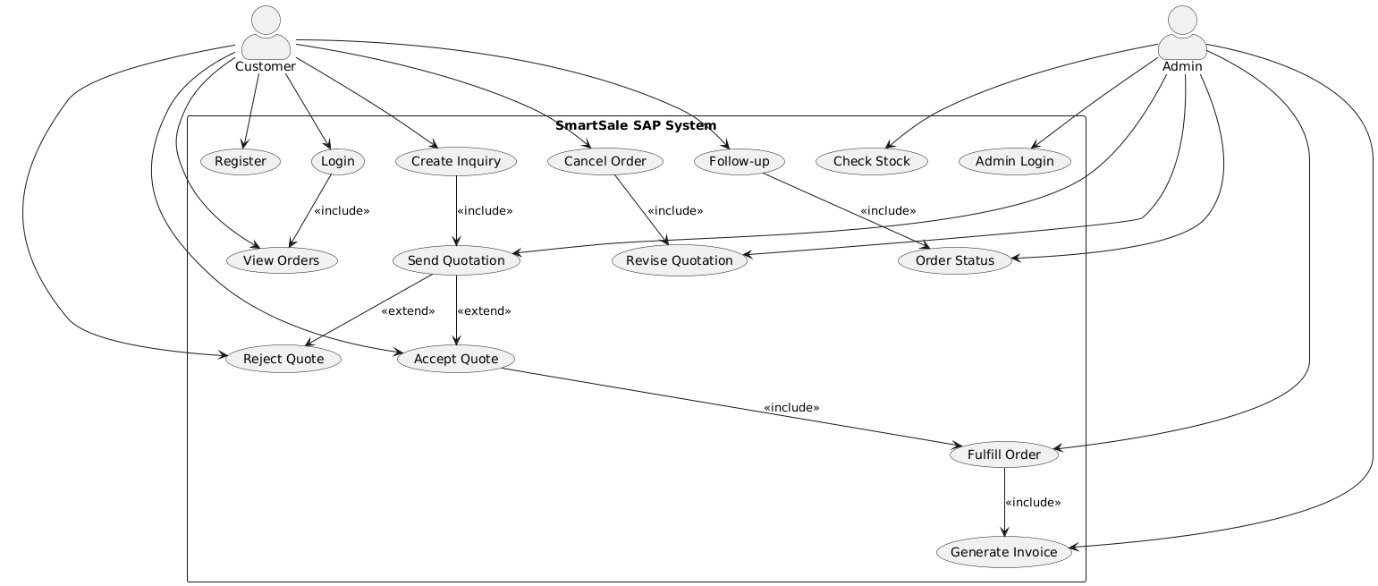


Figure 3.2.2 Use Case Diagram

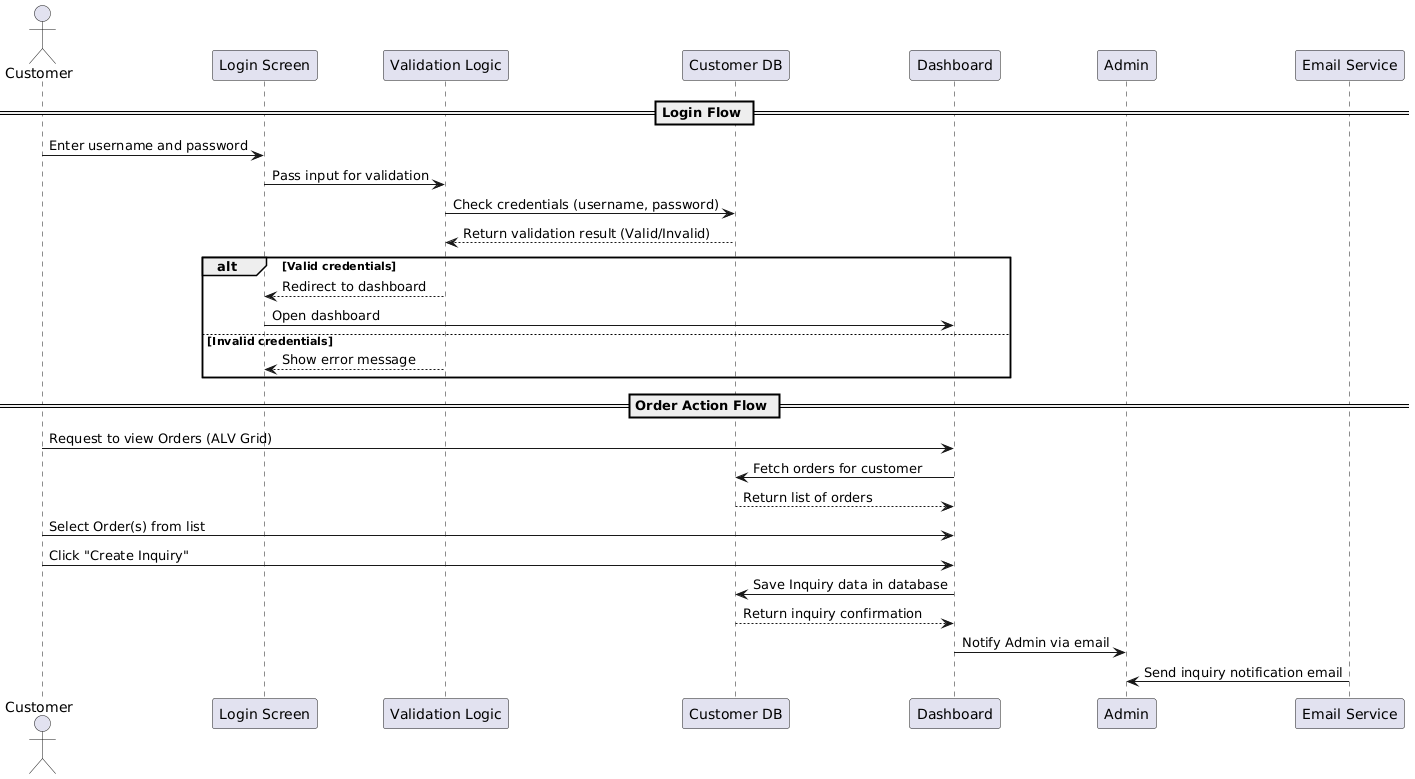


Figure 3.2.3 Sequence Diagram

The sequence diagram demonstrates two essential processes on the personalized SAP Sales & Distribution portal, the login flow, and the order inquiry action flow. The customer begins the login process by doing a login with their username and password. When the credentials are entered, the validation logic looks for them in comparison with the customer database. In case the entered login details are right, the customer is authenticated and redirected to his/her customized dashboard. In case the application login details diverge, an error message will appear for purpose of system security and non-accessibility.

Once logged in, customers can also see their previous orders from the order action flow. With order data fetched from the database, the system, through visual organization, presents it in an ALV grid. If particular orders are chosen, then the customer has the ability to make an inquiry by pressing the “Create Inquiry” button. Through validation of the submission, the system logs the inquiry in the database. This creates a fast information flow and minimizes the attention by the administrative staff to client-related requests. The sequence diagram shows how modules such as the login screen, database, dashboard, administrator interface, and even email service led to a seamless and friendly experience.

**3.3 DATA PREPARATION**

Efficient preparation of data is crucial for both verification and confirmation of the functionality of the one of a kind Sales and Distribution Portal to be used for SmartSale Ltd. This phase ensures all system components interact correctly and mimic real-time business scenarios. The data was prepared and loaded into custom Z-tables using automated ABAP methods, primarily through BDC programs and module pool interfaces.

1. **Customer Master Data:**

To support login and registration functionalities, customer details such as Customer ID, Name, Password, Security Question, and Answer were loaded into the ZCUSTOMER\_DETAILS table. A BDC (Batch Data Communication) program was created to simulate bulk customer registration, reducing manual entry and ensuring data consistency. Each Customer ID was auto-generated based on the last record, and validations were implemented to prevent duplicate entries and ensure data accuracy.

1. **Material Master Data:**

Product and material-related information such as Material ID, Name, Price, and Available Stock was uploaded into the ZMATERIAL table. A custom BDC upload process was developed for efficient insertion of large volumes of material records. This data formed the basis for product selection, stock availability checks, and pricing during the inquiry and order processes.

1. **Transactional Data Setup:**

To support order processing features, sample transactional data was generated for inquiries, quotations, and customer orders. These were stored in dedicated custom tables like ZINQUIRY\_HEAD, ZINQUIRY\_ITEM, ZQUOTATION\_HEAD, and ZQUOTATION\_ITEM. Data was prepared to simulate different order statuses such as "Open," "Cancelled," and "Completed." These records helped in thoroughly testing the flow of actions like follow-ups, cancellations, quotation acceptance or rejection, and order fulfillment.

1. **Data Entry Methods:**

All necessary data was populated using ABAP-based BDC upload programs or directly through the custom module pool input screens. No data was inserted via standard transaction codes or table maintenance generators (TMG), to ensure system realism and better control during validation and workflow testing. Data consistency was further verified using custom ALV reports developed during the project.

1. **Test Scenarios Covered:**

The prepared dataset supported a wide range of test scenarios including new customer registration, login validation, profile updates, inquiry creation with valid/invalid data, triggering follow-up and cancellation workflows, quotation generation, and final order fulfillment. This structured approach ensured that each module and event in the system could be tested reliably and efficiently.

**3.4 IMPLEMENTATION**

The implementation of the custom SAP Sales & Distribution application for smartSale Ltd. follows a modular, phased approach to ensure seamless development, testing, and deployment. The foundation starts with the creation of custom tables using the SAP Data Dictionary, designed to manage critical business data such as customer credentials, product inventory, and transactional details. The customer master table includes fields like Customer ID, Name, Password, Security Question, and Answer, while the product table manages product ID, name, price, and stock availability. Proper primary keys and field validations are defined to maintain data integrity across all operations.

At the core of the application lies a module pool program, linked to a custom transaction code that acts as the main entry point. The program contains multiple dynpros (screens) built using SAP dialog programming to support user interaction and business processing. It begins with a login screen, where users can enter credentials for authentication. Features like a hyperlink for customer registration and admin-specific login credentials are also included. After a successful login, users are redirected to their respective dashboards—customers access their order dashboard, while admins enter the order management panel. Each dynpro manages its own processing logic using ABAP code for data validation, flow control, and backend integration.

The customer registration screen ensures secure and structured onboarding. It auto-generates a unique customer ID and validates inputs such as password confirmation and security question selection. Upon registration, the customer data is stored in the custom table. Customers, once logged in, are presented with a detailed ALV grid displaying all open orders. They can select orders and perform multiple actions like creating inquiries, sending follow-up requests, initiating cancellations, or responding to quotations. Each of these actions is integrated with SAP’s email service to notify the admin team in real-time using dynamic content via ABAP email triggers.

To enhance user control and personalization, an additional screen—Screen 500 for Profile Updation—has been incorporated. This screen allows customers to view and update their personal details, such as name, password, and security question/answer. The fields are pre-populated based on the logged-in customer’s existing data and can be modified through a structured form. The system performs necessary validations before updating the records in the customer table. This feature improves the user experience by offering self-service capabilities while maintaining data security and integrity.

On the administrative side, the dashboard is equipped with dropdown filters for viewing orders by their current status—Open, Cancelled, or Completed. Admins can manage open orders by verifying stock availability, sending quotations through a popup input interface, and fulfilling orders once quotations are accepted by customers. Invoices issued using SAP Smartforms are produced and emailed as PDF attachments to recipients directly. When handling orders with more than one item, the system checks stock availability per item and changes status accordingly with order fulfillment process. At the same time for better respond and speediness, the administrator can decide to sharpen and re-send quotations in the event of cancellation of orders.

Modularity should be well supported by a thorough concentration as its implementation develops. In order to perform mundane processes like dispatch of mail, validation of stock, and creation of customer IDs, the development team develops re-usable function modules. This approach will be helpful to maintain the code base making it reasonably easy to manage. By modularizing the screen logic into include programs, we modularize presentation logic, event handling, and database manipulation creating an easier to read and debug code.

Not only does this improve the visual presentation of the admin and customer screens, but it also adds to the level of visual clarity and user action. The customers are able to easily categorize and (re)search through their orders, and the administrators have powerful tools to monitor statuses of the orders thanks to hierarchical views for data. Buttons and confirmation popups are practical dynamic elements that help enhance ease-of-use and ensure intent of the user gets confirmed before action taken.

The SAP CL\_BCS email framework rationalizes communication workflow management, as every significant activity – receipt of an inquiry, QUO response, or INVO creation etc – is taken care of automatically, with structured messages. In order to keep emails clear and trackable, they are augmented with dynamic information, appropriate user details, and PDF attachments. Each person’s email transaction is recorded within its own dedicated table for tracking and auditing purposes.

In addition, SAP Smartforms are critical in issuing invoices of superb standard. Using these forms automatically gathers the information from orders and presents them in a clear invoice template that touches all the important financial and item-level points. The design of the documents, that are converted to PDFs, emailed, and includes considerations for future features for example barcode print or QR code, digital signature etc.

Optimization of sales operations by smartSale Ltd. by means of automation and simplicity is supported by comprehensive and flexible SAP implementation.

**3.4.1 TOOLS AND TECHNOLOGIES USED**

The development of a custom SAP Sales & Distribution application by smartSale Ltd. employs many SAP-native tools and technologies that combine in order to provide a fast and easy to use solution. The core of this application is put forward through SAP’s Core programming language, ABAP (Advanced Business Application Programming). ABAP is used to manage the module pool program, perform the database operations, implement validations and also develop the modules behind business activities. As it effortlessly fits into the SAP’s runtime environment, it is an ideal application for developing scalable and secure custom applications.

Dialog Programming, equivalent to SAP Module Pool Programming, constitutes the chief development strategy chosen for this project. It allows for the design of individualized Dynpro screens which allow immediate user interaction. These screens are created using Screen Painter tool, and the working of event-driven Flow Logic (PBO/PAI) enables them to run without glitch. This strategy allows easy navigation between the different screens used in order management, updates, enrollments and also log in. Design on each dynpro is aimed to provide a smooth and interactive platform, with ABAP code used for automatic management of data and validation procedures.

In order to be able to work with the data effectively, the project utilizes the SAP Data Dictionary to develop custom tables. The tables in the project are used to maintain crucial information such orders, products, workflow history and customer information. With strong database layer design, the application expresses integrity and makes maintenance tasks very easy. Presentation of data is carried out by both the administration and customer views utilizing ALV (ABAP List Viewer) Grids. Using functionalities such as sorting, filtering, as well as row selection, ALV enables users to manage and engage with orders effectively during task such as asking a quote or making an inquiry response.

By dealing with Application communication dynamics, SAP Business Communication Services (CL\_BCS) enables automated email notification to be triggered when a user does specific activities. CL\_BCS ensures delivered emails include personalized content, making communication more efficient and adding value to user interaction where communicating on the aspects of configuring the confirmation of an inquiry, updating on the status of quotation and invoice notifications. E-mails are often comprised of other uploads in PDF format or created using SAP Smartforms. Invoices that contain product snapshots, tax, totals, and customer specifics that are customized are created with the help of smartforms.

The arrangement of ABAP code as reusable function modules and include programs contributes towards an application being good practice compliant. By applying function modules to dedicated functions such as stock validation, email transfer and ID generation, the application improves code modularity and simplifies debugging activities. By splitting responsibilities between include programs, the codebase remains in order and easier to manage in due time. A case in point would be the one including program control screen logic, the next managing database interaction, and the last overseeing input validation, separately.

Together, these tools and technologies provide a powerful development framework that supports the end-to-end sales and distribution operations at smartSale Ltd. The modularity, reusability, and automation achieved using SAP’s native capabilities ensure that the application remains robust, user-friendly, and ready for future enhancements.

**3.4.2 PROJECT CODE SNIPPETS**

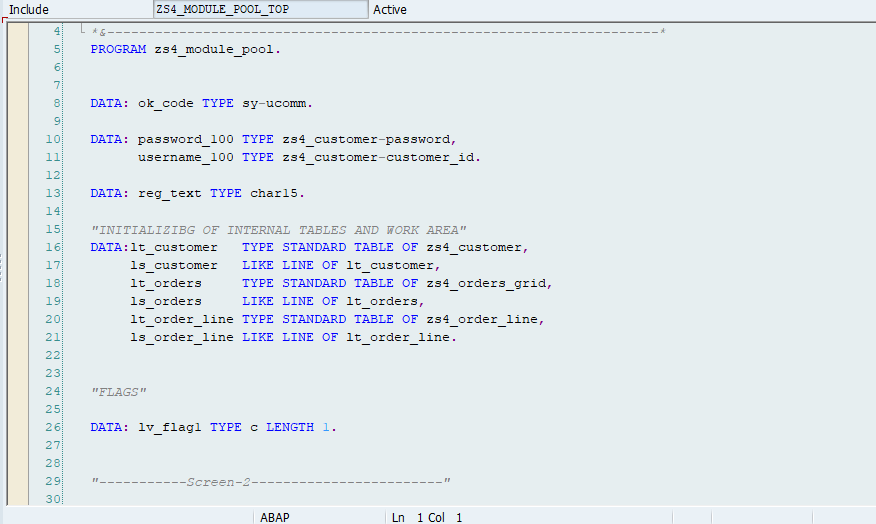


Figure 3.4.2.1 Data Declaration For Screen 1.

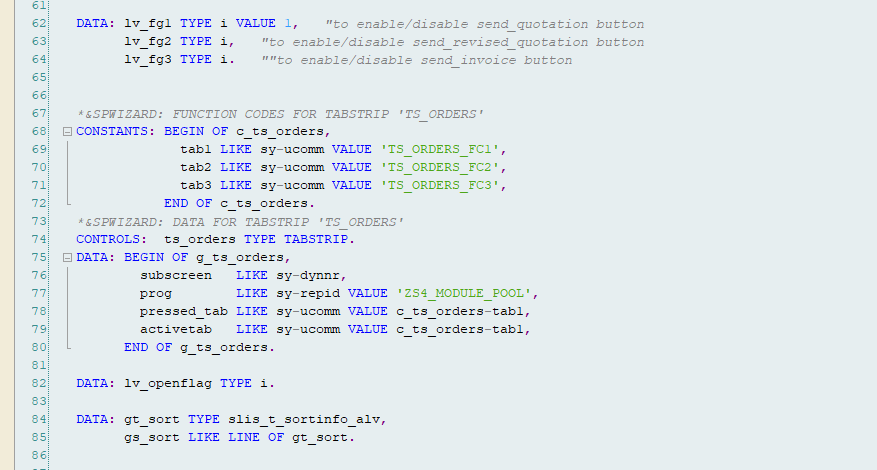


Figure 3.4.2.2 Data Declaration For Screen 3.

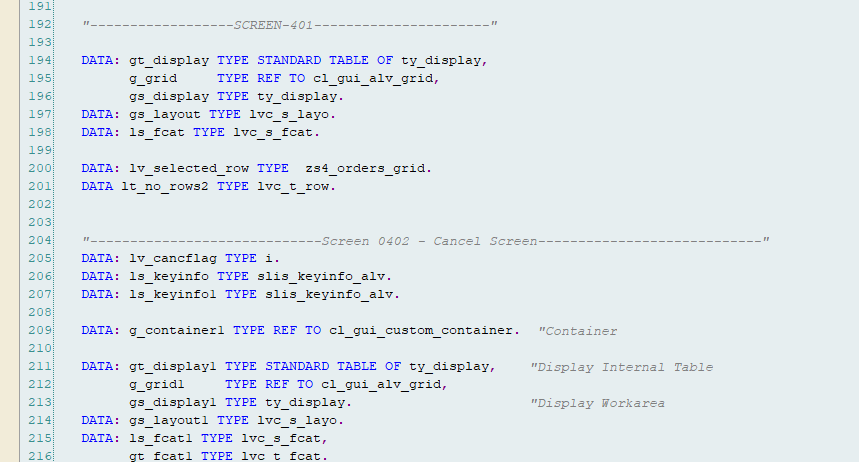


Figure 3.4.2.3 Data Declaration For Screen 4.

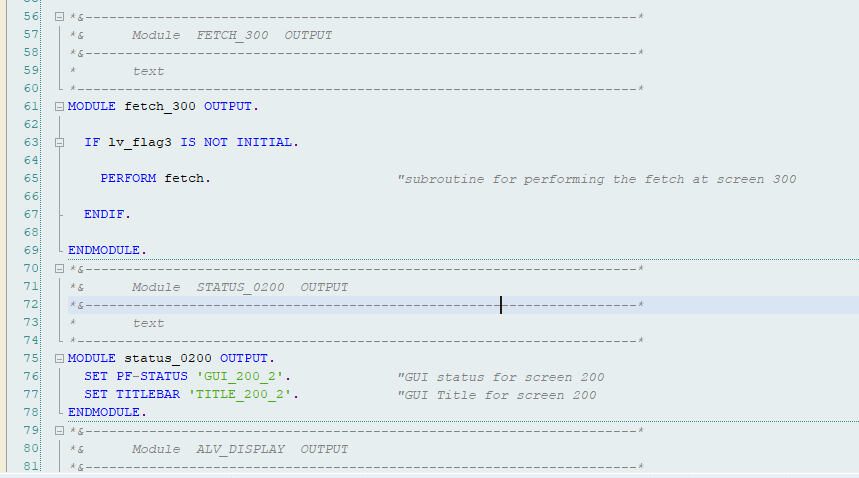


Figure 3.4.2.4 PBO (Process Before Output) Module for Screen 1.

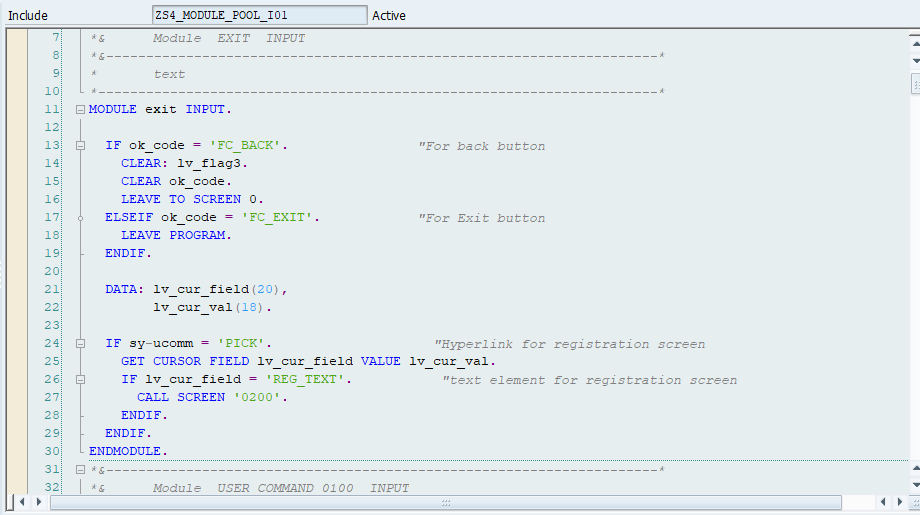


Figure 3.4.2.5 PAI (Process After Input) Module for Screen 1.

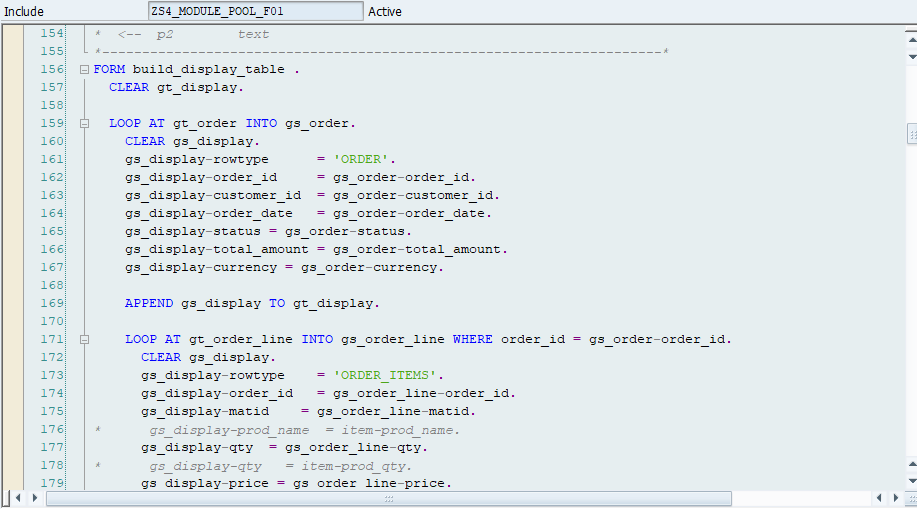


Figure 3.4.2.6 Form-Routines Module for Build\_FieldCatalog.

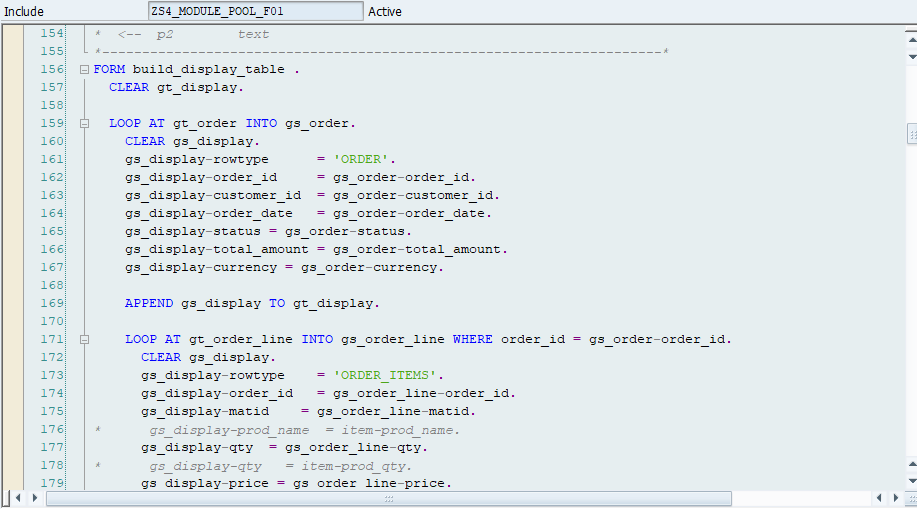


Figure 3.4.2.7 Form-Routines Module for Build\_Display\_table.

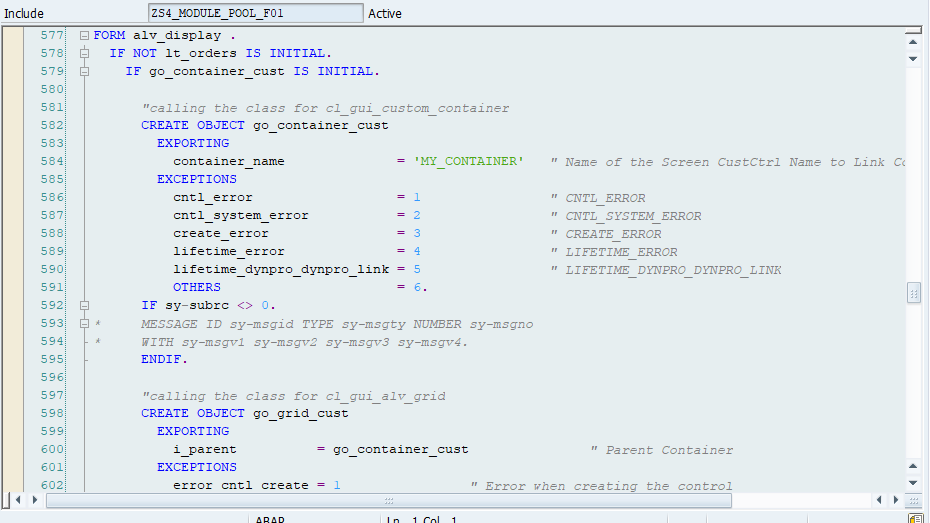


Figure 3.4.2.8 Form-Routines Module for ALV Display.

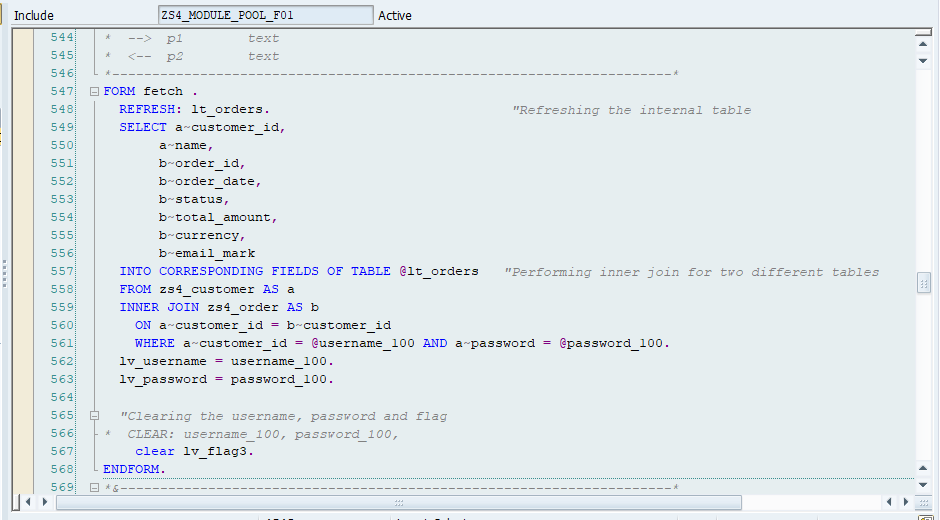


Figure 3.4.2.9 Form-Routines Module for Fetching Data.

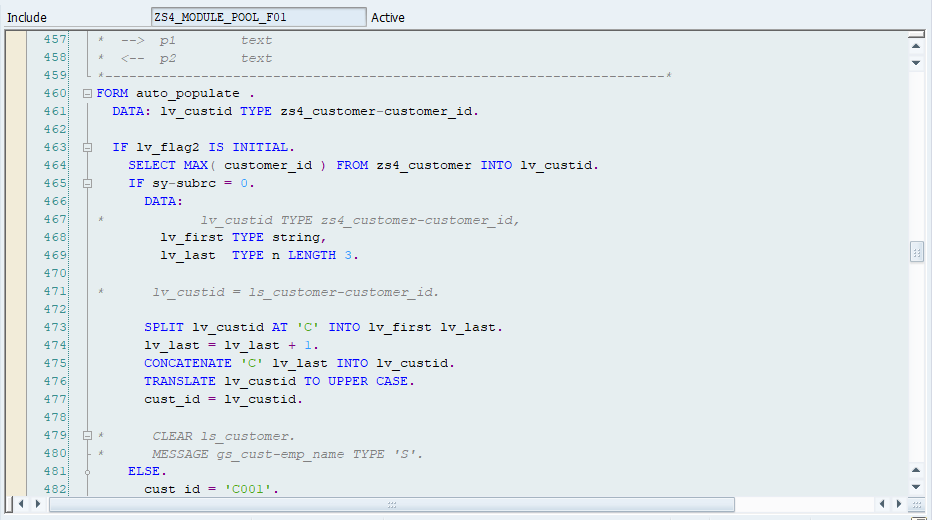


Figure 3.4.2.10 Form-Routines Module for Auto Populate the customer ID.

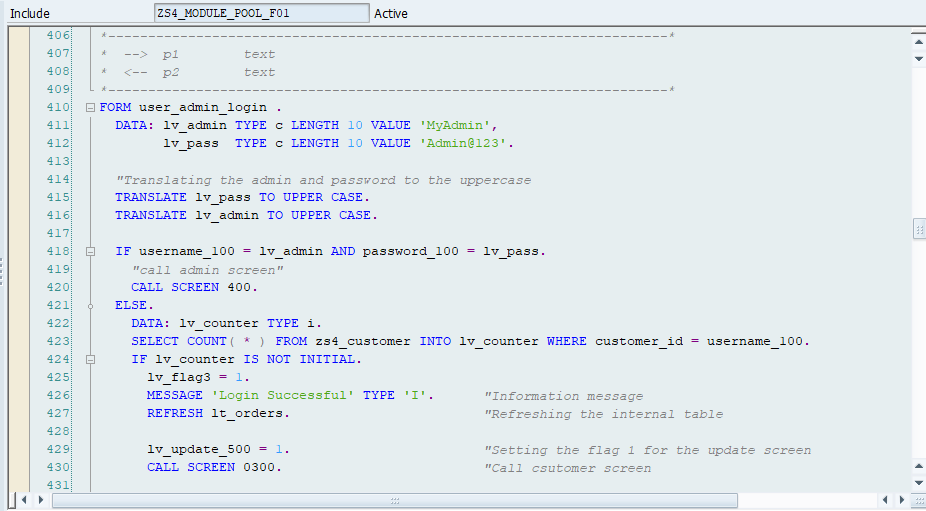


Figure 3.4.2.11 Form-Routines Module for User\_Admin\_Login.

**3.5 KEY CHALLENGES**

During the development of the custom SAP Sales & Distribution portal for smartSale Ltd., several technical and functional challenges were encountered. These challenges required thoughtful design, debugging, and ABAP programming strategies to ensure a stable and user-friendly system.

1. **Handling Screen-to-Screen Navigation**

One of the fundamental challenges was managing smooth and logical navigation between multiple dynpro screens in the module pool program. Each user interaction—such as moving from login to customer dashboard or from an order list to an inquiry creation screen—had to maintain session data and context. Ensuring that variables were passed correctly between screens, while handling exceptions and invalid inputs, required precise control using CALL SCREEN, LEAVE TO SCREEN, and conditional checks in PBO/PAI logic.

1. **Auto ID Generation**

Generating unique customer IDs dynamically during registration posed another key challenge. In order to record a new entry the system needed to obtain the latest ID from the database and increment it by one since IDs had to be unique and in a sequence. That no duplicates were created and that performance would be stable even in case of concurrent registrations was of paramount significance. In order to preserve data integrity, the system implemented strong locking approaches as well as modularity design principles.

1. **ALV Grid Selection Logic**

By implementing ALV Grids to receive order data, we improved the function and usability in the system. One of the great challenges was the ability to accurately identify and process multiple selected rows. In order to proceed with follow-up, cancellation, or quotation acceptance works, users had to find out the relevant orders. To do this, systematic utilization of ALV events and the GET\_SELECTED\_ROWS method facilitated development of logic that identified selected lines, captured relevant order IDs, performed relevant backend operations, and did not interfere with other records.

1. **Workflow Email Integration**

Automation of communication was based on integrating email notifications within SAP’s CL\_BCS framework. It was necessary to develop the required code to facilitate email time control and inclusion of important dynamic data which include name of the customer, an order ID, or quotation data only if necessary. In order to address workflow failures and increase reliability, one solution was needed for missing email addresses, bad format, or connectivity issues.

1. **Smartform Dynamic Generation**

The biggest challenge involved creating Smartforms that could be adapted to such order data as different line items, or personalized headers. Using the order ID to single out specific information, the form needed the capability to get, arrange, and output that data in a tidy formatted and sequenced PDF. Ensuring that this PDF could be attached to an email and delivered to the customer with minimal delay required coordination between Smartform output, spool processing, and the CL\_BCS email logic.

Each of these challenges contributed to the learning curve of the project and helped in building a more robust and production-ready solution. Overcoming them led to better code modularization, enhanced user experience, and a deeper understanding of real-world SAP development practices.

**CHAPTER 4: TESTING**

**4.1 TESTING STRATEGY**

* **Manual and Unit testing:**

Manual testing for the smartSale Ltd. SAP application involves validating each user interface component and workflow without the use of automated tools. Testers will manually simulate real-world user actions such as login, registration, order creation, and administrative order handling to ensure the application behaves as expected. This includes verifying navigation between screens, checking field validations (like password match and dropdown selections), and confirming that emails are triggered correctly for inquiry creation, follow-ups, cancellations, and quotations. Particular focus will be given to ensuring correct data handling in ALV grids and dropdown selections.

* **Customer Module Manual Testing:**

The Customer module will be manually tested for both new registrations and login functionality. Testers will verify that Customer ID auto-populates, password fields match, and security questions function as intended. After successful registration, testers will check login credentials and access to the customer dashboard. Test scenarios will include order selection, creating inquiries, sending follow-ups, cancellation requests, and accepting or rejecting quotations. Each button's functionality will be validated for proper email triggers and updates in the database. Negative test cases such as invalid login attempts and incomplete registration will also be performed.

* **Admin Module Manual Testing:**

For the Admin screen, manual testing will include checking dropdown filters for different order statuses and ensuring that order details are displayed hierarchically. Each order action — such as checking stock availability, sending quotations, fulfilling orders, and generating invoices — will be manually tested. The Smartform-based invoice generation will be validated for accuracy and formatting. The testing team will also verify the application's response to insufficient stock and test the functionality of sending revised quotations for cancelled orders. Testers will validate mail contents and attachments to ensure correct communication with customers.

* **Unit Testing Approach:**

Unit testing will be conducted using ABAP Unit Framework for each functional module — such as Customer Registration, Login Authentication, Order Processing, and Email Notifications. Developers will write test classes to mock database inputs and validate output logic, such as correct generation of Customer ID, validation checks for password fields, stock availability logic, and quotation generation. Boundary conditions, exception handling, and failure scenarios will be included to ensure robustness. Each data manipulation module, including database inserts and updates for orders and customer records, will be tested in isolation to guarantee correctness and reliability before integration.

* **Functional Testing of screens, workflows:**

Login and Registration Screen Functional Testing

Functional testing of the login and registration screens focuses on ensuring proper user access management and data integrity. Testers will verify that only valid users (customers and admin) can log in, with correct handling of incorrect usernames, passwords, and case sensitivity. The registration screen will be validated for correct population of the Customer ID, proper handling of mandatory fields, password matching, dropdown selections for security questions, and the successful saving of data into the customer table. Additionally, the "Customer Registration" hyperlink on the login screen must navigate users accurately to the registration form, confirming seamless user experience flow.

* **Customer Dashboard Functional Testing:**

In the customer screen, functional testing ensures that the ALV grid properly displays open orders specific to the logged-in user. Buttons for creating inquiries, sending follow-up or cancellation requests, and accepting or rejecting quotations will be tested for functionality and accuracy. Each action must reflect the correct changes in order status, trigger appropriate backend processes, and send notification emails. Testing will also ensure that only valid order selections are processed, with proper validation messages shown if no order is selected or if an invalid action is attempted. The integration with the standard SAP inquiry screen must also be tested for successful navigation and return flow.

* **Admin Screen Functional Testing:**

The admin screen will undergo functional testing for handling and processing customer orders. This includes verifying the functionality of the status dropdown filter and confirming that the hierarchical order list correctly reflects Open, Cancelled, and Completed orders. Actions such as checking stock availability, sending quotations with a pricing popup, fulfilling orders upon quotation acceptance, and triggering invoice generation through Smartforms will be functionally validated. The application’s ability to detect and notify users about stock shortages and partial fulfillment will be tested as part of the order fulfillment process.

* **Workflow and Integration Testing:**

End-to-end functional testing will validate the workflows connecting multiple screens and backend processes. A typical workflow starting from customer login, inquiry creation, quotation generation, acceptance, fulfillment, and invoice dispatch will be tested to ensure data consistency and logical transitions. Email integration at each critical point (inquiry, follow-up, cancellation, quotation, invoice) will be validated for correct recipients, content, and attachments. Moreover, workflows involving rejected or cancelled orders will be tested to ensure proper transitions and administrative actions. This holistic functional testing guarantees that the application meets business requirements and delivers a consistent user experience across roles.

SAP transaction usage: SE80, SE38, SE11, SE41, SWDD, SMARTFORMS

* **SE80 – Object Navigator**

SE80 is the central development environment in SAP, known as the Object Navigator. It allows developers to manage and access all components of a development project, including programs, function modules, classes, screens, and tables. For the smartSale Ltd. project, SE80 is heavily used to create and manage the module pool program, define screen layouts, and integrate custom logic into the transaction code. It serves as the main workspace where the entire application structure can be built and maintained.

* **SE38 – ABAP Editor**

SE38 is used to create, modify, and execute ABAP programs, particularly report and executable programs. In this project, SE38 can be utilized to write and test backend logic such as data retrieval, validations, and email triggers, either as standalone utilities or as support modules for the main application. It is also useful for writing test programs to validate functional components independently before integrating them into the main module pool.

* **SE11 – Data Dictionary**

SE11 is used for managing data dictionary objects such as tables, views, data elements, and domains. It plays a vital role in this project by allowing the creation and maintenance of custom tables like the Customer Table and Product/Material Table. All fields, data types, and relationships between database objects are defined here, forming the foundation for data storage and retrieval in the application.

* **SE41 – Menu Painter**

Creation and editing of the GUI menus in SAP applications is undertaken through SE41. Using the SE41 allows smartSale Ltd to realize the application menu arrangement in module pool programmes like top-level menus, push buttons and function keys, with their bespoke software. Consequently, users can seamlessly and hassle-free access such things as order management, starting inquiries, and logging out from the system.

* **SWDD – Workflow Builder**

The design and management of SAP Business Workflows are achieved through the SWDD, Workflow Builder. With this project, you are able to automate many order lifecycle events, like when generating emails when an order is completed or when an order is quoted. In order to manage approvals, updates, and communication through a controlled and record-able system, custom SWDD workflows can be created which send tasks between customers and administrators.

* **SMARTFORMS – Smart Form Builder**

SMARTFORMS is the transaction used to design and manage smart forms for SAP output, such as invoices or quotations. In this project, SMARTFORMS is utilized to generate the customer invoice after order fulfillment. It allows developers to create professional, print-ready documents with dynamic data integration, layout customization, and conditional formatting—all without needing SAPscript coding.

**4.2 Test Cases and Outcomes**

1. **Login validation (valid/invalid user)**

* **Test Case 1:** Valid Admin Login

Test Steps:

1. Open the custom SAP login screen.
2. Enter Username: MyAdmin
3. Enter Password: Admin@123
4. Click on the Login button.

**Expected Outcome**:

The system successfully validates the credentials and navigates to the Admin Dashboard. A welcome message or appropriate role-based interface is displayed.

* **Test Case 2:** Valid Customer Login

Test Steps:

1. Open the login screen.
2. Enter a valid customer Username and Password as saved in the Customer Table.
3. Click on the Login button.

**Expected Outcome:**

The system authenticates the user and redirects them to the Customer Screen showing their open orders in an ALV grid.

* **Test Case 3:** Invalid Username

Test Steps:

1. Enter a non-existent username.
2. Enter any password.
3. Click on the Login button.

**Expected Outcome:**

An error message is displayed stating "Invalid username or password." User is not granted access.

* **Test Case 4:** Invalid Password

Test Steps:

1. Enter a valid username (e.g., MyAdmin or an existing customer).
2. Enter an incorrect password.
3. Click on the Login button.

**Expected Outcome:**

Login fails with an error message: "Invalid username or password." User remains on the login screen.

* **Test Case 5:** Empty Fields

Test Steps:

1. Leave the username and/or password field empty.
2. Click on the Login button.

**Expected Outcome:**

The system displays a prompt such as "Please enter both Username and Password." No login attempt is made.

* **Test Case 6**: SQL Injection Attempt

Test Steps:

1. Input a SQL injection string (e.g., ' OR '1'='1) into the username or password field.
2. Click on the Login button.

**Expected Outcome:**

The system treats the input as plain text and rejects it. The login fails, and the system remains secure with no unauthorized access or errors.

1. **Registration field checks (empty fields, password mismatch)**

* **Test Case 1**: All Fields Valid

Test Steps:

1. Navigate to the Customer Registration screen.
2. Fill in all fields: Customer Name, Password, Re-enter Password (matching), select Security Question, and provide Answer.
3. Click on Register.

**Expected Outcome:**

Registration is successful. The data is saved to the Customer Table, and a success message is displayed. The user may be redirected to the login screen.

* **Test Case 2:** Empty Customer Name Field

Test Steps:

1. Leave the Customer Name field blank.
2. Fill in the remaining fields correctly.
3. Click on Register.

**Expected Outcome:**

System shows an error message such as "Customer Name is required." Registration is not processed.

* **Test Case 3**: Empty Password Fields

Test Steps:

1. Leave both Password and Re-enter Password fields empty.
2. Fill in the remaining fields correctly.
3. Click on Register.

**Expected Outcome:**

System prompts: "Password is required." Registration does not proceed.

* **Test Case 4:** Password Mismatch

Test Steps:

1. Enter a value in the Password field.
2. Enter a different value in the Re-enter Password field.
3. Fill in other fields correctly.
4. Click on Register.

**Expected Outcome:**

System displays an error: "Passwords do not match." The user is prompted to re-enter both fields.

* **Test Case 5:** Security Question Not Selected

Test Steps:

1. Leave the Security Question dropdown unselected or on the default value.
2. Fill in all other fields correctly.
3. Click on Register.

**Expected Outcome:**

System displays: "Please select a security question." Registration is halted.

* **Test Case 6**: Duplicate Customer Name (if restricted)

Test Steps:

1. Enter a customer name that already exists in the system.
2. Complete the rest of the fields correctly.
3. Click on Register.

**Expected Outcome:**

If customer names must be unique, an error like "Customer name already exists" should be shown. Otherwise, the registration proceeds.

1. **Inquiry creation and email trigger**

* **Test Case 1**: Successful Inquiry Creation and Email Trigger

Test Steps:

1. Login as a valid customer.
2. Navigate to the Customer Screen.
3. Select a valid order (or product if applicable).
4. Click on Create Inquiry.
5. Complete the required fields on the SAP Standard Inquiry screen.
6. Save the inquiry.

**Expected Outcome:**

Inquiry is successfully created in the SAP system.

A confirmation email is automatically sent to the admin team with relevant inquiry details. A success message is shown to the customer.

* **Test Case 2:** Inquiry Creation Without Order Selection

Test Steps:

1. Login as a valid customer.
2. Do not select any order.
3. Click on Create Inquiry.

**Expected Outcome:**

System displays a message: "Please select an order before creating an inquiry." No inquiry is created, and no email is triggered.

* **Test Case 3:** Email Server or Configuration Error During Trigger

Test Steps:

1. Simulate or force a failure in the email configuration (e.g., disconnect mail server or use invalid SMTP settings).
2. Attempt to create an inquiry.

**Expected Outcome:**

Inquiry may still be created (if handled separately), but an error message like "Email could not be sent. Please contact support" is shown.

A log or retry mechanism may be triggered depending on system design.

* **Test Case 4**: Inquiry Creation With Missing Mandatory Fields (Standard Screen)

Test Steps:

1. Login as a customer and navigate to the inquiry screen.
2. Leave mandatory fields (e.g., material or quantity) blank.
3. Click Save.

**Expected Outcome:**

Standard SAP validation prevents saving. Error messages such as "Material is required" appear. Inquiry is not created, and no email is sent.

* **Test Case 5:** Multiple Inquiries in a Session

Test Steps:

1. Login as a customer.
2. Create multiple inquiries consecutively.
3. Monitor email triggers for each.

**Expected Outcome:**

Each successful inquiry creation results in a separate email sent to the admin team with accurate details. System handles multiple actions without delay or data overlap.

* **Test Case 6:** Admin Email ID Not Maintained

Test Steps:

1. Remove or simulate absence of admin email in the configuration.
2. Attempt to create an inquiry.

**Expected Outcome:**

System shows a message such as "Admin email not found. Email not sent."

Inquiry creation may still succeed, but email trigger fails gracefully with proper logging.

1. **Admin actions for quotation, stock validation**

* **Test Case 1:** Successful Stock Validation for an Order

Test Steps:

1. Login as Admin.
2. Navigate to the "Open Orders" section.
3. Select an order containing one or more products.
4. Click on Check Stock.

**Expected Outcome:**

The system checks available stock for each product in the order. If all products are in stock, a message like "Stock available for all products" is shown.

* **Test Case 2:** Stock Shortage for One or More Products

Test Steps:

1. Login as Admin.
2. Select an open order with at least one product quantity greater than the stock available.
3. Click on Check Stock.

**Expected Outcome:**

The system displays which products are short in stock (e.g., “Product X: Only 5 available, 10 requested”). Admin is prevented from proceeding to fulfillment without updating stock or revising the order.

* **Test Case 3:** Create Quotation for Valid Order

Test Steps:

1. Login as Admin.
2. Go to "Open Orders", select an order with valid product and quantity.
3. Click on Send Quotation.
4. Enter prices in the popup and submit.

**Expected Outcome:**

Quotation is created successfully. An email is triggered to the respective customer with quotation details or an attachment. System shows success message.

**CHAPTER 5: RESULTS AND EVALUATION**

**5.1 RESULTS**

The implementation of the custom SAP Sales & Distribution application for smartSale Ltd. was successfully completed, meeting all the defined functional and technical requirements. The module pool program, built using ABAP and custom DDIC objects, functioned as expected across all user roles and processes.

All major functional screens, including login, registration, customer dashboard, admin interface, and profile update (Screen 500), were fully developed and tested. Navigation between screens was smooth, and user validations were handled correctly at each step. Role-based redirection ensured customers and admins accessed the appropriate dashboards with relevant features.

A total of five key events—Inquiry Created, Follow-Up Requested, Order Canceled, Quotation Accepted, and Quotation Rejected—were successfully integrated with SAP workflows. These events triggered automated email notifications using CL\_BCS, enabling real-time communication between customers and the admin team.

Additionally, ALV Grids played a central role in the application’s UI. Features like row selection, sorting, and filtering were implemented and functioned accurately, supporting operations such as bulk follow-ups, cancellations, and quotation approvals. All selection-based logic worked reliably under various test cases, contributing to a smooth and intuitive user experience.

The following screenshots provide evidence of the working solution and demonstrate key outputs of the system.

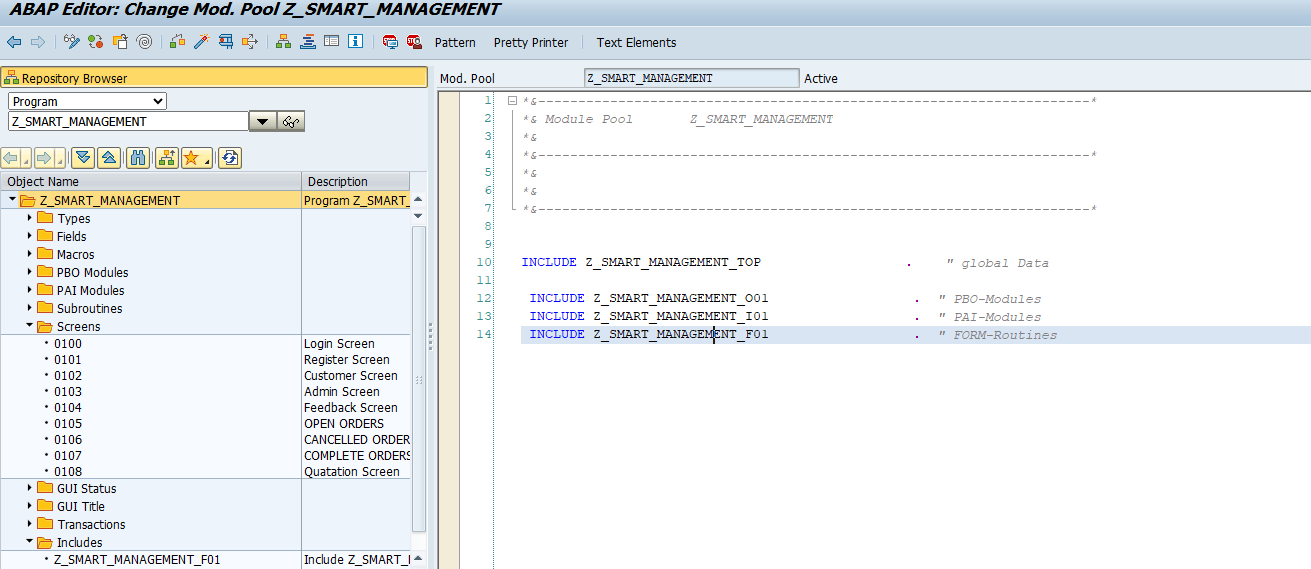
****

Figure 5.1.1 Module Pool Editor Screen

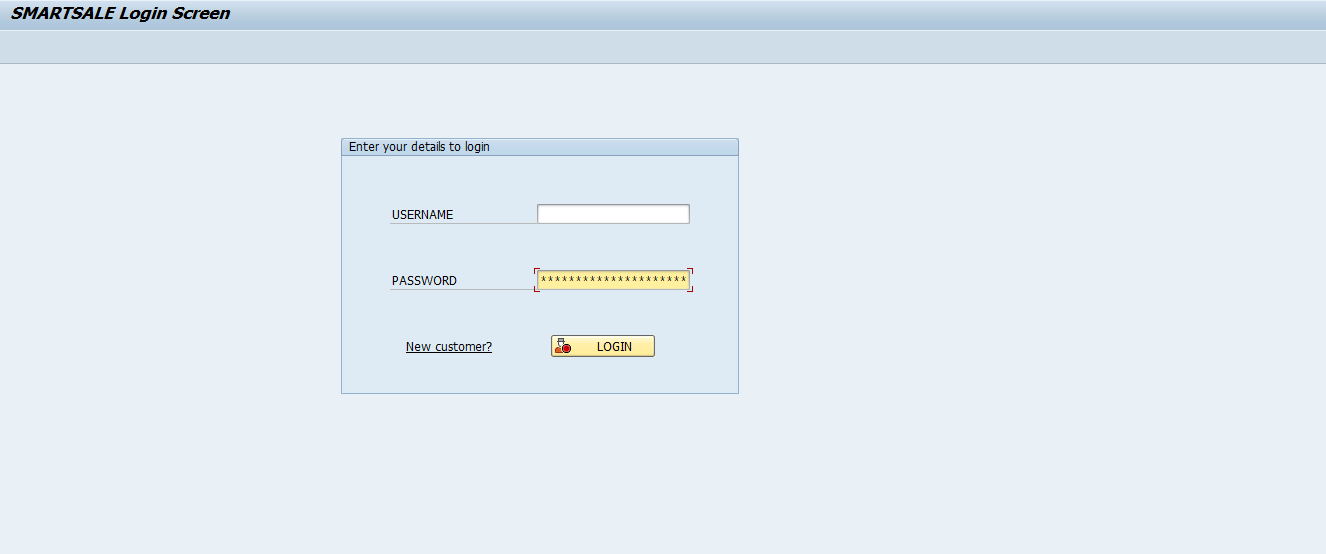


Figure 5.1.2 Smartsale Login Screen

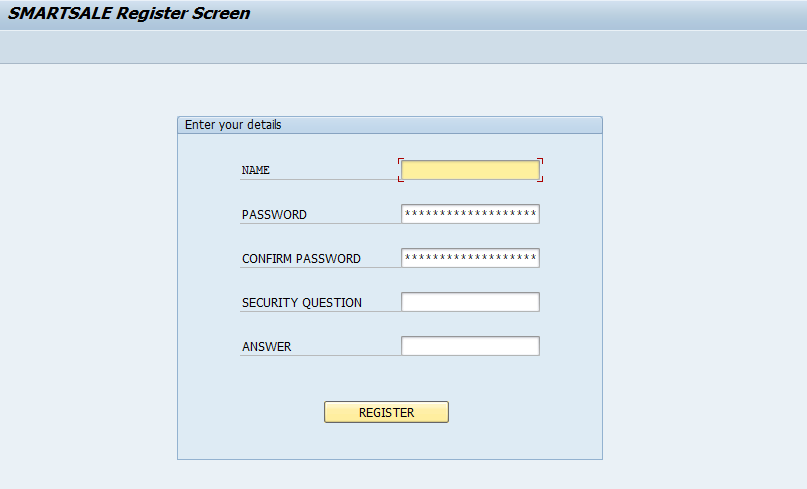
****

Figure 5.1.3 Register Screen

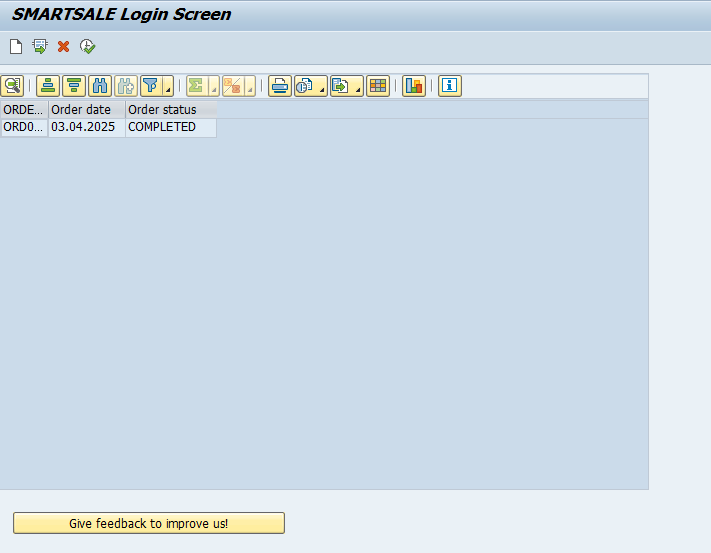
****

Figure 5.1.4 Customer Screen showing orders.

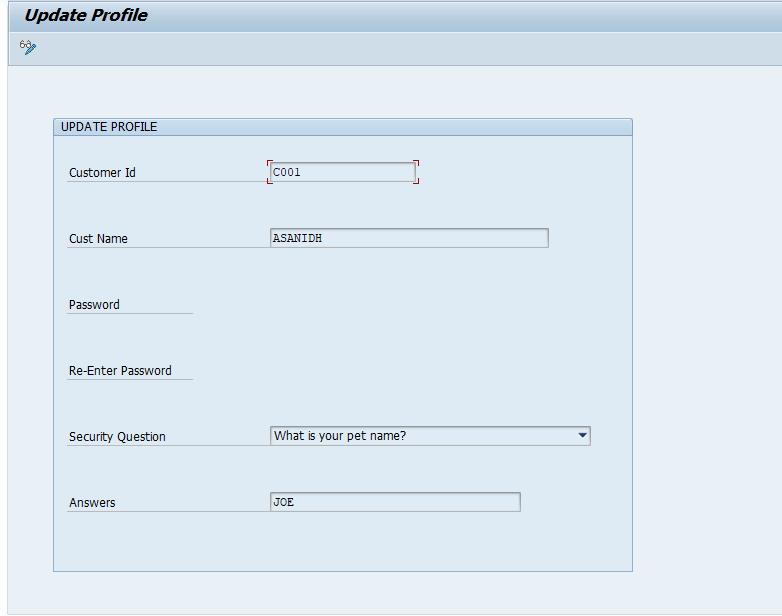
****

Figure 5.1.5 Update Profile Screen.

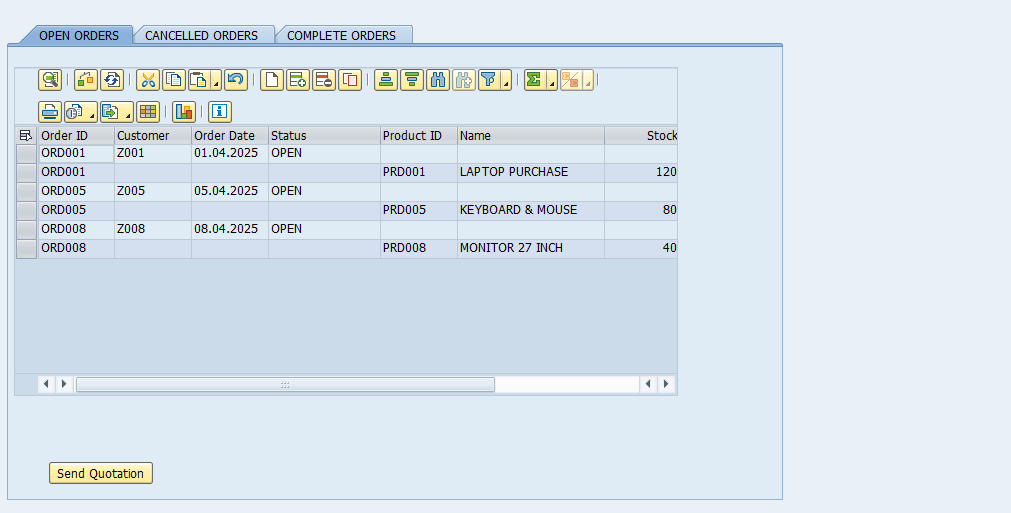
****

Figure 5.1.6 Admin Screen with Open orders

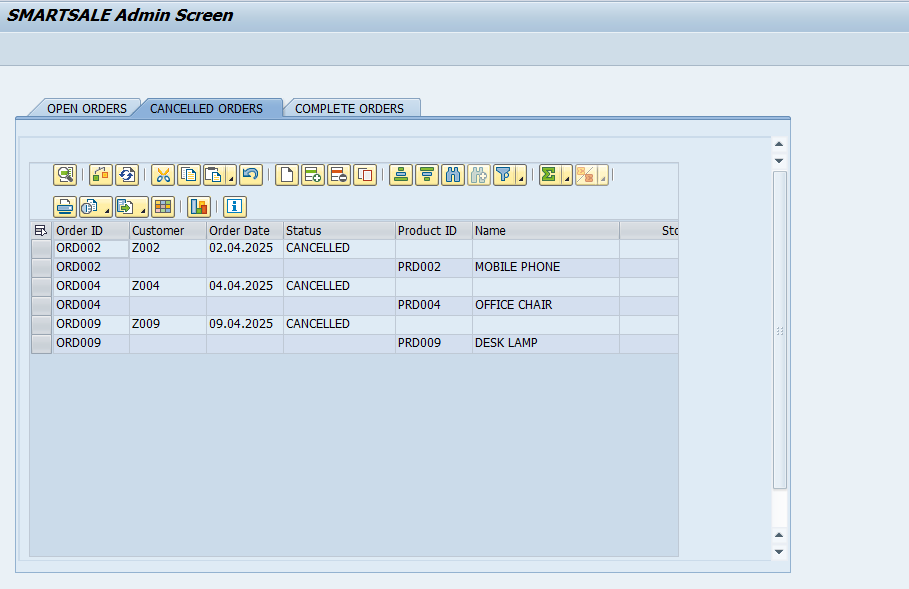
****

Figure 5.1.7 Admin Screen with Cancelled Orders

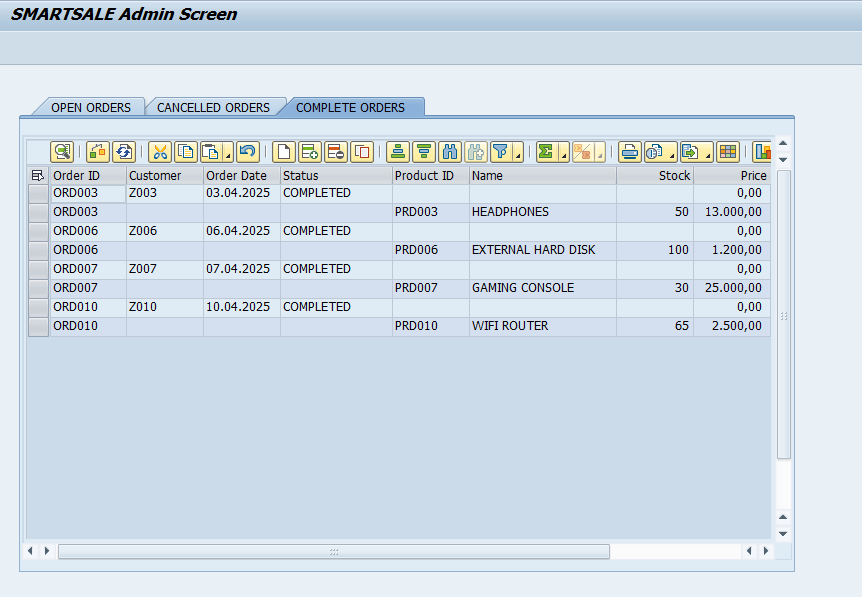
****

Figure 5.1.8 Admin Screen with Completed Orders

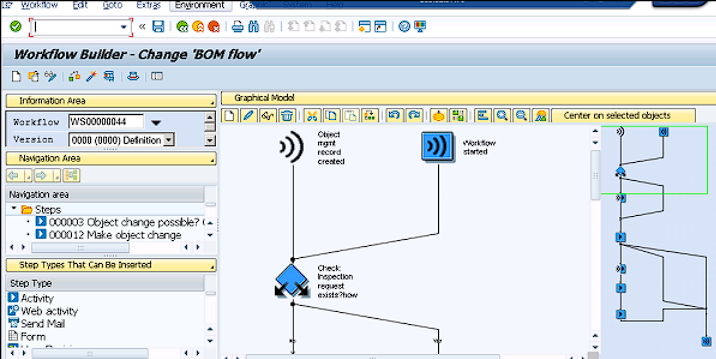


Figure 5.1.9 Workflows using T-code SWDD

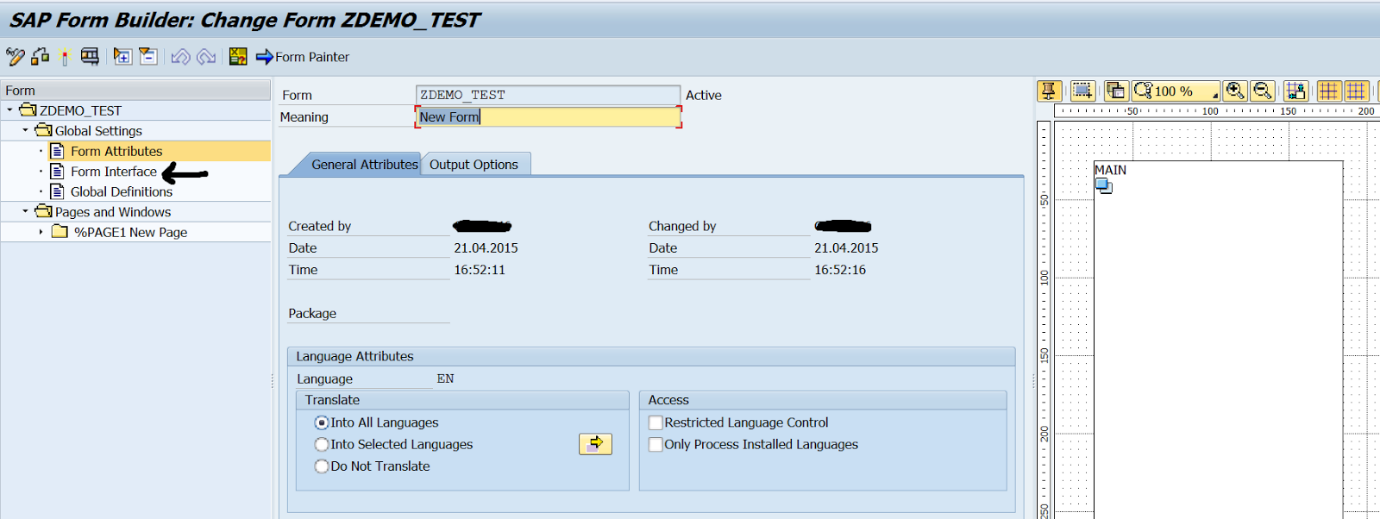


Figure 5.1.10 Smart forms Creation

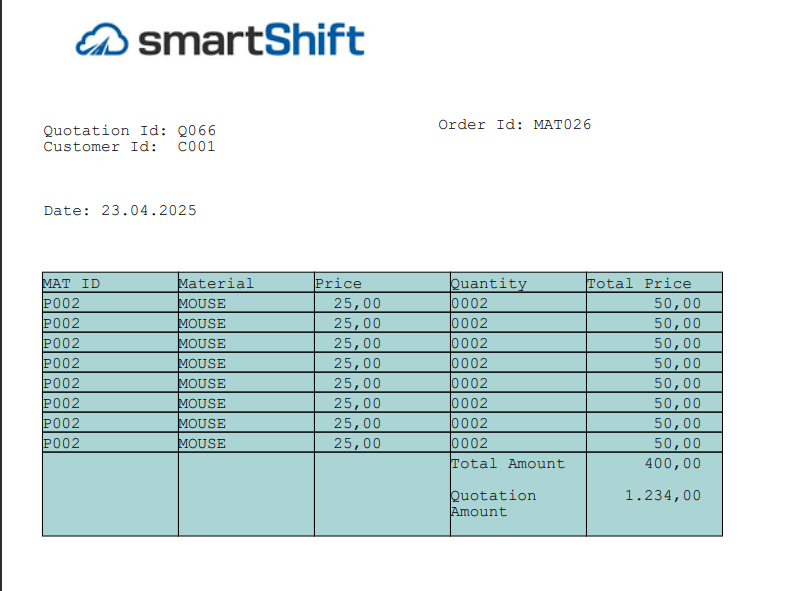


Figure 5.1.11 Smart forms Quotation Details

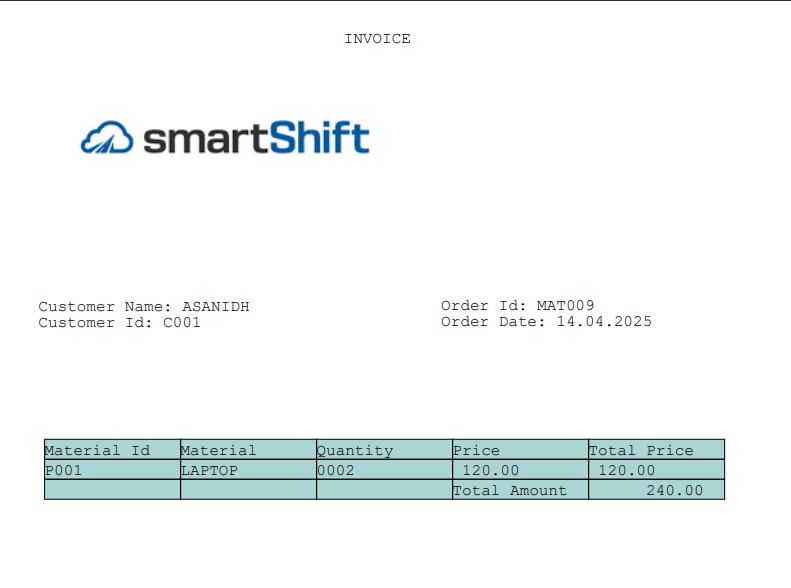
****

Figure 5.1.12 Smart forms Invoice Detail

**CHAPTER 6: CONCLUSION AND FUTURE SCOPE**

**6.1 CONCLUSION**

The implementation of the custom SAP Sales & Distribution application for SmartSale Ltd. marks a significant step toward the digital transformation of the company's sales operations. This project successfully automated key business activities such as customer onboarding, order tracking, inquiry creation, quotation handling, and invoice generation. By integrating SAP’s standard capabilities with custom module pool programming, the application offers a user-centric and process-driven platform for managing the entire order lifecycle.

The use of ABAP programming, ALV Grids, Smartforms, and CL\_BCS email functionality made it possible to create a responsive and reliable system that aligns well with the business needs of both customers and administrators. The clear segregation of user roles, dynamic screen transitions, and workflow-based email notifications contributed to a smooth user experience and operational efficiency. Additionally, features like the customer profile update screen (Screen 500) provided flexibility and self-service capabilities, further enhancing user satisfaction.

From a development perspective, modular coding and reusable function modules ensured better maintainability and scalability. Business validations, secure login mechanisms, and real-time data handling were successfully implemented using SAP best practices. Smartforms allowed the creation of professional and automated documents, contributing to the credibility and professionalism of customer communication.

Moreover, the project helped the team gain hands-on experience with real-time SAP scenarios, improving technical and problem-solving skills. It demonstrated the effectiveness of integrating standard SAP tools with custom logic to address specific organizational needs, setting the stage for future projects of greater complexity and scale.

**Limitations**

While the application fulfills the core requirements effectively, there are a few limitations that were identified during development and testing:

* **Email Dependency on SAPconnect**: The email functionality relies on the SAPconnect setup, which can be restrictive in certain testing or production environments without proper SMTP configuration. Any issue in mail server connectivity could delay workflow notifications.
* **Hardcoded Admin Credentials:** For demonstration purposes, admin login credentials were hardcoded. In a production system, a secure and dynamic user authentication system using roles and authorization objects would be more appropriate.
* **Limited Error Handling for User Inputs:** Although basic validations are implemented across screens, certain edge cases (like special characters or excessive input lengths) may not be fully handled in all scenarios. Enhanced field-level validations can be introduced for better data quality.
* **Scalability for Large Data Sets**: The current implementation is designed for functional validation and small data volumes. For enterprise-level deployments, further optimization of data loading, screen responsiveness, and memory management will be necessary.

**6.2 FUTURE SCOPE**

While the current implementation of the custom SAP Sales & Distribution application effectively addresses SmartSale Ltd.'s immediate needs, there are several opportunities for future enhancement and expansion of the system. These improvements can further elevate the functionality, usability, and scalability of the solution.

One major area of enhancement is the integration of SAP Fiori or SAP UI5 for a more modern, web-based interface. This would allow the application to be accessible across multiple devices, including tablets and smartphones, providing a responsive and user-friendly experience beyond the SAP GUI. Implementing Fiori elements for customer dashboards and admin panels can also improve visual clarity and ease of use.

Another future enhancement is the implementation of role-based authorization and security management. Currently, the application uses a hardcoded admin login. In a production environment, integrating SAP’s standard authorization objects would ensure secure access control, allowing user-specific views and actions based on predefined roles.

In terms of communication, the email functionality can be extended to include SMS or push notifications using SAP Cloud Integration or third-party APIs. This would improve real-time interaction and keep both customers and admins updated even when they’re not actively logged into the system.

The reporting and analytics capabilities of the system can also be improved. Incorporating SAP’s reporting tools or integrating with SAP BW/BI systems could provide detailed sales insights, product movement trends, and customer behavior analysis—helping management make informed business decisions.

Lastly, for further automation and business process optimization, integration with SAP Workflow or SAP Business Process Management (BPM) tools can be explored. These would allow the definition of flexible business rules and escalations, especially for quotation approvals, delayed orders, or stock shortages.

By pursuing these future enhancements, SmartSale Ltd. can transform this foundational system into a comprehensive digital solution that not only supports current operations but also adapts to future business growth and technological advancements.

**REFERENCES**

**[1]** SAP SE (2022). *Smart Forms in SAP – Creating and Sending Documents*. SAP Documentation. Available online: <https://help.sap.com/viewer/product/SMARTFORMS>

**[2]** B. R. Khan (2022). "Integrating SmartForms with Business Processes in SAP," *International SAP Solutions Journal*, vol. 6, no. 4, pp. 18–23.

**[3]** M. D. Srinivas (2022). "Designing Role-Based Login and Authentication Modules in SAP," *International Journal of SAP Technologies*, vol. 8, no. 1, pp. 34–39.

**[4]** H. Frank and T. Allgaier, SAP S/4HANA Sales Certification Guide (C\_TS462\_2021): Application and Configuration Preparation, SAP Press, 2021

**[5]** N. Singh, "Manual and Unit Testing Strategies in SAP-Based Projects," *Journal of Enterprise Software Testing*, vol. 7, no. 2, pp. 50–55, 2021.

**[6]** M. T. Smith, “Designing SAP Fiori UIs for SAP GUI Module Pool Applications: A Transition from Classical Dynpro to Modern UX,” in SAP UI/UX Developer Summit, Palo Alto, CA, USA, Sep. 2020.

**[7]** D. Obermeier, Efficient SAP Business Workflow: Practical Solutions for Workflow Automation, SAP Press, Rheinwerk Publishing, 2020.

**[8]** V. N. Kumar and S. Rathi, “Improving Customer Registration and Authentication in SAP ERP Using Secure ABAP Techniques,” International Journal of Engineering Research and Applications (IJERA), vol. 10, no. 2, pp. 51–56, Feb. 2020

**[9]** B. T. Robinson . "SAP Email Integration via ABAP and SMTP Configuration," *International Journal of SAP Practices*, vol. 5, no. 1, pp. 12–17, 2020.

**[10]** A. Bose . *Custom SAP Applications: Real-World Development Using ABAP*, SAP Press, Boston, MA. 2020.

**[11]** C. C. Smith, Customizing SAP S/4HANA: Tools and Techniques for Tailoring the System to Meet Business Requirements, SAP Press, Boston, MA, USA, 2020.

**[12]** R. Deshmukh . "ABAP-Based Email Automation Using SAP Workflows," *SAP Workflow Insights*, vol. 7, no. 4, pp. 10–15, 2020.

**[13]** V. Mohan, “Email Notifications in SAP ERP Systems: Enhancing Customer Communication in SD Module Through SMTP Integration,” in Proc. International Conference on Communications and Signal Processing (ICCSP), Chennai, India, Apr. 2020, pp. 338–342.

**[14]** G. Fritsche . *ABAP Objects: ABAP Programming in SAP NetWeaver and SAP S/4HANA Environments*, SAP Press, Boston, MA. 2019.

**[15]** S. K. Jain and D. Bhattacharya, “Risk Assessment and Mitigation in SAP ERP SD Module Implementation Projects,” International Journal of Management and Applied Science (IJMAS), vol. 5, no. 3, pp. 89–94, Mar. 2019.

**[16]** S. B. Pandey and M. P. Gupta, “Digital Transformation of Sales and Distribution Systems Using SAP ERP in Mid-Sized Firms,” Journal of Enterprise Information Management, vol. 32, no. 5, pp. 807–825, 2019.

**[17]** B. Gupta and K. Varma, “Implementation of SAP ERP Order to Cash (O2C) Cycle: A Case Study in Automobile Sector,” International Journal of Engineering and Advanced Technology (IJEAT), vol. 8, no. 5, pp. 1453–1458, Jun. 2019.

**[18]** S. Sharma and R. Pandey, “Custom Module Pool Programming for Order Management: SAP ABAP Approach,” International Journal of Scientific and Research Publications (IJSRP), vol. 8, no. 11, pp. 565–570, Nov. 2018.

**[19]** K. L. Rajput and S. Jaiswal, “Optimizing Sales Order Processing Through SAP ERP in Manufacturing Enterprises,” International Journal of Business Information Systems, vol. 30, no. 2, pp. 145–159, 2018.

**[20]** F. Pech, SAP ABAP Advanced Cookbook: Real-World Techniques and Best Practices for Modern SAP Developers, Packt Publishing, Birmingham, UK, 2018.

**[21]** A. Menon and R. Kumar, “Smart Data Entry Using SAP GUI Dynpro Programming: Best Practices for User Interaction,” in Proc. International Conference on Software Engineering and Technology (ICSET), Bengaluru, India, Aug. 2017, pp. 212–217.

**[22]** P. L. Parker, “Smart Forms and Output Management in SAP SD Module: A Comprehensive Approach,” in Proc. SAP TechEd Conference, Las Vegas, NV, USA, 2017, pp. 75–82.

**[23]** M. Dole . *SAP Testing: A Practical Guide for Functional, Regression, and Integration Testing*, SAP Press, Boston, MA. 2016.

**[24]** M. Prickett, ABAP Development for SAP Business Workflow: User Decisions, Emails, and Workflow Templates, SAP Press, Boston, MA, USA, 2016.

**[25]** P. Bhagat, “Design and Implementation of Inventory Control Using SAP ERP: A Case Study on Material Management,” in Proc. International Conference on Advances in Science and Engineering (ICASE), Pune, India, May 2016, pp. 236–241.

**[26]** A. P. Kulkarni, “Enterprise Application Integration Using SAP NetWeaver: Techniques for Developing Custom ABAP Interfaces and Real-Time Process Integration,” SAP Press, Rheinwerk Publishing, Boston, MA, 2015.