****

**COMSATS University Islamabad**

**Abbottabad, Pakistan**

**An ERP System**

***By***

**Abdullah Khan CIIT/SP20-BSE-042/ATD**

**Syed Muhammad Saqib CIIT/SP20-BSE-050/ATD**

**Shameer Mukhtar CIIT/SP20-BSE-076/ATD**

***Supervisor*Dr. Kashif Nasr**

***Bachelor of Science in Software Engineering (2020-2024)***

**The candidate confirms that the work submitted is their own and appropriate  
 credit has been given where reference has been made to the work of others**.

****

**COMSATS University, Islamabad Pakistan**

**An ERP System**

**A project presented to**

**COMSATS Institute of Information Technology, Islamabad**

**In partial fulfillment**

**of the requirement for the degree of**

***Bachelor of Science in Software Engineering (2020-2024)***

**By**

**Abdullah Khan CIIT/SP20-BSE-042/ATD**

**Syed Muhammad Saqib CIIT/SP20-BSE-050/ATD**

**Shameer Mukhtar CIIT/SP20-BSE-076/ATD**

**DECLARATION**

We hereby declare that this software, neither whole nor as a part has been copied out from any source. It is further declared that we have developed this software and accompanied report entirely on the basis of our personal efforts. If any part of this project is proved to be copied out from any source or found to be reproduction of some other. We will stand by the consequences. No Portion of the work presented has been submitted of any application for any other degree or qualification of this or any other university or institute of learning.

Abdullah Khan Syed M. Saqib Shameer Mukhtar

--------------------------- --------------------------- ----------------------------

**CERTIFICATE OF APPROVAL**

It is to certify that the final year project of BS (SE) “An ERP System” was developed by   
**ABDULLAH KHAN (CIIT/SP20-BSE-042/ATD)** and **SYED M. SAQIB (CIIT/SP20-BSE-050/ATD)** and **SHAMEER MUKHTAR (CIIT/SP20-BSE-076/ATD)** under the supervision of “DR. KASHIF NASR” and that in his opinion; it is fully adequate, in scope and quality for the degree of Bachelors of Science in Computer Sciences.

---------------------------------------

**Supervisor**

---------------------------------------

**External Examiner**

---------------------------------------

**Head of Department**

**(Department of Computer Science)**

**EXECUTIVE SUMMARY**

The ERP System, as detailed in this thesis, represents a groundbreaking venture into the realm of Enterprise Resource Planning tailored explicitly for small-scale suppliers. In a world where ERP solutions predominantly cater to large corporations, this project addresses a glaring gap by providing an ERP system finely tuned to the unique challenges and aspirations of small-scale suppliers. The ERP System comprises modules for finance management, customer relationship management (CRM), human resource management (HRM), and inventory management, each meticulously designed to enhance operational efficiency and empower data-driven decision-making.

By leveraging a client-server architecture and the MERN stack, we've ensured accessibility, scalability, and global reach. The system's architecture promotes modularity and flexibility, while design models like component-based design and user-centered design principles prioritize user-friendliness and adaptability. Furthermore, our document-oriented data modeling approach aligns perfectly with the database system, MongoDB, ensuring efficient data handling.

**ACKNOWLEDGEMENT**

All praise is to Almighty Allah who bestowed upon us a minute portion of His boundless knowledge by virtue of which we were able to accomplish this challenging task.

We are greatly indebted to our project supervisor “Dr. Kashif Nasr”. Without their personal supervision, advice and valuable guidance, completion of this project would have been doubtful. We are deeply indebted to them for their encouragement and continual help during this work.

And we are also thankful to our parents and family who have been a constant source of encouragement for us and brought us the values of honesty & hard work.

Abdullah Khan Syed M. Saqib Shameer Mukhtar

--------------------------- --------------------------- ----------------------------

**ABBREVIATIONS**

|  |  |
| --- | --- |
| **SRS** | Software Require Specification |
| **PC** | Personal Computer |
| **SMEs** | Small and Medium Enterprises |

**TABLE OF CONTENTS**

[**1.** **Introduction** 12](#_Toc154027274)

[**1.1.** **Brief** 12](#_Toc154027275)

[**1.2.** **Relevance to Course Modules** 12](#_Toc154027276)

[**1.3.** **Project Background** 12](#_Toc154027277)

[**1.4.** **Literature Review** 12](#_Toc154027278)

[**1.5.** **Analysis from Literature Review** 13](#_Toc154027279)

[**1.6.** **Methodology and Software Lifecycle for This Project** 14](#_Toc154027280)

[**1.6.1.** **Rationale behind Selected Methodology** 14](#_Toc154027288)

[**2.** **Problem Definition** 15](#_Toc154027289)

[**2.1.** **Problem Statement** 15](#_Toc154027294)

[**2.2.** **Deliverables and Development Requirements** 15](#_Toc154027295)

[**2.3.** **Current System (if applicable to your project)** 16](#_Toc154027296)

[**3.** **Requirement Analysis** 16](#_Toc154027297)

[**3.1.** **Use Cases Diagram(s)** 17](#_Toc154027302)

[**3.2.** **Detailed Use Case** 18](#_Toc154027303)

[**3.3.** **Functional Requirements** 36](#_Toc154027304)

[**3.4.** **Non-Functional Requirements** 39](#_Toc154027305)

[**4.** **Design and Architecture** 40](#_Toc154027306)

[**4.1.** **System Architecture** 40](#_Toc154027307)

[**4.2** **. Data Representation** 41](#_Toc154027316)

[**4.2.1.** **DFD Level 0** 41](#_Toc154027317)

[**4.2.2. DFD Level 1** 42](#_Toc154027318)

[**4.3. Process Flow/Representation** 44](#_Toc154027319)

[**4.4.** **Design Models** 44](#_Toc154027320)

[**5.** **Implementation** 45](#_Toc154027321)

[**5.1.** **Algorithm** 45](#_Toc154027322)

[**5.2.** **External APIs** 46](#_Toc154027324)

[**5.3.** **User Interface** 46](#_Toc154027325)

[**6.** **Testing and Evaluation** 51](#_Toc154027326)

[**6.1.** **Manual Testing** 51](#_Toc154027334)

[**6.1.1.** **System testing** 51](#_Toc154027342)

[**6.1.2.** **Unit Testing** 51](#_Toc154027343)

[**6.1.3.** **Functional Testing** 62](#_Toc154027344)

[**6.1.4.** **Integration Testing** 65](#_Toc154027345)

[**6.2.** **Automated Testing:** 66](#_Toc154027346)

[**6.2.1.** **Tools used:** 66](#_Toc154027347)

[**7.** **Conclusion** 66](#_Toc154027348)

[**8.** **Future Work** 67](#_Toc154027349)

[**9.** **References** 68](#_Toc154027350)

**LIST OF FIGURES**

Fig 3.1 Use Case Diagram 17

Fig 4.1 System Architecture 40

Fig 4.3 Process Flow/Representation 44

**LIST OF TABLES**

Table 1.1 Table1 8

Table 2.1 9

Table 5.1 47

1. **Introduction**

This chapter provides the overview of the project. The first paragraph of every chapter should provide the chapter summary.

* 1. **Brief**

The ERP (Enterprise Resource Planning) System presented in this thesis serves as a comprehensive solution designed to meet the specific needs of suppliers, addressing a critical gap in the market. This chapter offers an introductory glimpse into the ERP System, summarizing its outcomes, tools, methodologies, and key discussions covered throughout this report.

* 1. **Relevance to Course Modules**

The development of this ERP System project is directly tied to the coursework undertaken during our BSE program. This project draws inspiration from various course modules, incorporating the knowledge and skills acquired over the academic journey.

* 1. **Project Background**

This project started when we realized the challenges faced by small-scale suppliers, who often grapple with manual record-keeping and management processes. This ERP System aims to streamline their daily operations by offering a tailored solution. It addresses the question of what ERP (Enterprise Resource Planning) is and how it can revolutionize supplier management.

* 1. **Literature Review**

**Cloud Based ERP System for SME Industry**

The Enterprise resource planning (ERP) system has become the necessity of nearly all businesses. ERP architecture incorporates and defines a multitude of business processes and facilitates information transfer between them. An ERP system prevents duplication of information and ensures data integrity by gathering shared transactional shared data from multiple sources of an enterprise. There is a range of ERP systems available providing solutions to large businesses, however ERP systems for small and medium-sized businesses are obligatory. (Ijcsis, 2020)

**The Impact of Enterprise Resource Planning Systems on Small and Medium Enterprises**

The deployment of ERP systems is common practice in today’s business environment. Kumar and Hillegersber (2000) described the impact of ERP systems on corporations, and confirmed that ERPs were becoming so common in today’s business environment that they were described as “the price of entry for running a business” (p.24). Kumar and Hillegersber highlighted the significance and importance of medium-size corporations in the ERP marketplace, and confirmed that small and medium-size

corporations are beginning to embrace ERP technologies. The number of small and

medium-sized businesses retiring legacy systems in favor of ERP systems is increasing

exponentially. Esteves (2009) explained that in recent years SMEs are in a better position to acquire and implement ERP systems, which in the past were only available to larger corporations due to financial limitations as well as other factors. (Buleje, 2014)

**The usefulness of ERP systems for effective management**

Enterprise resource planning (ERP) systems offer distinct advantages in this new business environment as they lower operating costs, reduce cycle times and (arguably) increase customer satisfaction. (Spathis & Constantinides, 2003)

* 1. **Analysis from Literature Review**

In the preceding literature review, several key insights regarding ERP systems, particularly within the context of small and medium-sized enterprises (SMEs), were presented. These insights offer valuable perspectives that inform the development and objectives of our ERP System project tailored for suppliers. Here, we analyze and contextualize these findings within the scope of our project:

**ERP Systems as a Necessity for Businesses**

The literature highlights the growing indispensability of ERP systems across various business sectors. ERP architecture streamlines critical business processes and ensures data integrity by consolidating transactional data from multiple sources within an enterprise (Ijcsis, 2020). This aligns with our project's core objective of enhancing operational efficiency and data management for small-scale suppliers through a specialized ERP system.

**Impact of ERP on Small and Medium Enterprises**

Kumar and Hillegersber (2000) assert that ERP systems have become essential in the modern business landscape, describing them as "the price of entry for running a business" (p. 24). This recognition underscores the significance of ERP adoption for small and medium-sized enterprises (SMEs) (Buleje, 2014). As our project focuses on catering to the needs of small suppliers, this literature aligns with our aim to empower SMEs with the tools and capabilities necessary to thrive in today's competitive marketplace.

**Accessibility of ERP Systems for SMEs**

Esteves (2009) observes a shift in the ERP landscape, with SMEs increasingly well-positioned to acquire and implement ERP systems, which were previously predominantly available to larger corporations (Buleje, 2014). This trend supports our project's premise that a specialized ERP system can be accessible and beneficial to small suppliers. We aim to address the financial and operational constraints that SMEs often encounter when adopting ERP technology.

**Enhancing Efficiency and Customer Satisfaction**

Spathis and Constantinides (2003) note that ERP systems offer advantages such as reduced operating costs, shorter cycle times, and potentially improved customer satisfaction. These benefits align with our project's objectives, as we aim to provide small-scale suppliers with the tools to streamline their operations, optimize resource utilization, and ultimately enhance their competitiveness in the market.

The analysis of the literature review underscores the relevance and importance of our ERP System project. It validates the need for a supplier-focused ERP system tailored to the unique requirements of small-scale businesses. By drawing on these insights, our project strives to bridge the gap in the ERP landscape for SMEs, offering a comprehensive and accessible solution that empowers suppliers to thrive in today's dynamic business environment.

* 1. **Methodology and Software Lifecycle for This Project**

The chosen design methodology is Function-Based Programming with React, emphasizing the use of pure functions and functional composition. This approach is well-suited for creating a streamlined and efficient ERP solution for small-scale suppliers. We have also opted for the Scrum agile methodology as our SDLC model, characterized by iterative development.

1. 6. 1. **Rationale behind Selected Methodology**

The chosen design methodology for our ERP System is Function-Based Programming with React, primarily driven by the emphasis on pure functions and functional composition in software design. This approach promotes modularity and maintainability, aligning with our goal of creating an efficient ERP solution for small-scale suppliers.

* + 1. **Rationale behind Selected Methodology**

The adoption of the Scrum agile methodology as our Software Development Life Cycle (SDLC) model is rooted in its *iterative approach*. Scrum's emphasis on collaboration, flexibility, and risk management aligns with our project's objectives. By breaking the project into manageable sprints and allowing for frequent inspection and adaptation, Scrum ensures that our *ERP System remains adaptable to changing requirements and market dynamics*.

1. **Problem Definition**

Small-scale suppliers often encounter operational inefficiencies and data management challenges due to the lack of tailored ERP solutions. The problem lies in the absence of ERP systems specifically designed to cater to their unique needs, leading to manual and disjointed processes that hinder productivity and decision-making capabilities.

The outcome we aspire to achieve with our ERP System is to empower small-scale suppliers by offering them a user-friendly, modular, and scalable solution that streamlines their operations, enhances data integrity, and enables informed decision-making. By addressing these challenges, our project aims to contribute to the improved efficiency and competitiveness of small-scale suppliers in the business landscape.

4. 1. **Problem Statement**

Small-scale suppliers, despite their critical role in the supply chain, often face significant challenges due to the lack of efficient ERP systems. Manual and disjointed processes, data duplication, and limited decision support tools hinder their operational efficiency and competitiveness in the market.

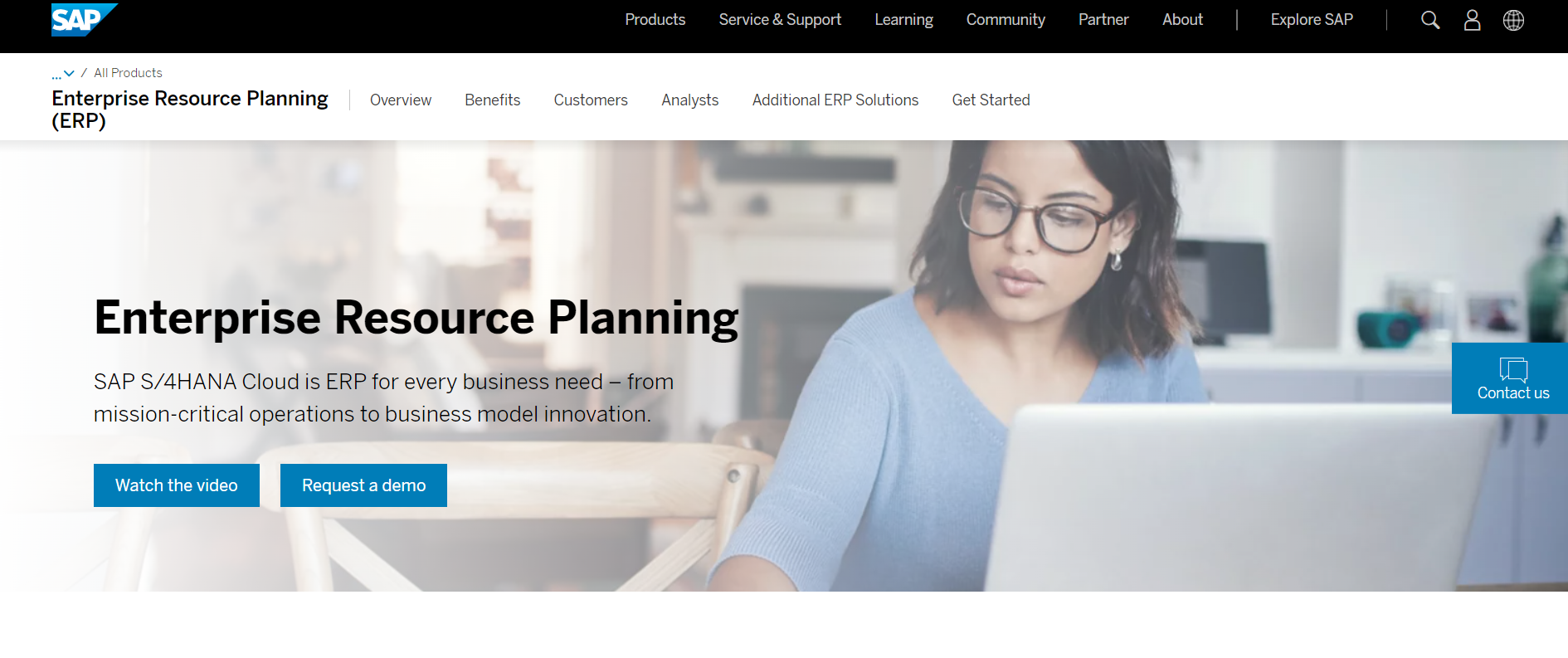
Small-scale suppliers require a dedicated ERP system that streamlines their operations, ensures data integrity, and empowers them to make informed decisions.

* 1. **Deliverables and Development Requirements**

The deliverables for our ERP System project encompass a fully functional and user-friendly web-based ERP system designed exclusively for small-scale suppliers. This system will comprise modules for *finance management, customer relationship management (CRM), inventory management, and human resource management (HRM)*. It will enable users to perform tasks such as *recording financial transactions, managing accounts payable and receivable, tracking customer interactions, updating product inventory, and overseeing employee details, including attendance*.

Development requirements include the utilization of the MERN stack, comprising MongoDB for the database, Express.js for the web application framework, React.js for the user interface, and Node.js for the server-side runtime environment. Additionally, the system must operate seamlessly across various web browsers and devices, ensuring global accessibility for users from different geographical locations.

* 1. **Current System (if applicable to your project)**



*Figure 2.1: SAP ERP*

*Table 2.1: Sample Table*

|  |  |  |
| --- | --- | --- |
| **Existing System** | **Weaknesses** | **Solutions** |
| SAP ERP | Complex implementation | The ERP System offers simplified and streamlined implementation processes, reducing complexity. |
| SAP ERP | High Costs | A cost-effective alternative, tailored to the needs of small-scale suppliers, is provided by the ERP System. |
| SAP ERP | Challenging for Small Suppliers | The ERP System is designed to cater specifically to the requirements and resources of small-scale suppliers, ensuring usability and affordability. |

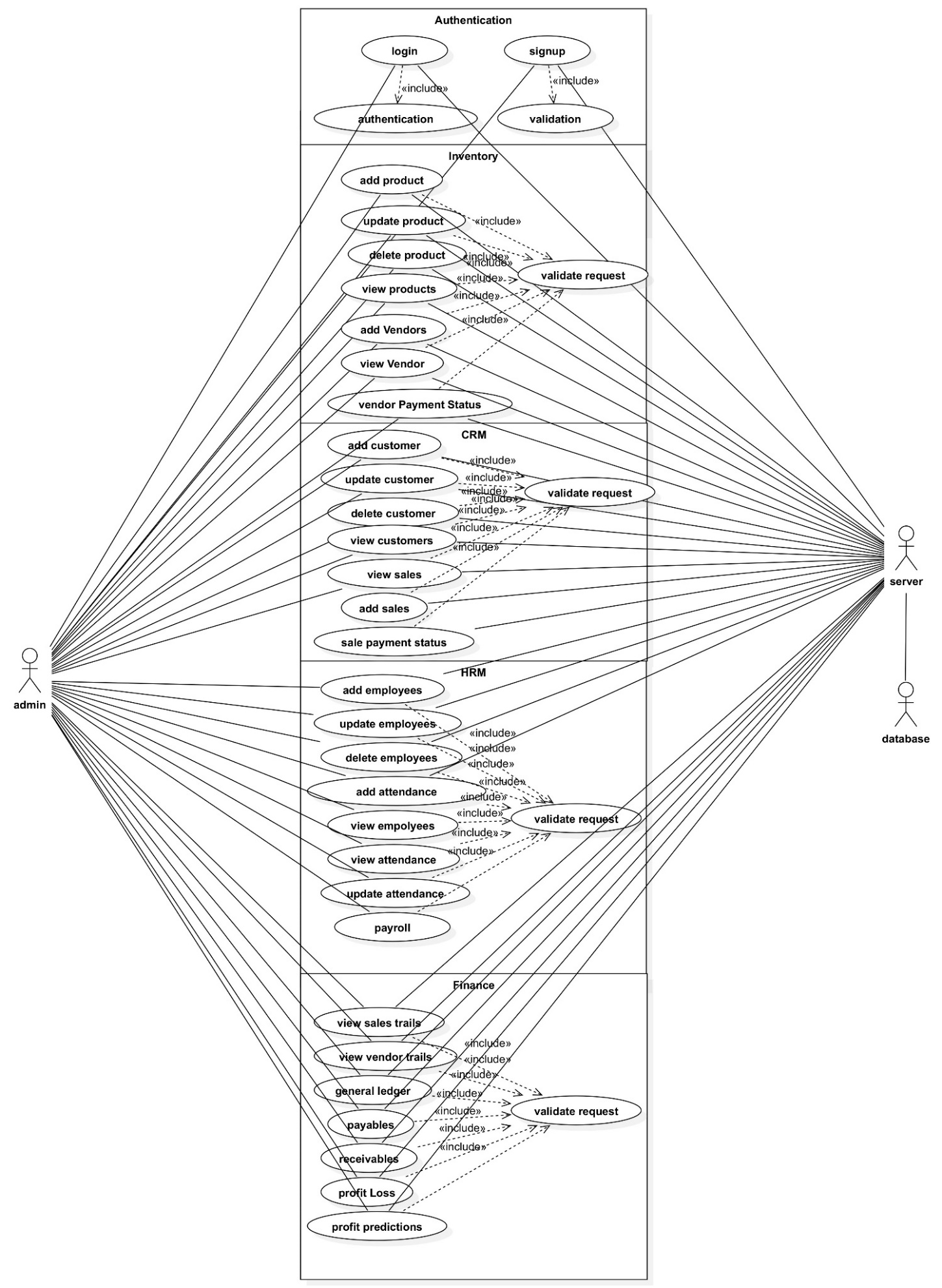
1. **Requirement Analysis**

The requirement analysis phase of the ERP System project involves in-depth examination and documentation of the specific needs and expectations of small-scale suppliers.

This section encompasses the identification of functional and non-functional requirements essential for the successful development and implementation of the ERP system.

4. 1. **Use Cases Diagram(s)**

*Fig 3.1 Use Case Diagram*



* 1. **Detailed Use Case**

**Use Case 1: Login**

|  |  |
| --- | --- |
| **Column** | Description |
| **Identifier** | UC-1 |
| **Use Case Name** | Login |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to log in to the system. |
| **Trigger** | The user initiates the login process. |
| **Preconditions** | None |
| **Postconditions** | The user is successfully logged in, granting access to the system. |
| **Normal Flow** | 1. The user provides valid credentials. 2. The system validates the credentials. 3. The system grants access to the user. |
| **Alternative Flows** | None |
| **Exceptions** | Invalid credentials: The system displays an error message. |
| **Business Rules** | None |

**Use Case 2: Signup**

|  |  |
| --- | --- |
| **Column** | Description |
| **Identifier** | UC-2 |
| **Use Case Name** | Signup |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows a new user to create an account. |
| **Trigger** | The user initiates the signup process. |
| **Preconditions** | None |
| **Postconditions** | The user account is successfully created. |
| **Normal Flow** | 1. The user provides necessary information for signup. 2. The system validates the information. 3. The system creates a new user account. |
| **Alternative Flows** | None |
| **Exceptions** | Existing username or email: The system prompts the user to choose different credentials. |
| **Business Rules** | None |

**Use Case 3: Add Product**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-3 |
| **Use Case Name** | Add Product |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to add a new product to the inventory. |
| **Trigger** | The user indicates the need to add a new product. |
| **Preconditions** | The user is authenticated and has access to the product management functionality. |
| **Postconditions** | The new product is added to the inventory. |
| **Normal Flow** | 1. The user selects the option to add a new product. 2. The user provides necessary details for the product. 3. The system validates the information. 4. The system adds the new product to the inventory. |
| **Alternative Flows** | None |
| **Exceptions** | Incomplete or incorrect product details: The system prompts The user to provide valid information. |
| **Business Rules** | None |

**Use Case 4: Update Product**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-4 |
| **Use Case Name** | Update Product |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to update product information in the inventory. |
| **Trigger** | The user indicates the need to update product information. |
| **Preconditions** | The user is authenticated and has access to the product management functionality. |
| **Postconditions** | The product information is successfully updated. |
| **Normal Flow** | 1. The user selects the option to update a product. 2. The user selects the product to be updated. 3. The user modifies the necessary product details. 4. The system validates the updated information. 5. The system updates the product information. |
| **Alternative Flows** | None |
| **Exceptions** | Unauthorized user: The system denies access to Unauthorized users. |
| **Business Rules** | None |

**Use Case 5: Delete Product**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-5 |
| **Use Case Name** | Delete Product |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to delete a product from the inventory. |
| **Trigger** | The user indicates the need to delete a product. |
| **Preconditions** | The user is authenticated and has access to the product management functionality. |
| **Postconditions** | The selected product is successfully deleted from the inventory. |
| **Normal Flow** | 1. The user selects the option to delete a product. 2. The user chooses the product to be deleted. 3. The system prompts for confirmation. 4. The user confirms the deletion. 5. The system removes the product from the inventory. |
| **Alternative Flows** | None |
| **Exceptions** | Unauthorized user: The system denies access to Unauthorized users. |
| **Business Rules** | None |

**Use Case 6: View Product**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-6 |
| **Use Case Name** | View Product |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to view details of a product in the inventory. |
| **Trigger** | The user indicates the need to view product details. |
| **Preconditions** | The user is authenticated and has access to the product management functionality. |
| **Postconditions** | The product details are displayed to the user. |
| **Normal Flow** | 1. The user selects the option to view products. 2. The system retrieves and displays a list of products. 3. The user selects a specific product to view details. 4. The system shows detailed information about the selected product. |
| **Alternative Flows** | None |
| **Exceptions** | None |
| **Business Rules** | None |

**Use Case 7: View Vendor**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-7 |
| **Use Case Name** | View Vendor |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to view details of a vendor. |
| **Trigger** | The user indicates the need to view vendor details. |
| **Preconditions** | The user is authenticated and has access to vendor information. |
| **Postconditions** | The vendor details are displayed to the user. |
| **Normal Flow** | 1. The user selects the option to view vendors. 2. The system retrieves and displays a list of vendors. 3. The user selects a specific vendor to view details. 4. The system shows detailed information about the selected vendor. |
| **Alternative Flows** | None |
| **Exceptions** | None |
| **Business Rules** | None |

**Use Case 8: Add Vendor**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-8 |
| **Use Case Name** | Add Vendor |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to add a new vendor to the system. |
| **Trigger** | The user indicates the need to add a new vendor. |
| **Preconditions** | The user is authenticated and has access to vendor management functionality. |
| **Postconditions** | The new vendor is successfully added to the system. |
| **Normal Flow** | 1. The user selects the option to add a new vendor. 2. The user provides necessary details for the vendor, such as name and contact information. 3. The system validates the vendor details. 4. The system adds the new vendor to the system. |
| **Alternative Flows** | None |
| **Exceptions** | Incomplete or Invalid vendor information: The system prompts The user to provide valid details. |
| **Business Rules** | Vendor data should be complete and accurate, including contact information. |

**Use Case 9: Vendor Payment System**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-9 |
| **Use Case Name** | Vendor Payment Status |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to view payments to vendors. |
| **Trigger** | The user indicates the need to view vendor payments. |
| **Preconditions** | PRE-1: The user is logged into the system. |
| **Postconditions** | POST-1: The vendor payment is processed and recorded. |
| **Normal Flow** | 1. The user selects the option to view vendor payments status. 2. The system displays a list of outstanding payments and cleared payments of vendors. |
| **Alternative Flows** | None |
| **Exceptions** | None |
| **Business Rules** | All vendor payments should be accurately recorded in The system. |

**Use Case 10: Add Customer**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-10 |
| **Use Case Name** | Add Customer |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to add a new customer to the system. |
| **Trigger** | The user indicates the need to add a new customer. |
| **Preconditions** | PRE-1: The user is logged into the system. |
| **Postconditions** | POST-1: The new customer is added to the system. |
| **Normal Flow** | 1. The user selects the option to add a new customer. 2. The user provides necessary details for the customer, such as name, contact information, and additional relevant data. 3. The system validates the customer details. 4. The system adds the new customer to the system. |
| **Alternative Flows** | None |
| **Exceptions** | Incomplete or Invalid customer information: The system prompts The user to provide valid details. |
| **Business Rules** | customer data should be complete and accurate, including contact information. |

**Use Case 11: Update Customer**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-11 |
| **Use Case Name** | Update Customer |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to update customer information. |
| **Trigger** | The user indicates the need to update customer information. |
| **Preconditions** | PRE-1: The user is logged into the system. |
| **Postconditions** | POST-1: The customer information is updated in the system. |
| **Normal Flow** | 1. The user selects the option to update customer information. 2. The user selects the customer to be updated. 3. The user modifies the necessary customer details. 4. The system validates the updated information. 5. The system updates the customer information. |
| **Alternative Flows** | None |
| **Exceptions** | Only authorized users can update customer data. |

**Use Case 12: Delete Customer**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-12 |
| **Use Case Name** | Delete Customer |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to delete a customer from the system. |
| **Trigger** | The user indicates the need to delete a customer. |
| **Preconditions** | PRE-1: The user is logged into the system. |
| **Postconditions** | POST-1: The customer is successfully removed from the system. |
| **Normal Flow** | 1. The user selects the option to delete a customer. 2. The user chooses the customer to be deleted. 3. The system prompts the user for confirmation. 4. The user confirms the deletion. 5. The system removes the customer from the system. |
| **Alternative Flows** | Cancel Deletion: If The user decides not to delete The customer, The system cancels The Deletion process. |
| **Exceptions** | Only authorized users can delete a customer. |

**Use Case 13: View Customer**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-13 |
| **Use Case Name** | View Customer |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to view detailed information about a customer. |
| **Trigger** | The user indicates the need to view customer details. |
| **Preconditions** | PRE-1: The user is logged into the system. |
| **Postconditions** | POST-1: The user views the detailed information about the selected customer. |
| **Normal Flow** | 1. The user selects the option to view customer details. 2. The system displays list of customers with all the information about the customer. |
| **Alternative Flows** | None |
| **Exceptions** | None |

**Use Case 14: Add Sales**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-14 |
| **Use Case Name** | Add Sales |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to add sales transactions to the system. |
| **Trigger** | The user indicates the need to add a new sales transaction. |
| **Preconditions** | PRE-1: The user is logged into the system. |
| **Postconditions** | POST-1: The new sales transaction is successfully added to the system. |
| **Normal Flow** | 1. The user selects the option to add a new sales transaction. 2. The user provides details for the sales, including product, quantity, and pricing information. 3. The system validates the sales transaction details. 4. The system adds the new sales transaction to the records. |
| **Alternative Flows** | None |
| **Exceptions** | Incomplete or Invalid sales transaction details: The system prompts The user to provide valid information. |

**Use Case 15: View Sales**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-15 |
| **Use Case Name** | View Sales |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to view a list of sales transactions. |
| **Trigger** | The user indicates the need to view sales transactions. |
| **Preconditions** | PRE-1: The user is logged into the system. |
| **Postconditions** | POST-1: The user views the list of sales transactions. |
| **Normal Flow** | 1. The user selects the option to view sales transactions. 2. The system retrieves and displays the list of sales transactions. |
| **Alternative Flows** | None |
| **Exceptions** | None |

**Use Case 16: Sale Payment Status**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-16 |
| **Use Case Name** | Sale Payment Status |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to check the payment status of a sales transaction. |
| **Trigger** | The user indicates the need to check the payment status of a sale. |
| **Preconditions** | PRE-1: The user is logged into the system. |
| **Postconditions** | POST-1: The user views the payment status of the selected sales transaction. |
| **Normal Flow** | 1. The user selects the option to check sale payment status. 2. The system displays the list of the payment status. |
| **Alternative Flows** | None |
| **Exceptions** | None |

**Use Case 17: Add Employees**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-17 |
| **Use Case Name** | **Add Employees** |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to add new employees to the system. |
| **Trigger** | The user indicates the need to add a new employee. |
| **Preconditions** | PRE-1: The user is logged into the system. |
| **Postconditions** | POST-1: The new employee is successfully added to the system. |
| **Normal Flow** | 1. The user selects the option to add a new employee. 2. The user provides necessary details for the new employee, including personal information and role. 3. The system validates the employee details. 4. The system adds the new employee to the records. |
| **Alternative Flows** | None |
| **Exceptions** | Incomplete or Invalid employee details: The system prompts The user to provide valid information. |

**Use Case 18: Update Employees**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-18 |
| **Use Case Name** | Update Employees |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to update information about existing employees. |
| **Trigger** | The user indicates the need to update employee information. |
| **Preconditions** | PRE-1: The user is logged into the system. PRE-2: There are existing employees in the system. |
| **Postconditions** | POST-1: The employee information is successfully updated in the system. |
| **Normal Flow** | 1. The user selects the option to update employee information. 2. The user chooses the specific employee to update. 3. The user modifies the necessary details for the selected employee. 4. The system validates the updated information. 5. The system updates the employee information in the records. |
| **Alternative Flows** | Cancel update: If The user decides not to proceed with The updates, The system cancels The process. |
| **Exceptions** | Incomplete or Invalid updated information: The system prompts The user to provide valid information. |

**Use Case 19: Delete Employees**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-19 |
| **Use Case Name** | Delete Employees |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to remove employees from the system. |
| **Trigger** | The user indicates the need to delete an existing employee. |
| **Preconditions** | PRE-1: The user is logged into the system. PRE-2: There are existing employees in the system. |
| **Postconditions** | POST-1: The selected employee is successfully removed from the system. |
| **Normal Flow** | 1. The user selects the option to delete an employee. 2. The user chooses the specific employee to delete. 3. The system confirms the deletion with the user. 4. The system removes the selected employee from the records. |
| **Alternative Flows** | Cancel Deletion: If The user decides not to proceed with The Deletion, The system cancels The process. |
| **Exceptions** | Invalid selection: If The selected employee is not found in The system, The system notifies The user. |

**Use Case 20: Add Attendance**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-20 |
| **Use Case Name** | Add Attendance |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to record attendance for employees. |
| **Trigger** | The user indicates the need to add attendance records. |
| **Preconditions** | PRE-1: The user is logged into the system. PRE-2: There are existing employees in the system. |
| **Postconditions** | POST-1: The attendance records are successfully added to the system. |
| **Normal Flow** | 1. The user selects the option to add attendance. 2. The user chooses the specific date and employees for attendance. 3. The user marks the attendance for the selected employees. 4. The system validates the attendance records. 5. The system adds the attendance records to the database. |
| **Alternative Flows** | None |
| **Exceptions** | Incomplete or Invalid attendance records: The system prompts The user to provide valid information. |

**Use Case 21: View Attendance**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-21 |
| **Use Case Name** | View Attendance |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to view attendance records for employees. |
| **Trigger** | The user indicates the need to view attendance records. |
| **Preconditions** | PRE-1: The user is logged into the system. |
| **Postconditions** | POST-1: The user views the attendance records. |
| **Normal Flow** | 1. The user selects the option to view attendance. 2. The user chooses the specific date or date range for attendance records. 3. The system retrieves and displays the attendance records. |
| **Alternative Flows** | None |
| **Exceptions** | None |

**Use Case 22: Update Attendance**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-22 |
| **Use Case Name** | Update Attendance |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to modify attendance records for employees. |
| **Trigger** | The user indicates the need to update attendance records. |
| **Preconditions** | PRE-1: The user is logged into the system. PRE-2: There are existing attendance records in the system. |
| **Postconditions** | POST-1: The attendance records are successfully updated in the system. |
| **Normal Flow** | 1. The user selects the option to update attendance. 2. The user chooses the specific date and employees for attendance update. 3. The user modifies the attendance status for the selected employees. 4. The system validates the updated attendance records. 5. The system updates the attendance records in the database. |
| **Alternative Flows** | None |
| **Exceptions** | Incomplete or Invalid updated attendance records: The system prompts The user to provide valid information. |

**Use Case 23: Payroll**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-23 |
| **Use Case Name** | Payroll |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to manage payroll for employees. |
| **Trigger** | The user indicates the need to process payroll. |
| **Preconditions** | PRE-1: The user is logged into the system. PRE-2: There are existing employees with recorded attendance in the system. |
| **Postconditions** | POST-1: The payroll is successfully processed, and employees are compensated accordingly. |
| **Normal Flow** | 1. The user selects the option for payroll processing. 2. The user chooses the month. 3. The system calculates salaries based on attendance records and predefined salary structures. 4. The system generates a payroll report for the selected period. |
| **Alternative Flows** | None |
| **Exceptions** | None |

**Use Case 24: Sale Trails**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-24 |
| **Use Case Name** | Sale Trails |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to view and pay trails of sales transactions. |
| **Trigger** | The user indicates the need to clear sale trails. |
| **Preconditions** | PRE-1: The user is logged into the system. PRE-2: There are existing sales transactions in the system. |
| **Postconditions** | POST-1: The user confirms the sales. |
| **Normal Flow** | 1. The user selects the option to view sale trails. 2. The system retrieves and displays a list of sales transactions with details. 3. The user can confirm the payment. 4. The system validates and confirms the payment. |
| **Alternative Flows** | None |
| **Exceptions** | None |

**Use Case 25: Vendor Trails**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-25 |
| **Use Case Name** | Vendor Trails |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to pay trails of vendor transactions. |
| **Trigger** | The user indicates the need to pay vendor trails. |
| **Preconditions** | PRE-1: The user is logged into the system. PRE-2: There are existing vendor transactions in the system. |
| **Postconditions** | POST-1: The user pays the vendor orders. |
| **Normal Flow** | 1. The user selects the option to view vendor trails. 2. The system retrieves and displays a list of vendor transactions with details. 3. The user can pay the remaining payment. 4. The system confirms the payment. |
| **Alternative Flows** | None |
| **Exceptions** | None |

**Use Case 26: General Ledger**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-26 |
| **Use Case Name** | General Ledger |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to view all the transactions in the general ledger. |
| **Trigger** | The user indicates the need to view general ledger transactions. |
| **Preconditions** | PRE-1: The user is logged into the system. |
| **Postconditions** | POST-1: The user views the general ledger transactions. |
| **Normal Flow** | 1. The user selects the option for the general ledger from the finance module. 2. The system retrieves and displays all transactions recorded in the general ledger. |
| **Alternative Flows** | None |
| **Exceptions** | None |

**Use Case 27: Payables**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-27 |
| **Use Case Name** | Payables |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to view payables for vendors. |
| **Trigger** | The user indicates the need to view payables. |
| **Preconditions** | PRE-1: The user is logged into the system. PRE-2: There are existing vendor transactions in the system. |
| **Postconditions** | POST-1: The payables are successfully viewed. |
| **Normal Flow** | 1. The user selects the option for payables view. 2. The system retrieves and displays a list of vendor transactions with all payments. |
| **Alternative Flows** | None |
| **Exceptions** | None |

**Use Case 28: Receivables**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-28 |
| **Use Case Name** | Receivables |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to view receivables for sales transactions. |
| **Trigger** | The user indicates the need to view receivables. |
| **Preconditions** | PRE-1: The user is logged into the system. PRE-2: There are existing sales transactions in the system. |
| **Postconditions** | POST-1: The receivables are successfully viewed. |
| **Normal Flow** | 1. The user selects the option for receivables view. 2. The system retrieves and displays a list of sales transactions. |
| **Alternative Flows** | None |
| **Exceptions** | None |

**Use Case 29: Profit/Loss**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-29 |
| **Use Case Name** | Profit/Loss |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to view the profit and loss statement. |
| **Trigger** | The user indicates the need to view the profit and loss statement. |
| **Preconditions** | PRE-1: The user is logged into the system. PRE-2: There are existing financial transactions in the system. |
| **Postconditions** | POST-1: The user views the profit and loss statement. |
| **Normal Flow** | 1. The user selects the option for the profit and loss statement from the finance module. 2. The system calculates and displays the profit and loss statement based on financial transactions. |
| **Alternative Flows** | None |
| **Exceptions** | None |

**Use Case 30: Profit Prediction**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-30 |
| **Use Case Name** | Profit Prediction |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to predict future profits. |
| **Trigger** | The user indicates the need to predict future profits. |
| **Preconditions** | PRE-1: The user is logged into the system. PRE-2: There is sufficient historical financial data available in the system. |
| **Postconditions** | POST-1: The system generates and displays the predicted profit using the Prophet Model. |
| **Normal Flow** | 1. The user selects the option for profit prediction from the finance module. 2. The system utilizes the Prophet Model to analyze historical financial data and generates a prediction for future profits. |
| **Alternative Flows** | None |
| **Exceptions** | None |

**Use Case 31: View Employees**

|  |  |
| --- | --- |
| **Column** | **Description** |
| **Identifier** | UC-31 |
| **Use Case Name** | View Employees |
| **Actors** | Primary Actor: User |
| **Description** | This use case allows the user to view a list of employees and their relevant details. |
| **Trigger** | The user indicates the need to access and view employee information. |
| **Preconditions** | PRE-1: The user is logged into the system. PRE-2: Employee records exist in the system. |
| **Postconditions** | POST-1: The user successfully views the list of employees and their details. |
| **Normal Flow** | 1. The user selects the option to view employees from the employees module. 2. The system retrieves and displays a list of employees along with their relevant details. |
| **Alternative Flows** | None |
| **Exceptions** | No employee records: If there are No Existing employee records, The system notifies The user that there is No information to display. |

* 1. **Functional Requirements**

FR1: Authentication for Login

* The system shall authenticate users during the login process using secure credentials.
* Only registered users with valid credentials shall be allowed to log in.

FR2: Secure User Registration for Signup

* The system shall provide a secure user registration process for new users.
* User registration shall require valid and unique information, including email and password.

FR3: Add Product to Inventory

* The system shall allow users to add new products to the inventory.
* Product information, including name, description, quantity, and price, must be captured during the addition process.

FR4: Update Product Information

* Users with appropriate privileges shall be able to update product information in the inventory.
* The system shall validate and apply the modifications to ensure accurate and up-to-date product records.

FR5: Delete Product from Inventory

* The system shall allow authorized users to delete products from the inventory.
* Deletion must be performed securely, and the system shall confirm the removal of the product.

FR6: View Product Details

* Users shall be able to view detailed information about products in the inventory.
* The system shall display product details, including name, description, quantity, and price.

FR7: View Vendor Information

* The system shall provide a view of vendor details.
* Users should be able to access information about vendors associated with products.

FR8: Add Vendor to System

* Authorized users shall be able to add new vendors to the system.
* Vendor information, including name and contact details, must be captured during the addition process.

FR9: Vendor Payment Status

* The system shall include a vendor payment status.
* Users should be able to view payments to vendors.

FR10: Add Customer Information

* The system shall allow users to add new customer records.
* Customer information, including name, contact details, and sales history, must be captured.

FR11: Update Customer Information

* Users with appropriate privileges shall be able to update customer information.
* The system shall validate and apply modifications to ensure accurate and up-to-date customer records.

FR12: Delete Customer Record

* Authorized users shall be able to delete customer records securely.
* The system shall confirm the deletion and remove the customer from the system.

FR13: View Customer Details

* Users shall have the ability to view detailed information about customers.
* The system shall display customer details, including name, contact information, and sales history.

FR14: Add Sales Transaction

* The system shall allow users to add new sales transactions.
* Sales details, including customer information and transaction amount, must be captured.

FR15: View Sales Records

* Users shall be able to view a list of sales records.
* The system shall display sales information, including details and amounts.

FR16: Sale Payment Status

* The system shall track and display the payment status of sales transactions.
* Users should be able to identify whether a sale has been paid or is pending.

FR17: Add Employees to HRM

* Authorized users shall be able to add new employees to the Human Resource Management (HRM) module.
* Employee details, including personal information and job roles, must be captured.

FR18: Update Employee Information

* Users with appropriate privileges shall be able to update employee information.
* The system shall validate and apply modifications to ensure accurate and up-to-date employee records.

FR19: Delete Employee Record

* The system shall allow authorized users to delete employee records securely.
* The system shall confirm the deletion and remove the employee from the HRM module.

FR20: Add Attendance Record

* Users shall be able to add attendance records for employees.
* The system shall capture attendance details, including day and month.

FR21: View Employee Attendance

* Users shall have the ability to view attendance records for employees.
* The system shall display attendance information, including dates and attendance status.

FR22: Update Attendance Record

* Users with appropriate privileges shall be able to update attendance records.
* The system shall validate and apply modifications to ensure accurate and up-to-date attendance information.

FR23: Payroll Processing

* The system shall include functionality for payroll processing.
* Users should be able to calculate and process payroll based on employee attendance and salary information.

FR24: Sale Trails

* Users shall be able to view trails of sales transactions and get paid for them.

FR25: Vendor Trails

* Users shall have the ability to pay for vendor-related activities.

FR26: General Ledger Management

* The system shall include a general ledger for financial transactions.
* Users should be able to view ledger transactions.

FR27: Payables

* The system shall provide functionality for viewing payables.
* Users should be able to track amounts payable to vendors.

FR28: Receivables

* The system shall include functionality for viewing receivables.
* Users should be able to track amounts receivable from customers.

FR29: Profit/Loss Calculation

* The system shall calculate and display profit/loss statements.
* Users should be able to view the financial performance of the organization.

FR30: Profit Prediction using Prophet Model

* The system shall include a prediction model, such as the Prophet Model, for forecasting profits.
* Users should be able to view predicted financial outcomes based on historical data and trends.
  1. **Non-Functional Requirements**

**Usability**

USE-1: The system shall have a consistent and intuitive user interface across all modules and screens.

USE-2: The system shall adhere to accessibility guidelines.

USE-3: The system shall provide clear and informative error messages and feedback to users for effective error handling and task completion.

**Performance**

PER-1: The system shall maintain fast response times less than 1 second.

PER-2: The system should be capable of handling a high volume of concurrent user requests.

PER-3: The system shall provide efficient search and retrieval capabilities, enabling users to quickly find and access relevant information.

PER-4: The system shall minimize network latency to ensure responsive communication between client and server components.

**Security**

SEC-1: The system shall enforce strict access control measures to ensure that only authorized personnel have the appropriate privileges to access and modify data.

1. **Design and Architecture**

This chapter encompasses the critical elements of the Software Design Description (SDD) report, providing a comprehensive overview of the design and architectural aspects of the ERP System.

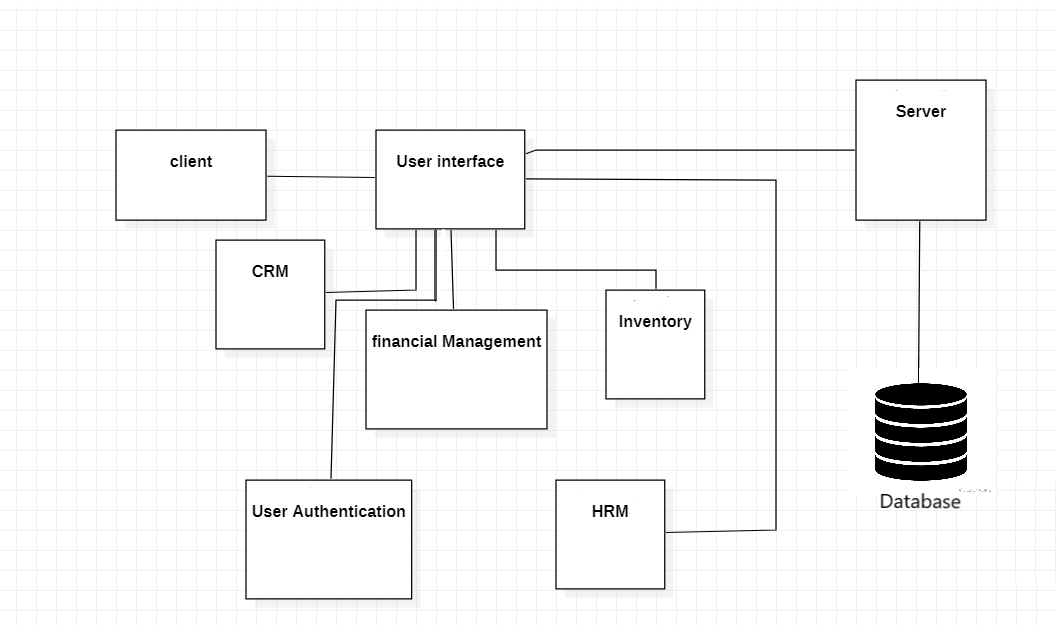
* 1. **System Architecture**

We adopted a Client-Server Architecture, where the system is divided into two distinct components: the client and the server.

The client represents the user interface, providing a web-based platform accessible from various devices and web browsers. It serves as the point of interaction for users, enabling them to input data, access information, and perform tasks seamlessly.

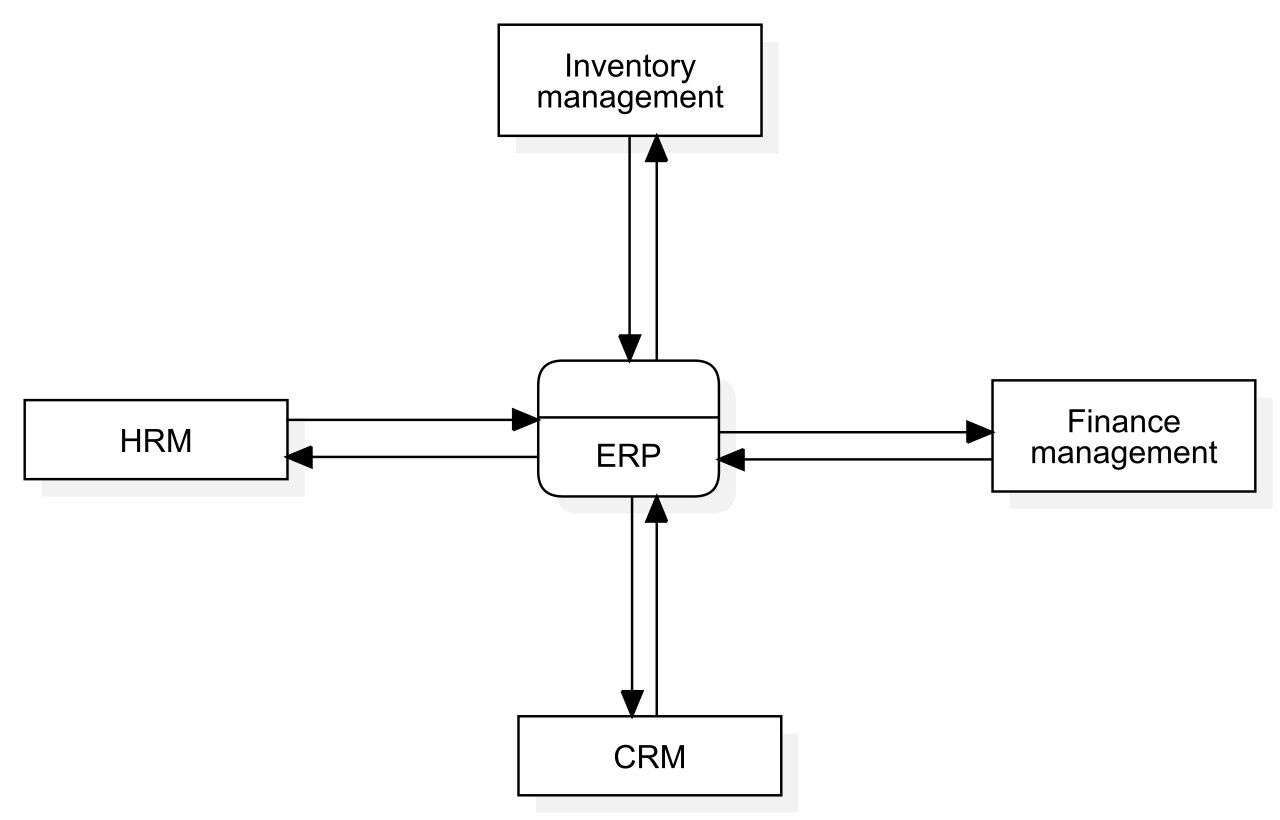
The server component hosts the core system functionalities, including data processing, storage, and business logic.

*Fig 4.1 System Architecture*

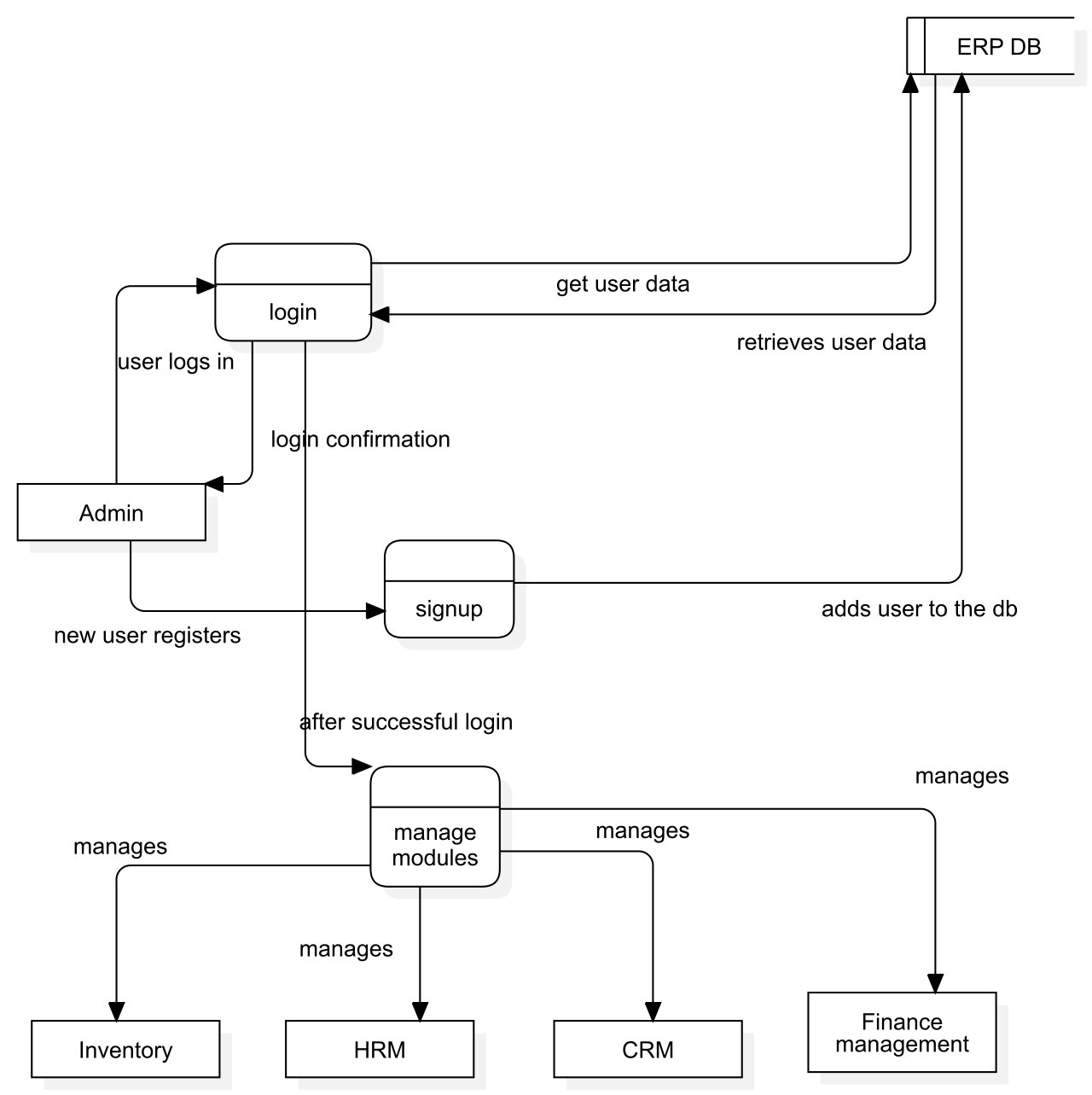


8. 1. **. Data Representation**

**4.2.1. DFD Level 0**



**4.2.2. DFD Level 1**



**Customer Relationship Management (CRM)**

* Manages customer records with information like names and contact details.
* Tracks customer interactions and sales history.
* Customer data is stored in a database for easy access.

**Human Resource Management (HRM)**

* Maintains employee records containing personal info, and roles.
* Supports attendance tracking.
* Employee data is organized in a structured database.

**Inventory Management**

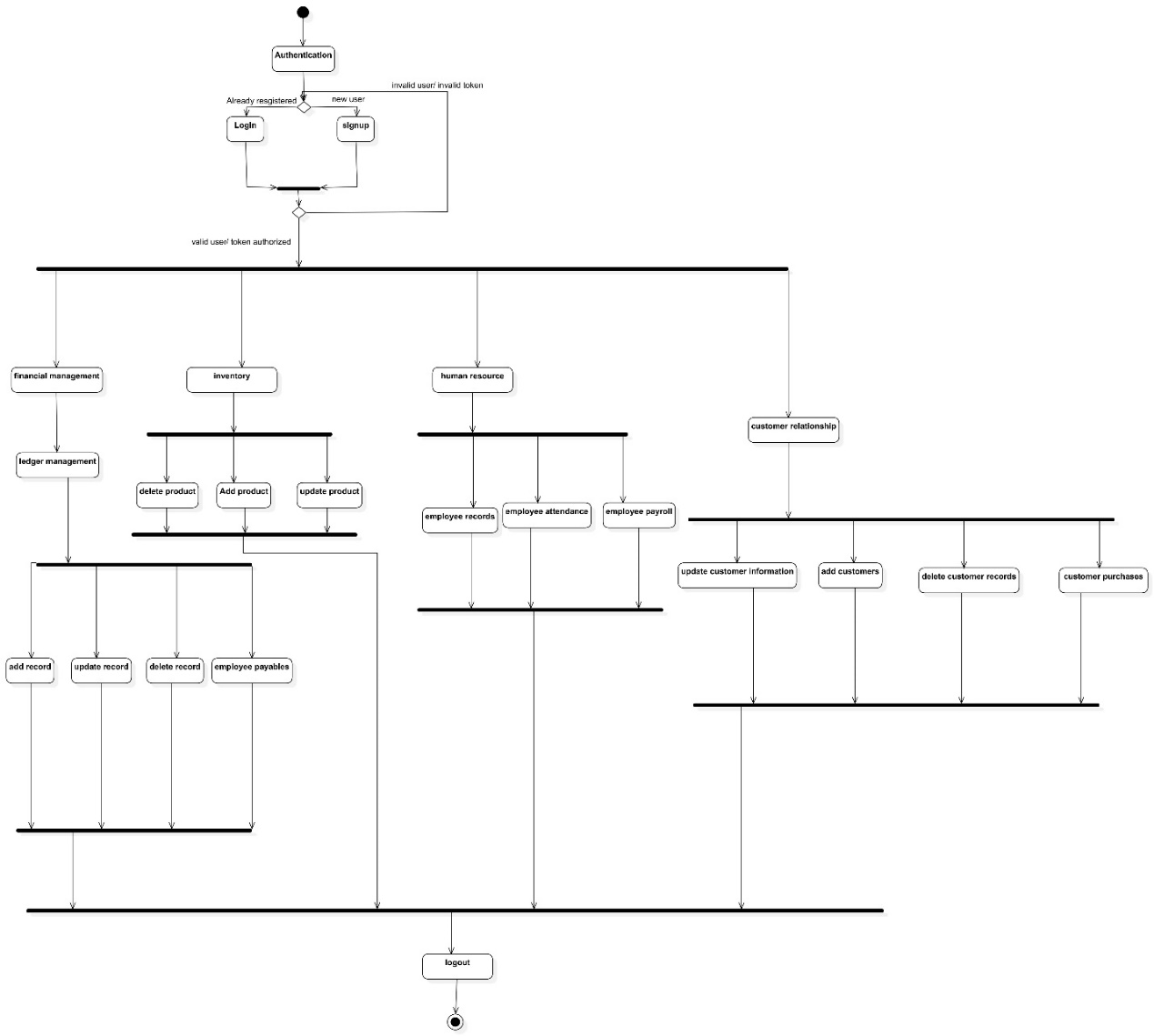
* Handles product information such as name, description, quantity, and price.
* Uses an inventory database to track stock levels accurately.
* Ensures efficient stock management.

**Database System**

* MongoDB is the chosen database system.
* Adopts a document-oriented data modeling approach.
* Utilizes JSON-like documents for data representation.
* Provides flexibility for data storage and retrieval.
* Enables scalability and adaptability of the ERP System.

**4.3. Process Flow/Representation**

*Fig 4.3 Process Flow/Representation*



* 1. **Design Models**

In the development of our ERP System, we employ several design models to structure and organize the software components effectively. These design models include:

**Component-Based Design**

Component-based design is central to our system architecture. It involves breaking down the ERP system into modular components, each responsible for specific functionalities such as finance management, CRM, HRM, and inventory management. This design approach promotes reusability, making it easier to maintain and extend the system in the future.

**Document-Oriented Data Modeling**

To structure and manage data efficiently, we employ a document-oriented data modeling approach. This design model aligns with our use of MongoDB as the database system. MongoDB's flexibility in handling JSON-like documents allows us to store and retrieve data in a way that suits the specific requirements of our ERP System.

**Agile Design**

We embrace an agile design approach, aligned with the Scrum methodology, to accommodate changing requirements and evolving user needs. Agile design allows for flexibility and adaptability throughout the development process, ensuring that the ERP System remains responsive to emerging challenges and opportunities.

1. **Implementation**
   1. **Algorithm**

**1. Authentication using JWT Tokens**

The system employs JWT Tokens to authenticate users, ensuring secure access to individual accounts. This authentication mechanism enables multiple users to utilize the system concurrently while maintaining distinct and protected user sessions.

**2. Main Modules**

The system is structured around four core modules: Inventory, Customers, Employees, and Finance. These modules serve as the primary organizational units, each focusing on specific aspects of the system's functionalities.

**3. Inventory Module**

The Inventory module encompasses key features such as View Products, View Vendors, Vendor Payment Status, Add Product, and Add Vendor. These functionalities enable users to manage and monitor product information, vendor details, and payment statuses efficiently.

**4. Customer Module**

Within the Customer module, users can access features such as View Customers, View Sales, Sale Payment Status, Add Customer, and Add Sale. This module facilitates the management of customer-related data, sales information, and payment statuses associated with sales transactions.

**5. Employees Module**

The Employees module incorporates functionalities like Add Employees, Add Attendance, View Employees, View Attendance, and Payroll. These features empower users to handle employee-related tasks, including attendance tracking and payroll management.

**6. Finance Module**

The Finance module provides a comprehensive set of features, including Sales Trails, View Vendor Trails, General Ledger, Payables, Receivables, Profit/Loss Analysis, and Profit Prediction utilizing the Prophet Model. This module focuses on financial aspects, offering insights into sales and vendor histories, ledger management, and advanced analytics for profit forecasting.

1. 1. **External APIs**

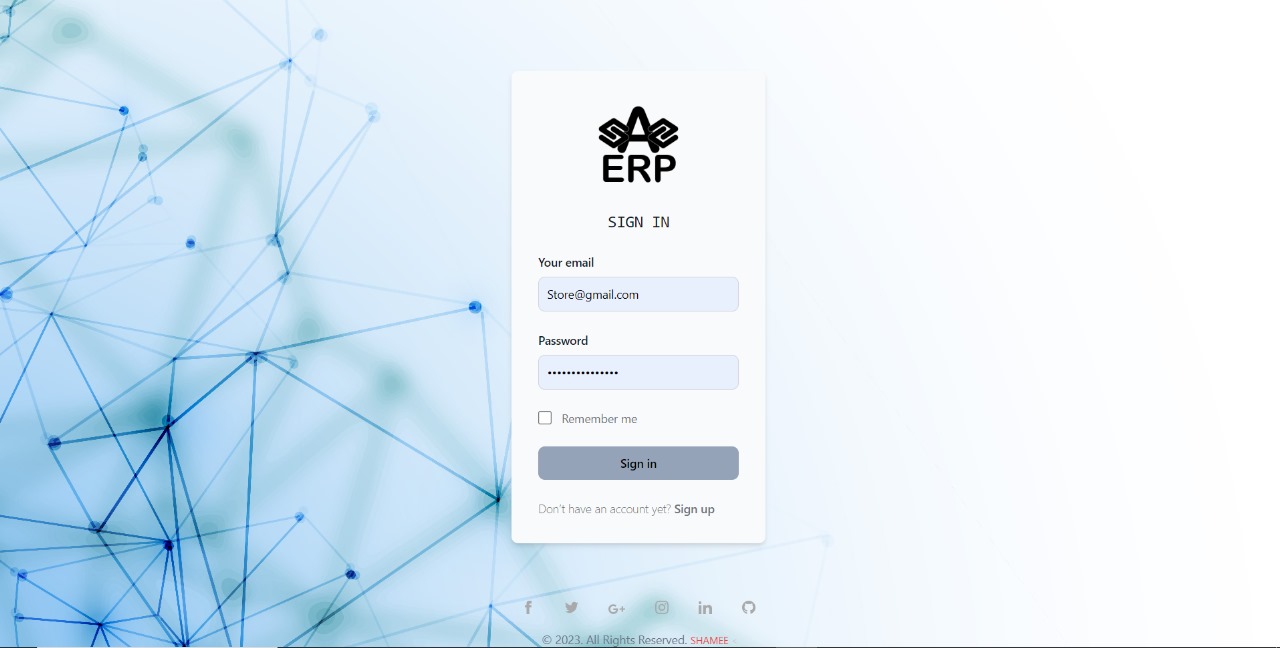
Table 5.1 shows the APIs we used in our ERP System

*Table 5.1: Details of APIs used in the project*

|  |  |  |  |
| --- | --- | --- | --- |
| **Name of API** | **Description of API** | **Purpose of Usage** | **Function/Class Name** |
| REST APIs | Communication interface allowing interaction with external systems/devices. | Facilitates data exchange and system integration. | Various functions across modules. |
| Flask API for Prophet Model | Flask API implementation for serving Prophet Model predictions. | Integrates Prophet Model for profit prediction in the Finance module. | ‘ProfitPrediction’ class in Finance module. |

* 1. **User Interface**

*Login Page*



*Sign Up Page*

A screenshot of a login form

Description automatically generated

*Dashboard Light Mode*



*Dashboard Dark Mode*



*View Products*

A screenshot of a computer

Description automatically generated

*Add Vendor*

A screenshot of a computer

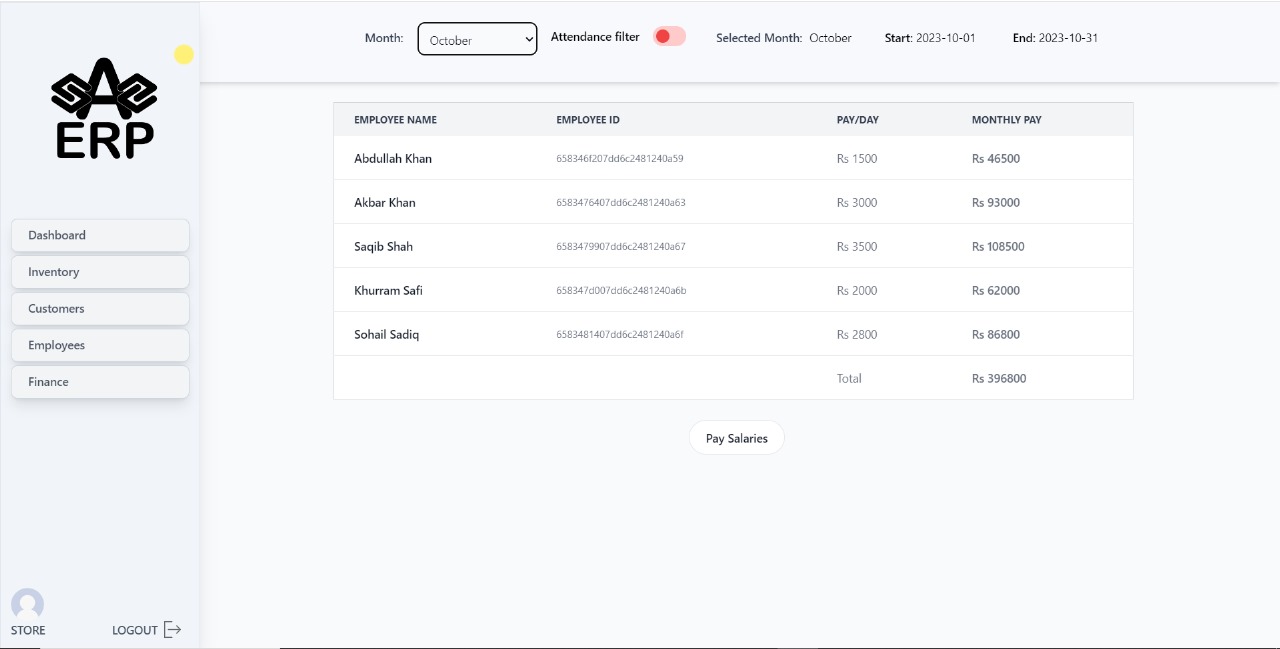
Description automatically generated

*View Customers*

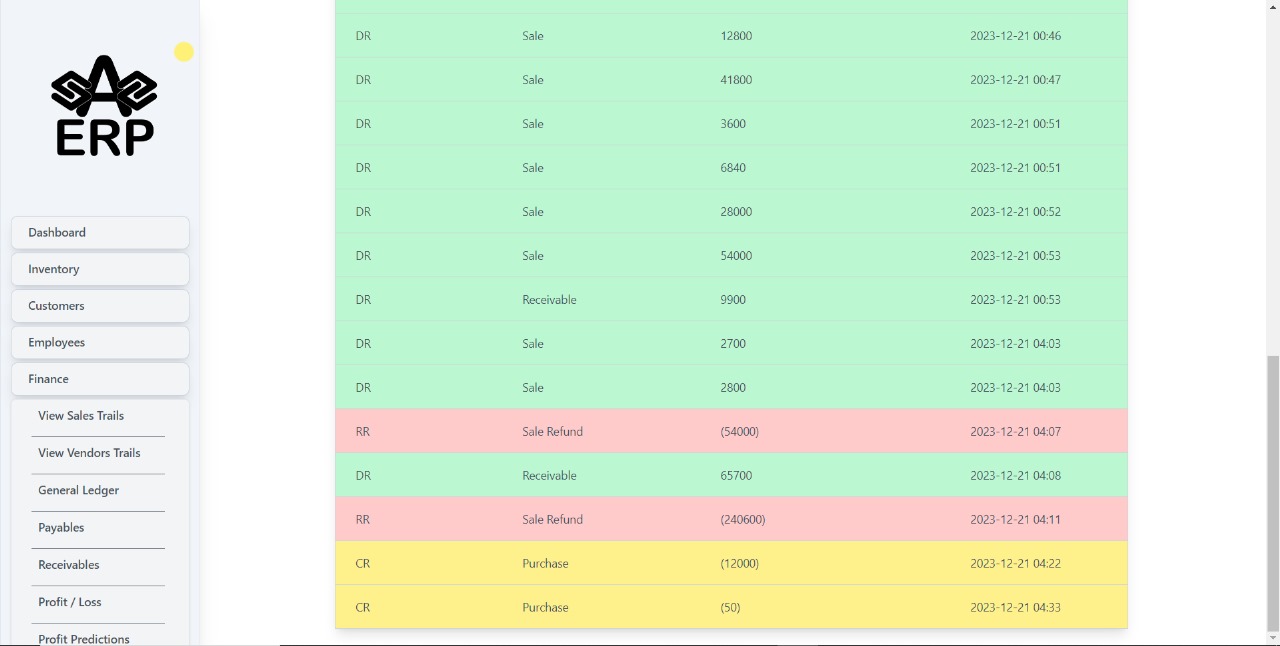
A screenshot of a computer

Description automatically generated

*Payroll*



*General Ledger*



*Profit Forecast*

A screenshot of a computer

Description automatically generated

1. **Testing and Evaluation**

This chapter describes the overall testing of our system “The ERP System”

7. 1. **Manual Testing**

Manual testing includes unit testing and functional testing of the system.

6. 1. 1. **System testing**

After the successful development of the system, rigorous testing is imperative to ensure that the system functions as intended and meets the predefined requirements. System testing serves the critical purpose of unveiling any hidden errors that may not be apparent to users. This testing phase comprises several essential types, including unit testing, functional testing, and integration testing. Each type of testing plays a unique role in ensuring the robustness and reliability of the ERP system.

* + 1. **Unit Testing**

Once the system has been successfully developed

**Unit Testing 1:** Login

**Testing Objective:** To ensure the login form is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify user login with correct data | email: Store Password: Store@gmail.com | Successfully log into the main page of the system | Pass |
| 2 | Verify user login with incorrect password | email: Store Password: Store@gmail | Display an error message indicating invalid credentials | Pass |
| 3 | Verify user login with empty fields | email:  Password: | Display an error message prompting to fill in both fields | Pass |

**Unit Testing 2:** Signup

**Testing Objective:** To ensure the signup form is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify user signup with valid data | Username: abc  Email: abc@gmail.com  Password: [abc@gmail.com](mailto:abc@gmail.com) | Successfully create a new user account | Pass |
| 2 | Verify user signup with existing email | email: Store@gmail.com  Password: [Store@gmail.com](mailto:Store@gmail.com) | Display an error message indicating username already exists | Pass |
| 3 | Verify user signup with weak password | email: Store  Password: Store | Display an error message indicating password strength requirements | Pass |

**Unit Testing 3:** Add Product

**Testing Objective:** To ensure the add product is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify adding a product with valid data | Product Name: [valid name] Quantity: [valid quantity] | Product is successfully added to the inventory | Pass |
| 2 | Verify adding a product with empty fields | Product Name: [empty] Quantity: [empty] | Display an error message prompting to fill in required fields | Pass |
| 3 | Verify adding a duplicate product | Product Name: [existing name] Quantity: [valid quantity] | Display an error message indicating duplicate product | Pass |

**Unit Testing 4:** Update Product

**Testing Objective:** To ensure the update product is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify updating a product with valid data | Product name: Lux  New Quantity: 100 | Product quantity is successfully updated | Pass |
| 2 | Verify updating a product with empty fields | Product name:  New Quantity: | Display an error message prompting to fill in required fields | Pass |
| 3 | Verify updating a non-existing product | Product name:  New Quantity: 50 | Display an error message indicating product not found | Pass |

**Unit Testing 5:** View product

**Testing Objective:** To ensure the view product is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify viewing a specific product | Product name: Mouse | Product details are displayed correctly | Pass |
| 2 | Verify viewing all products | No specific attributes | List of all products is displayed | Pass |
| 3 | Verify viewing a non-existing product | Product name: Car | Display an error message indicating product not found | Pass |

**Unit Testing 6:** View Vendor

**Testing Objective:** To ensure the view vendor is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify viewing a specific vendor | Vendor name: Abdullah | Vendor details are displayed correctly | Pass |
| 2 | Verify viewing all vendors | No specific attributes | List of all vendors is displayed | Pass |
| 3 | Verify viewing a non-existing vendor | Vendor name: Shakeel | Display an error message indicating vendor not found | Pass |

**Unit Testing 7:** Add Vendor

**Testing Objective:** To ensure the add vendor is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify adding a vendor with valid data | Vendor Name: Saqib Contact: 1234567897 | Vendor is successfully added | Pass |
| 2 | Verify adding a vendor with empty fields | Vendor Name: [empty] Contact: [empty] | Display an error message prompting to fill in required fields | Pass |

**Unit Testing 8:** Vendor Payment System

**Testing Objective:** To ensure the Vendor Payment System is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify recording payment for an existing vendor | Vendor ID: [valid ID] Amount: [valid amount] | Payment is successfully recorded | Pass |
| 2 | Verify recording payment for a non-existing vendor | Vendor ID: [non-existing ID] Amount: [valid amount] | Display an error message indicating vendor not found | Pass |
| 3 | Verify recording payment with invalid amount | Vendor ID: [valid ID] Amount: [invalid amount] | Display an error message indicating invalid amount | Pass |

**Unit Testing 9:** Signup

**Testing Objective:** To ensure the signup form is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify adding a customer with valid data | Customer Name: [valid name] Contact: [valid contact] | Customer is successfully added | Pass |
| 2 | Verify adding a customer with empty fields | Customer Name: [empty] Contact: [empty] | Display an error message prompting to fill in required fields | Pass |
| 3 | Verify adding a duplicate customer | Customer Name: [existing name] Contact: [valid contact] | Display an error message indicating duplicate customer | Pass |

**Unit Testing 10:** Add Customer

**Testing Objective:** To ensure the Add Customer is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify updating a customer with valid data | Customer ID: [valid ID] New Contact: [valid contact] | Customer contact is successfully updated | Pass |
| 2 | Verify updating a customer with empty fields | Customer ID: [valid ID] New Contact: [empty] | Display an error message prompting to fill in required fields | Pass |
| 3 | Verify updating a non-existing customer | Customer ID: [non-existing ID] New Contact: [valid contact] | Display an error message indicating customer not found | Pass |

**Unit Testing 11:** Delete Customer

**Testing Objective:** To ensure the delete customer is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify deleting an existing customer | Customer ID: [valid ID] | Customer is successfully deleted | Pass |
| 2 | Verify deleting a non-existing customer | Customer ID: [non-existing ID] | Display an error message indicating customer not found | Pass |
| 3 | Verify deleting a customer with associated transactions | Customer ID: [valid ID] | Display a warning message indicating associated transactions | Pass |

**Unit Testing 12:** View Customer

**Testing Objective:** To ensure the view customer is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify viewing a specific customer | Customer ID: [valid ID] | Customer details are displayed correctly | Pass |
| 2 | Verify viewing all customers | No specific attributes | List of all customers is displayed | Pass |
| 3 | Verify viewing a non-existing customer | Customer ID: [non-existing ID] | Display an error message indicating customer not found | Pass |

**Unit Testing 13:** Add Sale

**Testing Objective:** To ensure the add sale is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify adding a sale with valid data | Sale details: [valid data] | Sale is successfully added | Pass |
| 2 | Verify adding a sale with empty fields | Sale details: [empty] | Display an error message prompting to fill in required fields | Pass |
| 3 | Verify adding a duplicate sale | Sale details: [existing sale data] | Display an error message indicating duplicate sale | Pass |

**Unit Testing 14:** View Sale

**Testing Objective:** To ensure the view sale working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify viewing a specific sale | Sale ID: [valid ID] | Sale details are displayed correctly | Pass |
| 2 | Verify viewing all sales | No specific attributes | List of all sales is displayed | Pass |
| 3 | Verify viewing a non-existing sale | Sale ID: [non-existing ID] | Display an error message indicating sale not found | Pass |

**Unit Testing 15:** Payment Status

**Testing Objective:** To ensure the payment system is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify checking payment status of a paid sale | Sale ID: [valid ID] | Display payment status as 'Paid' | Pass |
| 2 | Verify checking payment status of an unpaid sale | Sale ID: [valid ID] | Display payment status as 'Unpaid' | Pass |
| 3 | Verify checking payment status of a non-existing sale | Sale ID: [non-existing ID] | Display an error message indicating sale not found | Pass |

**Unit Testing 16:** Add Employee

**Testing Objective:** To ensure the add employee form is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify adding an employee with valid data | Employee details: [valid data] | Employee is successfully added | Pass |
| 2 | Verify adding an employee with empty fields | Employee details: [empty] | Display an error message prompting to fill in required fields | Pass |
| 3 | Verify adding a duplicate employee | Employee details: [existing employee data] | Display an error message indicating duplicate employee | Pass |

**Unit Testing 17:** Update Employee details

**Testing Objective:** To ensure the update employee details form is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify updating employee details with valid data | Employee ID: [valid ID] New Role: [valid role] | Employee details are successfully updated | Pass |
| 2 | Verify updating employee details with empty fields | Employee ID: [valid ID] New Role: [empty] | Display an error message prompting to fill in required fields | Pass |
| 3 | Verify updating a non-existing employee | Employee ID: [non-existing ID] New Role: [valid role] | Display an error message indicating employee not found | Pass |

**Unit Testing 18:** Delete Employee

**Testing Objective:** To ensure the signup form is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify deleting an existing employee | Employee ID: [valid ID] | Employee is successfully deleted | Pass |
| 2 | Verify deleting a non-existing employee | Employee ID: [non-existing ID] | Display an error message indicating employee not found | Pass |
| 3 | Verify deleting an employee with associated data | Employee ID: [valid ID] | Display a warning message indicating associated data | Pass |

**Unit Testing 19:** Add Attendance

**Testing Objective:** To ensure the add attendance is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify adding attendance for an existing employee | Employee ID: [valid ID] Date: [valid date] | Attendance is successfully added | Pass |
| 2 | Verify adding attendance for a non-existing employee | Employee ID: [non-existing ID] Date: [valid date] | Display an error message indicating employee not found | Pass |
| 3 | Verify adding attendance with an invalid date | Employee ID: [valid ID] Date: [invalid date] | Display an error message indicating invalid date | Pass |

**Unit Testing 20:** View attendance

**Testing Objective:** To ensure the view attendance is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify viewing attendance for a specific employee | Employee ID: [valid ID] | Attendance details are displayed correctly | Pass |
| 2 | Verify viewing attendance for all employees | No specific attributes | List of all attendance records is displayed | Pass |
| 3 | Verify viewing attendance for a non-existing employee | Employee ID: [non-existing ID] | Display an error message indicating employee not found | Pass |

**Unit Testing 21:** Update Attendance

**Testing Objective:** To ensure the update attendance is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify updating attendance for an existing employee | Employee ID: [valid ID] Date: [valid date] New Status: [valid status] | Attendance status is successfully updated | Pass |
| 2 | Verify updating attendance for a non-existing employee | Employee ID: [non-existing ID] Date: [valid date] New Status: [valid status] | Display an error message indicating employee not found | Pass |
| 3 | Verify updating attendance with an invalid status | Employee ID: [valid ID] Date: [valid date] New Status: [invalid status] | Display an error message indicating invalid status | Pass |

**Unit Testing 22:** Payroll

**Testing Objective:** To ensure the payroll is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify generating payroll for an existing employee | Employee ID: [valid ID] | Payroll details are successfully generated | Pass |
| 2 | Verify generating payroll for a non-existing employee | Employee ID: [non-existing ID] | Display an error message indicating employee not found | Pass |
| 3 | Verify generating payroll with invalid data | Employee ID: [valid ID] Payroll details: [invalid data] | Display an error message indicating invalid data | Pass |

**Unit Testing 23:** sale trails

**Testing Objective:** To ensure the view sale trails is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify viewing sale trails for a specific sale | sale ID: [valid ID] | Sale trails for the product are displayed correctly | Pass |
| 2 | Verify viewing sale trails for all sale | Sale IDs | List of sale trails for all products is displayed | Pass |
| 3 | Verify viewing sale trails for a non-existing sale | sale ID: [non-existing ID] | Display an error message indicating product not found | Pass |

**Unit Testing 24:** Vendor trails

**Testing Objective:** To ensure the view vendor trails is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify viewing vendor trails for a specific purchase | Purchase ID: [valid ID] | Vendor trails are displayed correctly | Pass |
| 2 | Verify viewing vendor trails for all purchase | Purchase IDs | List of vendor trails for all purchases are displayed | Pass |
| 3 | Verify viewing vendor trails for a non-existing purchase | Purchase ID: [non-existing ID] | Display an error message indicating vendor not found | Pass |

**Unit Testing 25:** General Ledger

**Testing Objective:** To ensure the general ledger is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify viewing general ledger | Journal entry, Amount, date , description | General ledger entries for the specified time period are displayed correctly | Pass |

**Unit Testing 26:** Payables

**Testing Objective:** To ensure the payables is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify viewing payables for a specific purchase | Journal entry, Amount, date ,description | Payables for the vendor are displayed correctly | Pass |
| 2 | Verify viewing payables for all unpaid purchase | Journal entry, Amount, date ,description | List of payables for all vendors is displayed | Pass |
| 3 | Verify viewing payables for a non-existing purchase | Journal entry, Amount, date ,description | Display an error message indicating vendor not found | Pass |

**Unit Testing 27:** Receivables

**Testing Objective:** To ensure the receivables is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify viewing receivables for a specific unpaid sale | Journal entry, Amount, date ,description | Receivables for the sale are displayed correctly | Pass |
| 2 | Verify viewing receivables for all unpaid sales | Journal entry, Amount, date ,description | List of receivables for all sales is displayed | Pass |
| 3 | Verify viewing receivables for a non-existing sale | Journal entry, Amount, date ,description | Display no data | Pass |

**Unit Testing 28:** Profit/Loss

**Testing Objective:** To ensure the calculation of profit/loss is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify calculating profit/loss for a specific month | Month:[valid month] | Profit/loss for the specified time period is calculated and displayed correctly | Pass |
| 2 | Verify calculating overall profit/loss | No specific attributes | Overall profit/loss is calculated and displayed correctly | Pass |

**Unit Testing 29:** Prediction of Profit

**Testing Objective:** To ensure the prediction of profit is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify predicting profit using valid data | profit data: [valid data] | Profit prediction is generated and displayed correctly | Pass |
| 2 | Verify predicting profit with incomplete data | profit data: [incomplete data] | Display an error message indicating incomplete data | Pass |
| 3 | Verify predicting profit with invalid data | profit data: [invalid data] | Display an error message indicating invalid data | Pass |

**Unit Testing 30:** Logout

**Testing Objective:** To ensure the logout is working correctly

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test case/Test script** | **Attribute and value** | **Expected result** | **Result** |
| 1 | Verify logging out from the system | No specific attributes | User is successfully logged out | Pass |
| 2 | Verify attempting to access secured pages after logout | No specific attributes | Display a login page, indicating unauthorized access | Pass |
| 3 | Verify attempting to perform actions after logout | No specific attributes | Display a login page, indicating unauthorized access | Pass |

* + 1. **Functional Testing**

The functional testing will take place after the unit testing. In this functional testing, the functionality of each of the module is tested. This is to ensure that the system produced meets the specifications and requirements.

* **Functional Testing 1:** Inventory Module

**Objective**: To ensure that the inventory module of the system is working correctly with all the functionalities

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| 1.1 | View Products | - | Successfully view the list of products. | Pass |
| 1.2 | View Vendors | - | Successfully view the list of vendors. | Pass |
| 1.3 | Vendor Payment Status | Vendor ID - V001 | View the payment status of the specified vendor. | Pass |
| 1.4 | Add Product | Product Name - NewProduct, Quantity - 50 | Successfully add a new product to the inventory. | Pass |
| 1.5 | Add Vendor | Vendor Name - NewVendor, Contact - VendorContact | Successfully add a new vendor to the system. | Pass |
| 1.6 | Update Product | Product ID - P001, New Quantity - 100 | Successfully update the quantity of an existing product. | Pass |
| 1.7 | Update Vendor | Vendor ID - V002, New Contact - UpdatedContact | Successfully update the contact information of an existing vendor. | Pass |
| 1.8 | Delete Product | Product ID - P002 | Successfully delete an existing product from the inventory. | Pass |
| 1.9 | Delete Vendor | Vendor ID - V003 | Successfully delete an existing vendor from the system. | Pass |

* **Functional Testing 2:** Customers Module

**Objective**: To ensure that the customers module of the system is working correctly with all the functionalities

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| 2.1 | View Customers | - | Successfully view the list of customers. | Pass |
| 2.2 | View Sales | - | Successfully view the list of sales transactions. | Pass |
| 2.3 | Sale Payment Status | Sale ID - S001 | View the payment status of the specified sale. | Pass |
| 2.4 | Add Customer | Customer Name - NewCustomer, Contact - CustomerContact | Successfully add a new customer to the system. | Pass |
| 2.5 | Add Sale | Product ID - P003, Customer ID - C001, Quantity - 10 | Successfully add a new sale transaction to the system. | Pass |
| 2.6 | Update Customer | Customer ID - C002, New Contact - UpdatedContact | Successfully update the contact information of an existing customer. | Pass |
| 2.7 | Update Sale | Sale ID - S002, New Quantity - 20 | Successfully update the quantity of an existing sale transaction. | Pass |
| 2.8 | Delete Customer | Customer ID - C003 | Successfully delete an existing customer from the system. | Pass |
| 2.9 | Delete Sale | Sale ID - S003 | Successfully delete an existing sale transaction from the system. | Pass |

* **Functional Testing 3:** Employees Module

**Objective**: To ensure that the employees module of the system is working correctly with all the functionalities

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| 3.1 | Add Employees | Employee Name - NewEmployee, Role - Manager | Successfully add a new employee with the specified role. | Pass |
| 3.2 | Add Attendance | Employee ID - E001, Date - 2023-03-01, Status - Present | Successfully record the attendance of an employee for a specific date. | Pass |
| 3.3 | View Employees | - | Successfully view the list of employees. | Pass |
| 3.4 | View Attendance | Employee ID - E002 | Successfully view the attendance record of a specific employee. | Pass |
| 3.5 | Update Employees | Employee ID - E003, New Role - Supervisor | Successfully update the role of an existing employee. | Pass |
| 3.6 | Update Attendance | Attendance ID - A001, New Status - Absent | Successfully update the attendance status for a specific record. | Pass |
| 3.7 | Payroll | Employee ID - E004, Month - March 2023 | Successfully generate the payroll for an employee for the specified month. | Pass |
| 3.8 | Delete Employees | Employee ID - E005 | Successfully delete an existing employee from the system. | Pass |

* **Functional Testing 4:** Finance Module

**Objective**: To ensure that the finance module of the system is working correctly with all the functionalities

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4.1 | View Sales Trails | - | Successfully view the sales trails, including transaction details. | Pass |
| 4.2 | View Vendor Trails | - | Successfully view the vendor trails, including transaction details. | Pass |
| 4.3 | General Ledger | - | Successfully view the general ledger, including all financial transactions. | Pass |
| 4.4 | Payables | Vendor ID - V001, Amount - $500 | Successfully record a payable transaction for a specific vendor. | Pass |
| 4.5 | Receivables | Customer ID - C001, Amount - $700 | Successfully record a receivable transaction for a specific customer. | Pass |
| 4.6 | Profit/Loss | - | Successfully view the overall profit/loss statement for the business. | Pass |
| 4.7 | Profit Prediction | Month - April 2023 | Successfully predict the profit for the specified month using the Prophet Model. | Pass |

* + 1. **Integration Testing**
* Integration Testing 1: To ensure the Customer Module integrated with Inventory Module and Finance Module is working correctly.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| 1.1 | Add Customer and Add Sale in Customer Module | - | Successfully add a new customer and record a sale transaction. Check if the customer information is updated in the Finance Module. | Pass |
| 1.2 | View Customer in Inventory Module | - | View customer details from the Inventory Module, ensuring data consistency between modules. | Pass |
| 1.3 | View Customer Trails in Finance Module | - | View the trails in the Finance Module to verify that customer-related transactions are correctly logged. | Pass |

* Integration Testing 2: To ensure the Product Module integrated with Finance Module and Customers Module is working correctly.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| 2.1 | Add Product and Add Sale in Product Module | - | Successfully add a new product and record a sale transaction. Check if the product information is updated in the Finance Module. | Pass |
| 2.2 | View Product in Customers Module | - | View product details from the Customers Module, ensuring data consistency between modules. | Pass |
| 2.3 | View Product Trails in Finance Module | - | View the trails in the Finance Module to verify that product-related transactions are correctly logged. | Pass |

* Integration Testing 3: To ensure the Employees Module with Finance Module is working correctly.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Test Case/Test Script** | **Attribute and Value** | **Expected Result** | **Result** |
| 3.1 | Add Employee and Record Payroll in Employees Module | - | Successfully add a new employee and record payroll. Check if the employee payroll information is updated in the Finance Module. | Pass |
| 3.2 | View Employee Trails in Finance Module | - | View the trails in the Finance Module to verify that employee-related transactions and payroll records are correctly logged. | Pass |

* 1. **Automated Testing:**
     1. **Tools used:**

The following table shows the tools used in automated testing

*Tools used*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1. | Thunder Client | A REST client extension for Visual Studio Code. | API Testing - REST APIs and Flask API | Pass |

1. **Conclusion**

The development and implementation of the ERP System represent a significant milestone in addressing the organizational needs for streamlined processes, efficient data management, and improved decision-making. The adoption of a Client-Server Architecture, component-based design, and agile methodologies has contributed to the creation of a robust and flexible system.

The ERP System's comprehensive modules, including Inventory, Customers, Employees, and Finance, ensure that key aspects of business operations are seamlessly integrated.

The testing phase, encompassing manual and automated testing, has been instrumental in ensuring the system's reliability, functionality, and performance. Through unit, functional, and integration testing, potential issues were identified and addressed, guaranteeing a high-quality software product.

As the ERP System moves from development to deployment, its success will be measured by its impact on organizational efficiency and productivity. User feedback and continuous improvement will be essential in refining the system's performance and addressing any emerging requirements.

1. **Future Work**

**User Feedback and Iterative Improvements**

Collect user feedback after deployment and use it to iteratively enhance the user experience and address any usability issues.

**Enhanced Reporting and Analytics**

Integrate advanced reporting and analytics features to provide stakeholders with deeper insights into business performance.

**Scalability and Performance Optimization**

Monitor system performance and optimize for scalability, especially as user demands and data volumes increase over time.

**AI and Machine Learning Integration**

Explore the integration of AI and machine learning algorithms for predictive analytics and intelligent decision support.

1. **References**

Ijcsis, J. O. C. S. (2019). *Cloud based ERP System for SME Industry*. Journal of Computer Science IJCSIS - Academia.edu. <https://www.academia.edu/41681836/Cloud_based_ERP_System_for_SME_Industry>

Spathis, C., & Constantinides, S. (2003). The usefulness of ERP systems for effective management. *Industrial Management and Data Systems*, *103*(9), 677–685. <https://doi.org/10.1108/02635570310506098>

Buleje, M. (2014). The Impact of Enterprise Resource Planning Systems on Small and Medium Enterprises (Doctoral dissertation). Nova Southeastern University. <https://nsuworks.nova.edu/gscis_etd/108>