Journal of **N**onlinear **A**nalysis and **O**ptimization
Vol. 16, Issue. 1: 2025
ISSN : **1906-9685**

CENTRAL DATABASE DESIGN

Abinash Behera
Biswajit Swain
Email ID: babinash2023@gift.edu.in , bwswain2023@gift.edu.in
Guided By: Er. Jagannath Ray Assistant Professor, Department of MCA, Gandhi Institute for Technology, BPUT, India

Abstract- The project titled "Central Database Design" using SAP presents the development of a customized SAP ABAP application to meet a specific business need not adequately addressed by standard SAP modules. It encompasses the entire software development lifecycle—starting with requirement gathering and analysis, followed by detailed design, coding, testing, and final deployment within the SAP environment. The application is primarily focused on effective database management through Open SQL operations, leveraging ABAP's internal tables for data handling, and producing structured reports using classical and ALV reporting techniques.

Keywords-ABAP, SAP ABAP Editor(SE38), Data Dictionary (SE11), SAP GUI (Graphical User Interface), SAP ABAP WORK BENCH(SE80).

I. INTRODUCTION

This project presents the design and implementation of a robust SAP ABAP (Advanced Business Application Programming) solution tailored to streamline a specific business process within the SAP ERP environment. As a core component of the SAP NetWeaver platform, ABAP is instrumental in extending and customizing standard SAP functionalities to align with unique organizational requirements. The solution encompasses the development of custom reports, data entry interfaces, database operations, and seamless integration with SAP modules using Open SQL and modular programming techniques. Key ABAP Workbench tools—including the ABAP Editor, Function Builder, and Data Dictionary—are extensively utilized to define the application's logic and structure. Furthermore, the project incorporates classical and interactive reporting techniques, internal tables, and the ABAP List Viewer (ALV) to enhance data visualization and user interaction. Emphasis is also placed on performance optimization to ensure efficient data processing. The report highlights ABAP's critical role in enterprise application development, emphasizing its integration capabilities withcore SAP modules such as Materials Management (MM), Sales and Distribution (SD), and Financial Accounting (FI).

Purpose : This project is centered on the development of a custom SAP ABAP application designed to meet specific business requirements that standard SAP functionalities cannot address. It showcases the use of ABAP to create tailored reports, forms, and data processing logic integrated with core SAP modules. Leveraging Open SQL for efficient database operations and employing modularization techniques like function modules and subroutines, the solution promotes code reuse and maintainability. The project also incorporates user-friendly interfaces using classical reports and ALV grids, while emphasizing adherence to SAP coding standards, performance optimization, and proper transport management across development, testing, and production environments—effectively preparing developers for real-world SAP implementation scenarios.

A. Scope

The scope of this project is limited to the development and implementation of a customized SAP ABAP application focused on essential data operations and report generation for a specific business process within the SAP ECC environment. It includes designing database structures using the Data Dictionary, developing ABAP programs for data retrieval and manipulation, creating user interfaces through Dynpro or selection screens, and integrating with standard SAP modules. The project emphasizes classical ABAP development, utilizing techniques such as ALV Grid and Classical Reports for data presentation, along with backend logic for validations, exception handling, and user- friendly messaging. While advanced technologies like SAP Fiori, Web Dynpro, or S/4HANA-specific features are excluded, the project lays a strong foundation for future enhancements, including potential integration with additional SAP modules such as HR, Finance, or CRM.

B. Literature Survey

SAP ABAP is a core programming language used to customize and extend SAP ERP functionalities. This literature survey highlights its importance through official documentation, academic studies, and community resources. ABAP enables the development of RICEF objects using tools like the ABAP Workbench and Data Dictionary. Studies show its effectiveness in automating processes, enhancing reports, and improving data accuracy. Key development practices include efficient Open SQL usage, performance optimization, modular programming, and adherence to SAP standards. Overall, the literature confirms ABAP's essential role in building scalable, business-specific applications within the SAP ecosystem. The support vector machine (SVM) is a machine learning algorithm that determines boundaries between data points based on predefined classes, labels, or outputs. The SVM algorithm's main goal, technically, is to locate a hyperplane that clearly divides the data points into separate classes.

Method

The methodology adopted for this SAP ABAP development project follows a structured and phased approach , covering the entire lifecycle of software development—starting from requirement analysis to deployment and testing within the SAP environment. Each phase is carefully designed to align with SAP development standards and ABAP best practices to ensure a robust and maintainable application.

Database Analyzing Design and implementation In SAP ABAP development, the design and implementation of the database layer are critical for ensuring data accuracy, integrity, and performance. The process begins with **database requirement analysis**, which involves identifying data types, relationships, constraints, and expected data volume. Based on these requirements, the **SAP Data Dictionary (SE11)** is used to define key database components such as **domains**, **data elements**, **transparent tables**, and **views**. Transparent tables are implemented with clearly defined **primary keys**, **foreign key constraints**, and appropriate **field types** to maintain referential integrity and performance. Secondary indexes may be added where necessary to optimize access speed. The use of **check tables**, **value tables**, and **F4 help** ensures robust data validation and user-friendly input options. Once the data structures are in place, they are integrated with ABAP programs using **Open SQL** statements for data operations like SELECT, INSERT, UPDATE, and DELETE. Internal tables and work areas are used within ABAP to manage data before interacting with the database, enabling efficient processing and ensuring data integrity throughout the application lifecycle.

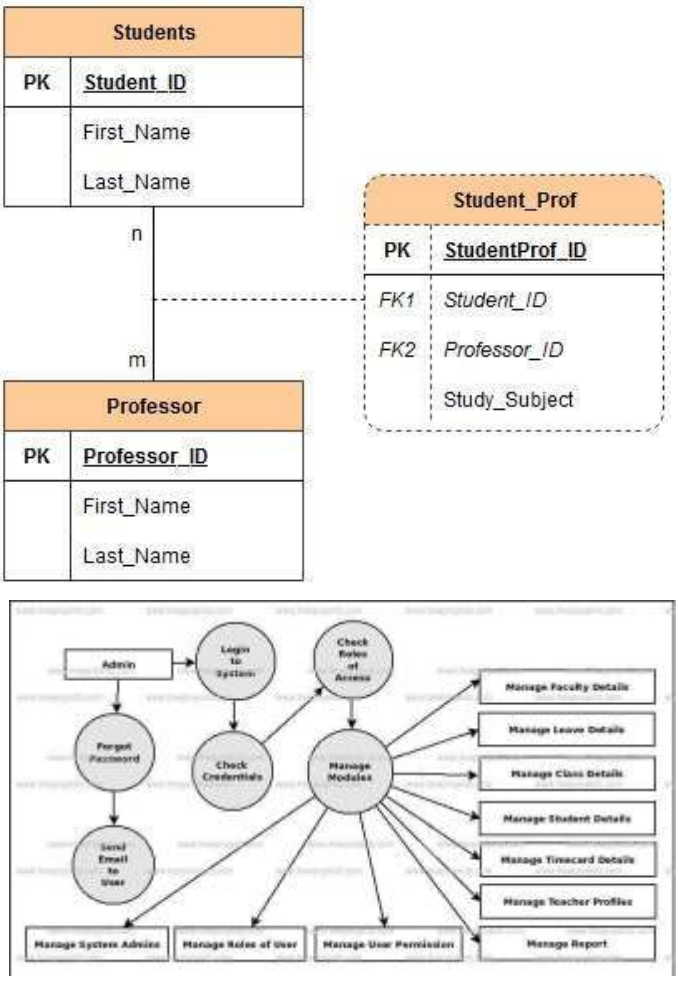


Figure 1: Proposed work

Figure 1 shows the proposed model. Where the first step is Data Extraction, Collecting the dataset for analysis and comparison from multiple sources, the second step is Data Transformation. Data Transformation is a process of removing all the outliers, missing values, converting and structuring into usable format. The third step is Algorithm Selection where different algorithms are selected and used in model building. In the fourth step models are trained from the past data. In the next step, performance is analyzed and the accuracy of the algorithms is compared and find the best algorithm based on accuracy. At last, it will predict whether a person has heart disease or not based on the given input.

II. METHODOLOGY

A. Requirement Gathering Analysis : The process begins with an in-depth requirement analysis phase. In collaboration with stakeholders, including end-users and system analysts, functional and non-functional requirements are identified. This includes determining what data needs to be stored, how it will be accessed, what operations will be performed, and the expected outcomes.

B. System Design : The Based on the requirements, a comprehensive system design is prepared. This includes:

- **Database Design**: Defining the structure of custom tables (Z-tables) and their relationships. Primary keys, foreign keys, and indexing strategies are finalized.
- **Program Design**: The application logic is designed using modular principles. Reusable function modules, classes, and methods are planned.
- **Interface Design**: User interaction is planned via classical reports, ALV (ABAP List Viewer) grids, and possibly custom screens using Dynpro or Web Dynpro.

C. Development and Implementation : The During this phase, the actual development of the SAP ABAP application takes place. Key tasks include:

- **Table Creation**: Custom Z-tables are created in the Data Dictionary (SE11).
- **Module Pool Programming**: Custom screens and input logic are developed for interactive applications.
- **Report Programming**: ABAP reports are written using SELECT, INSERT, UPDATE, and DELETE statements, making use of internal tables and work areas.
- **Form and Function Modules**: Reusable pieces of code are encapsulated into function modules and forms, which helps in modularizing the development.

D. Testing : Testing ensures that the developed application meets user expectations and functions correctly. Various types of testing are carried out such as unit testing , integration testing and user acceptance testing.

E. Deployment : After thorough testing and approval, the application is transported from the development system to the quality assurance (QA) system, and finally to the production system using SAP's transport management system (TMS). All transport requests are tracked to maintain version control and auditing.

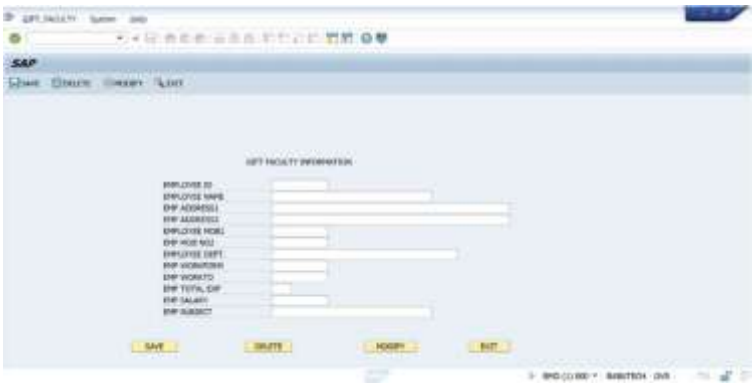
F. Documentation and Training : Comprehensive documentation is prepared, including technical specifications, user manuals, and developer notes. Additionally, training sessions are conducted for end-users to ensure a smooth transition and adoption of the new system.

G. Comparison of various machine learning algorithms Post-deployment, the system is monitored for performance and user feedback is gathered. Any bugs or enhancements are addressed in subsequent cycles. This feedback loop ensures continuous improvement and alignment with evolving business needs.

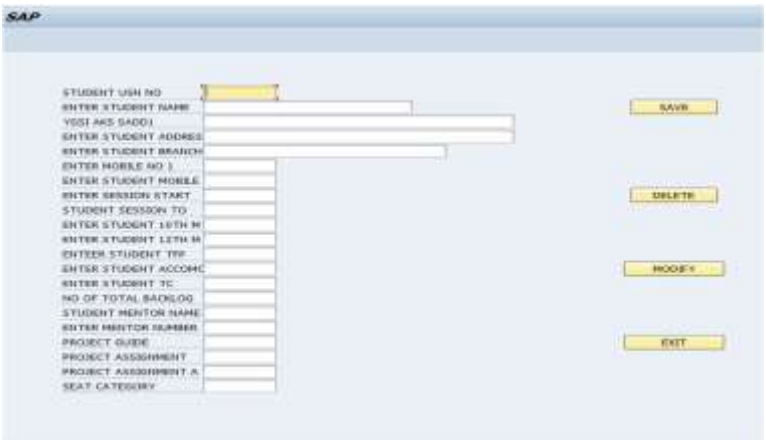
III. RESULTS

The SAP ABAP development project aimed to design and implement a custom database-driven application to efficiently manage business data operations such as data retrieval, insertion, modification, and deletion. After the successful completion of the implementation and testing phases, the results were analyzed to determine the effectiveness and performance of the system.

EMPLOYEE TABLE



STUDENT TABLE



SAP Data Dictionary objects (Z-tables), and interactive user interfaces, resulting in a user-friendly and reliable solution.

FUTURE SCOPE

While the current implementation fulfills all fundamental requirements, several future enhancements can be introduced to improve the system's efficiency, scalability, and user experience. Transitioning from classic SAP GUI to modern UI technologies such as SAP Fiori or SAPUI5 would provide a responsive, mobile-friendly, and intuitive interface. Integration with SAP Workflow could further streamline and automate business processes. Incorporating advanced analytics tools like SAP BusinessObjects or SAP Analytics Cloud would enable more insightful data visualization and support data-driven decision-making. Additionally, implementing robust security measures, such as role-based access control (RBAC) through SAP authorization objects, would help safeguard sensitive operations by ensuring only authorized users can access or modify critical data.

REFERENCES

SAP ABAP development project. Throughout the analysis, design, and implementation phases, the following references were consulted to ensure best practices, adherence to SAP standards, and effective problem-solving:

1. JSR-74. *SAP_ABAP_JSR*. GitHub, 2023, https://github.com/jsr-74/SAP_ABAP_JSR.
This GitHub repository showcases a student and faculty management system built using SAP ABAP. It includes modules for user login, data display using ALV grids, and various SAP standard T-codes integrated into the workflow.

2. "SAP ABAP Tutorials and Guide." *eLearning Solutions*, www.elearningsolutions.co.in/tutorials-on-sap-abap-enjoy-reading/.
A comprehensive guide covering SAP ABAP fundamentals, including ALV reporting, data dictionary objects, and custom transaction code usage, essential for developing structured applications like the Central Database Design.

3. "ALV Reports in SAP – ABAP List Viewer." *Guru99*, www.guru99.com/alv-list-view-programming.html.
This tutorial introduces the ABAP List Viewer (ALV), a powerful tool for creating dynamic and interactive reports in SAP

4. "SAP ABAP Corporate Training Course." *Edstellar*, www.edstellar.com/course/sap-abap-training.
A professional training course that offers practical knowledge of ABAP programming, including internal tables, T-codes. and ALV reporting techniques that are relevant for enterprise-level applications like student-faculty systems.

5. Reddy, Sree Harsha. *ALV Reports Using Function Modules in SAP ABAP (Beginners)*. Udemy, https://www.udemy.com/course/reports-in-abap/.
This beginner-friendly Udemy course focuses on building ALV reports using ABAP function modules. It provides step-by-step examples helpful for implementing report-driven interfaces in SAP-based applications.