**COMSATS University Islamabad,   
Abbottabad Campus**

**SOFTWARE DESIGN DESCRIPTION   
(SDD DOCUMENT)**

**for**

**<PROJECT NAME>**  
Version 1.0

***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 Computer Science (20xx-20xx)***

**Table of Contents**

**Revision History 3**

**1.** **Introduction 5**

**2.** **Design Methodology and software process model 5**

**3.** **System Overview 5**

3.1 Architectural Design 5

3.2 Process Flow/Representation 5

**4.** **Design Models [along with descriptions] 5**

**5.** **Data Design 6**

5.1 Data Dictionary 6

**6.** **Algorithm & Implementation 6**

**7.** **Software Requirements Traceability Matrix 6**

**8.** **Human Interface Design 7**

8.1 Screen Images 7

8.2 Screen Objects and Actions 7

**9.** **Appendix I 7**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason for changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

**Application Evaluation History**

|  |  |
| --- | --- |
| **Comments (by committee)**  **\*include the ones given at scope time both in doc and presentation** | **Action Taken** |
|  |  |
|  |  |

**Supervised by**

**Dr. Kashif Nasr**

Signature\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Introduction**

The scope of the project covered till now includes the development of the User Authentication and Inventory Management System module within the ERP System.

**User Authentication:**

The project shall include the development of a login page that will allow users to securely access the ERP system.

The system will implement user authentication and authorization mechanisms that will ensure secure access to different system modules and functionalities.

Integration with user management functionality will be implemented to manage different users access to the relevant data.

Each user's authentication token will be associated with specific data in the database, providing personalized access to relevant information.

**Inventory Module:**

The project will include the development of a feature that will allow users to add new products to the inventory.

The system shall capture relevant product details such as name, description, quantity, price, and other attributes as required.

Integration with the database will be implemented to store product information accurately and associate it with the user's authentication token.

The system will provide a feature to view the list of available products in the inventory.

Product information, including details such as name, description, quantity, and price, will be displayed based on the user's authentication token.

Integration with the database will retrieve and present only the relevant product data associated with the user.

The system shall allow users to delete selected products from the inventory.

Integration with the database will enable secure removal of product records associated with the user's authentication token.

The project will include a feature to update existing product information.

Users will be able to modify product details such as quantity, price, and attributes associated with their authentication token.

Integration with the database will ensure accurate updating of product information specific to the user.

**Additional Modules (Future Expansion):**

The ERP system will be designed and developed to accommodate future expansion with additional modules.

Each module will be tailored to provide personalized access to data based on the user's authentication token.

User-specific data will be securely managed and associated with the relevant modules to ensure appropriate access and functionality.

**Design methodology and software process model**

**Choice of Design Methodology**

For the development of the ERP system, the chosen design methodology is Function-Based Programming with React. Function-based programming is a paradigm that emphasizes the use of pure functions and functional composition to design software systems. The decision to use function-based programming with React is justified by the following reasons:

**Simplicity and Modularity:** Function-based programming with React provides a simpler and more concise syntax compared to class-based components. It promotes modularity by allowing developers to create reusable and composable functions that can be easily composed together to build complex UI components. This modularity improves code organization and maintainability.

**Functional Composition:** Function-based programming encourages the composition of small, focused functions to achieve complex behaviors. React's functional components and hooks facilitate this composability, allowing developers to build reusable and testable code by combining smaller functions. This approach simplifies code understanding, debugging, and future enhancements.

**Reusability and Scalability:** Function-based programming with React promotes the development of reusable components. By encapsulating logic and state within functions, these components can be easily shared and reused across different parts of the system or in future development phases. This reusability improves development efficiency and scalability.

**Functional Purity:** Pure functions, which are a fundamental aspect of function-based programming, have no side effects and always produce the same output for the same input. React's functional components encourage pure functions, which improves predictability and testability. It also helps prevent unexpected bugs and simplifies the debugging process.

**Process Model**

The chosen process model for this project is Agile development, specifically Scrum. Scrum is an iterative and incremental approach that promotes flexibility, collaboration, and adaptability. The decision to follow Scrum is justified by the following reasons:

**Iterative Development:** Scrum divides the project into short iterations called sprints. Each sprint focuses on delivering a working increment of the ERP system. This iterative approach allows for regular feedback from stakeholders and enables continuous improvement. It also facilitates early detection and resolution of issues.

**Flexibility and Adaptability:** Scrum provides flexibility in terms of changing requirements and priorities. As ERP systems often involve evolving business needs, Scrum's adaptive nature allows for adjusting the project scope and incorporating changes in an incremental manner. This helps ensure that the final system meets the evolving requirements of the organization.

**Collaboration and Communication:** Scrum emphasizes collaboration and regular communication among team members, stakeholders, and end-users. Daily stand-ups, sprint planning, and sprint reviews facilitate effective communication, feedback, and alignment of expectations. This collaborative approach ensures that the developed ERP system meets the specific needs and requirements of the organization.

**Risk Management**

Scrum provides a framework for identifying and managing risks early in the development process. By breaking the project into short sprints, risks can be identified and addressed promptly, reducing the likelihood of major setbacks. This proactive risk management approach minimizes project risks and increases the chances of successful delivery.

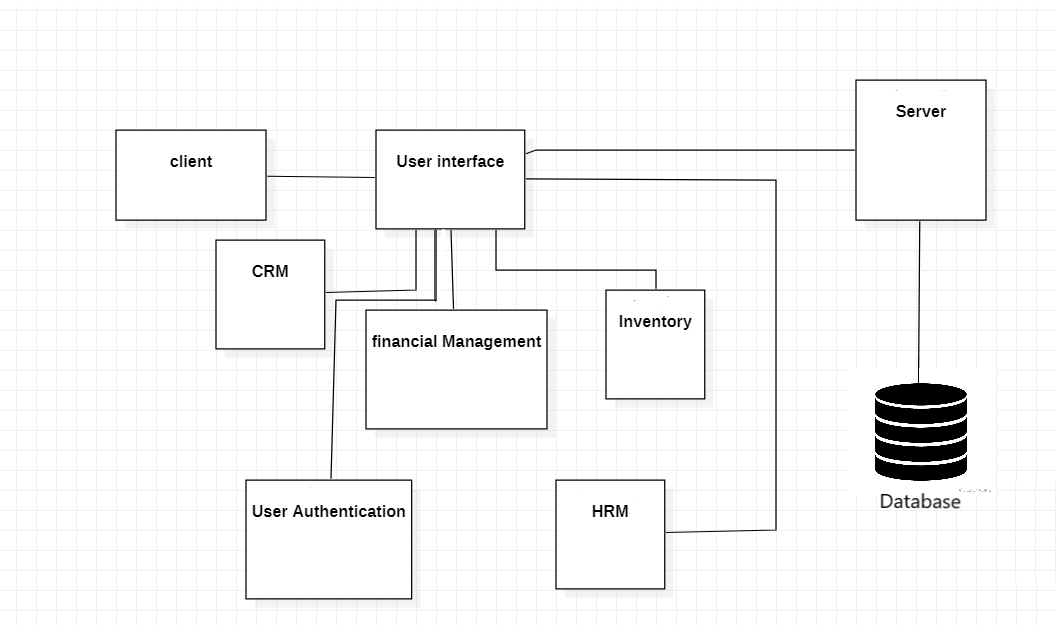
**System overview**

The ERP system is developed using a function-based programming approach with React for frontend development. It includes modules such as user authentication, inventory management, and future expandability.

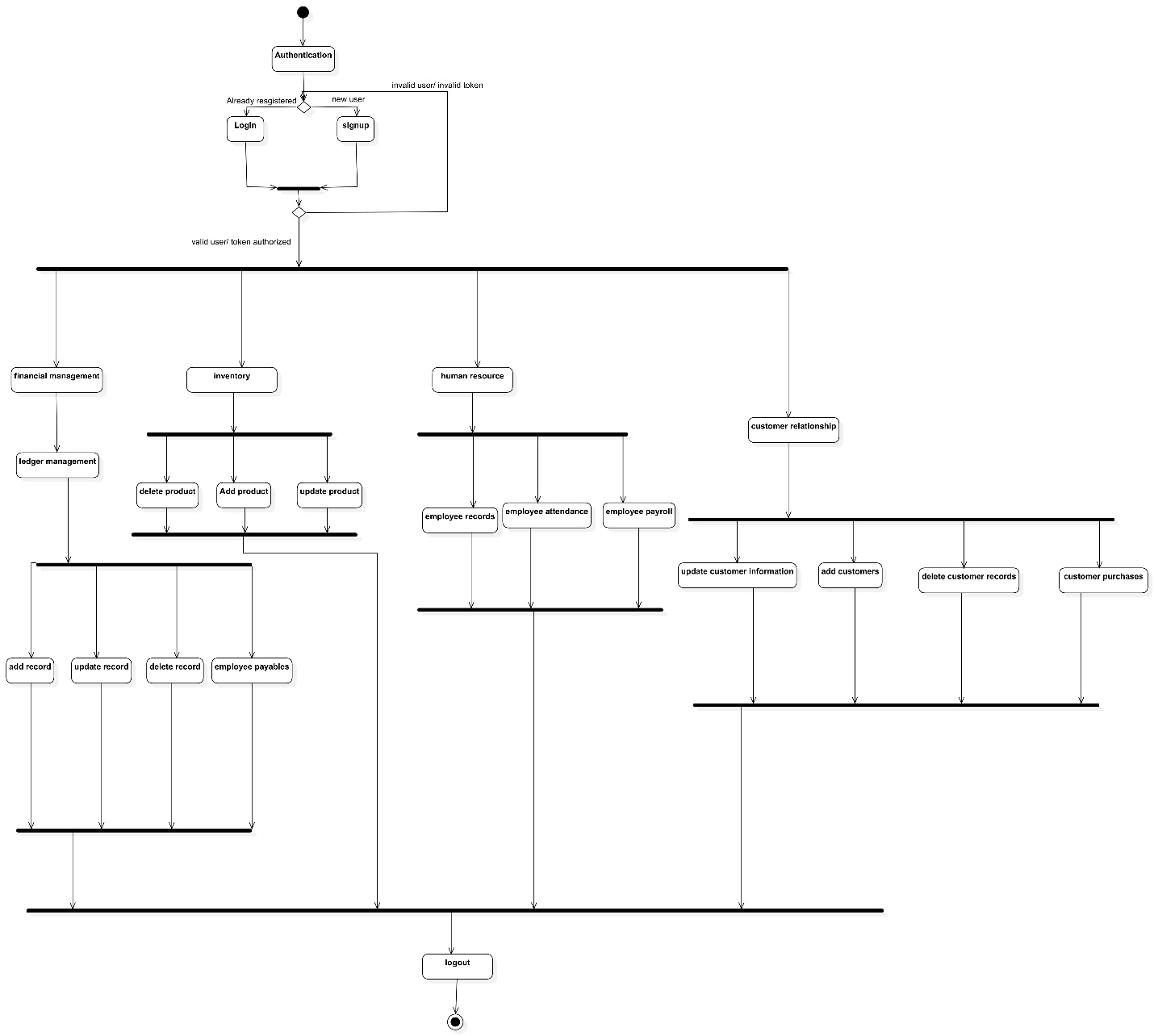
The system leverages the Scrum agile methodology for iterative development, promoting collaboration, flexibility, and risk management. The combination of function-based programming and Scrum ensures a modular, scalable, and adaptable ERP system that enhances operational efficiency and decision-making capabilities.

**Architectural design**

**Client-Server Architecture**

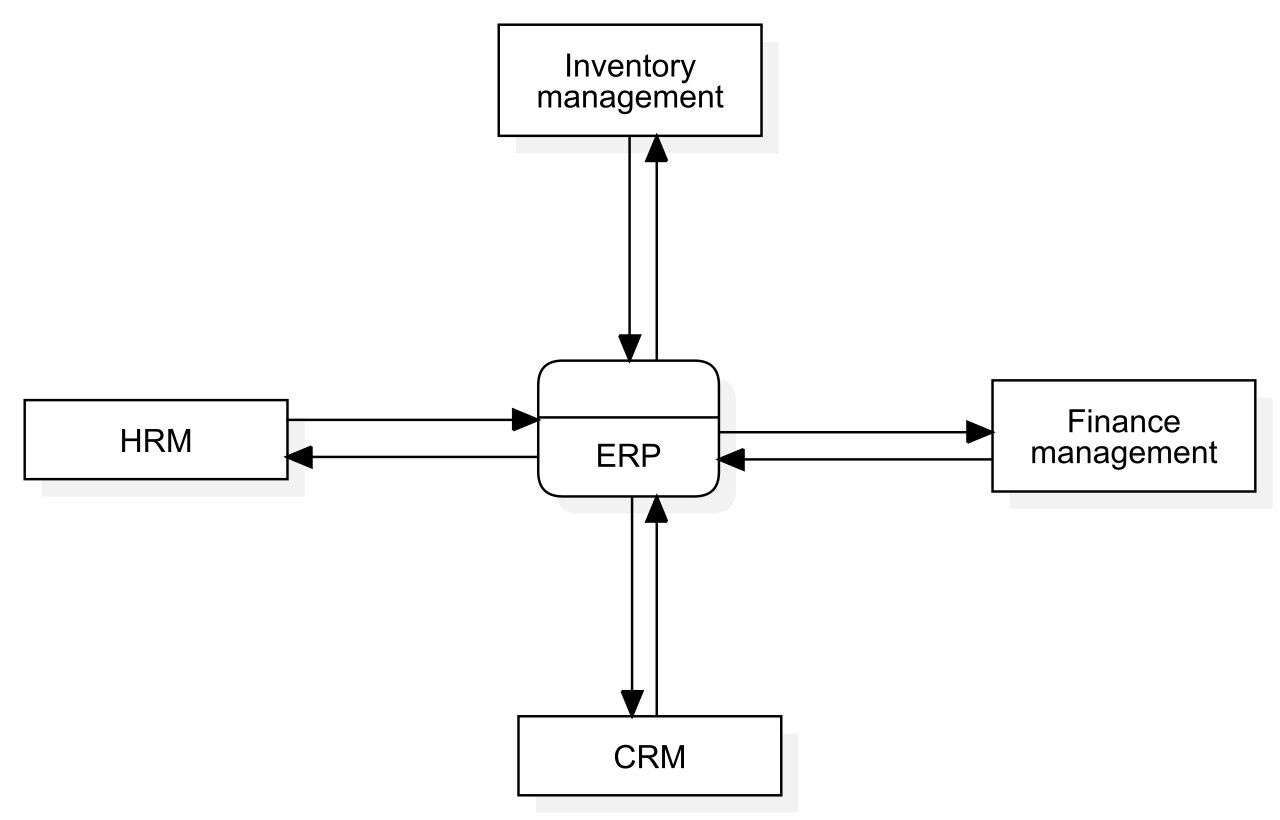


**Process flow/Representation**

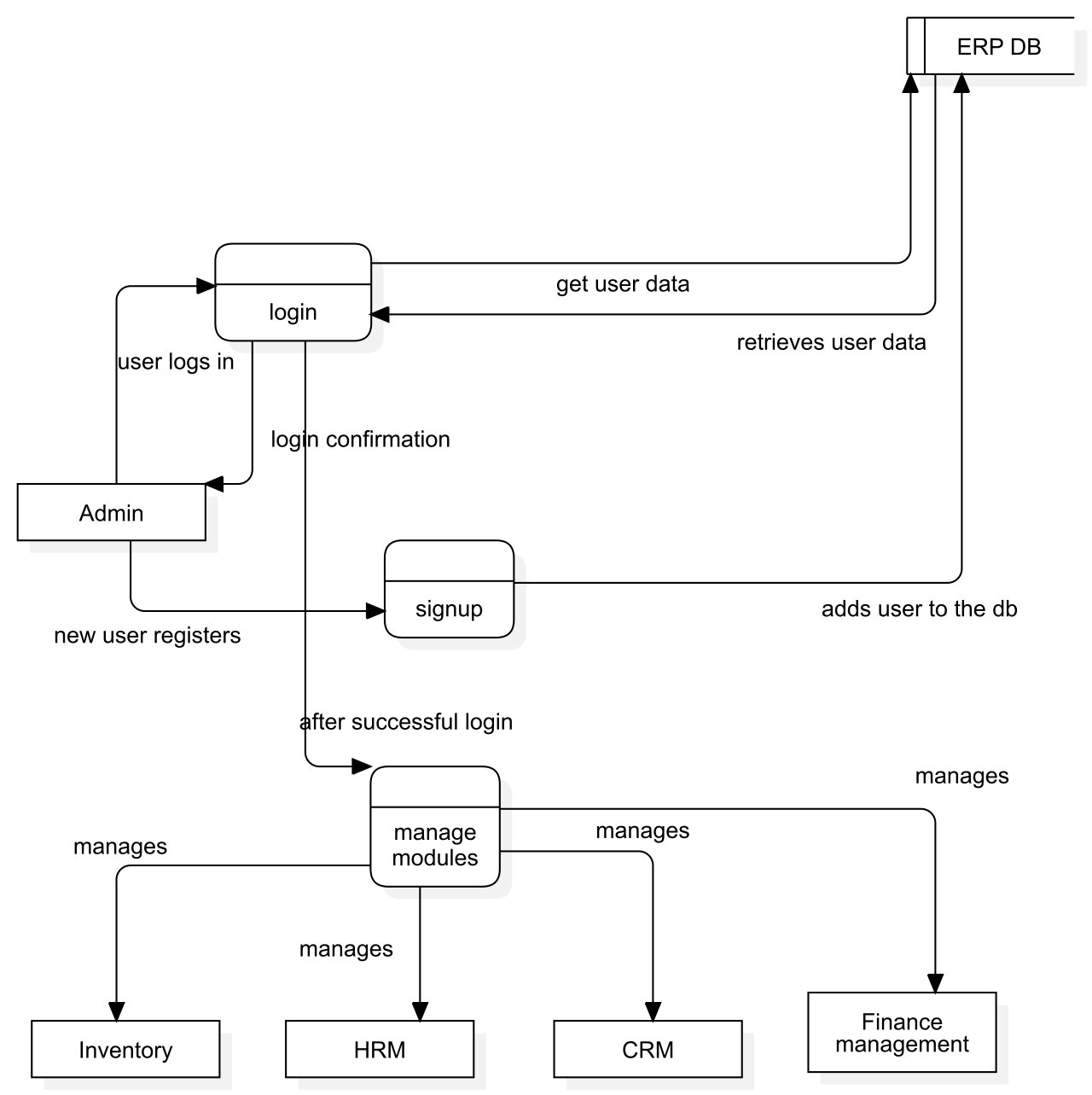
****

**Design models**

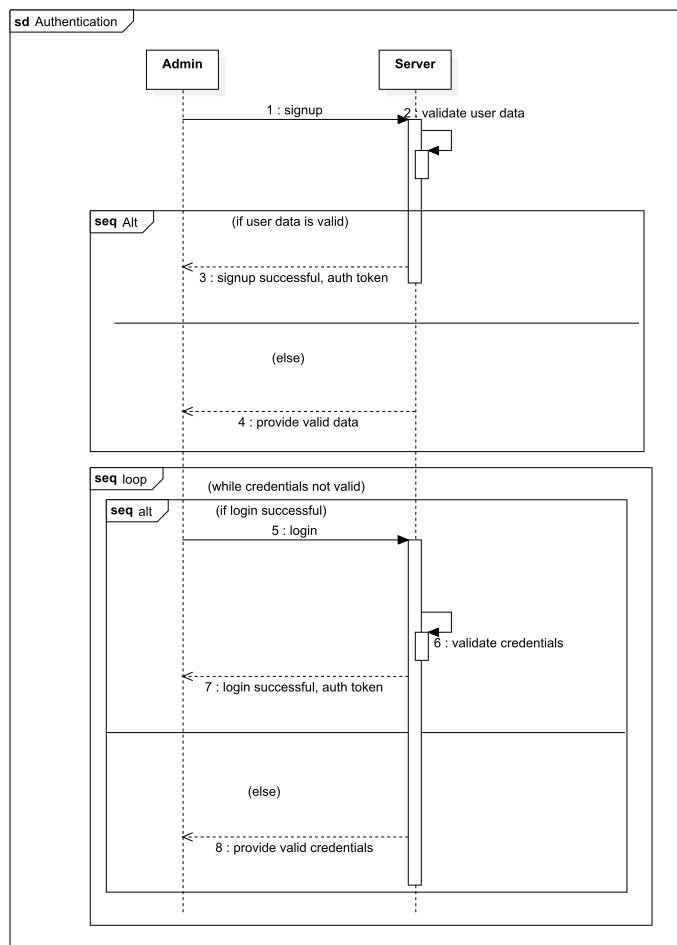
**Level 0 DFD**

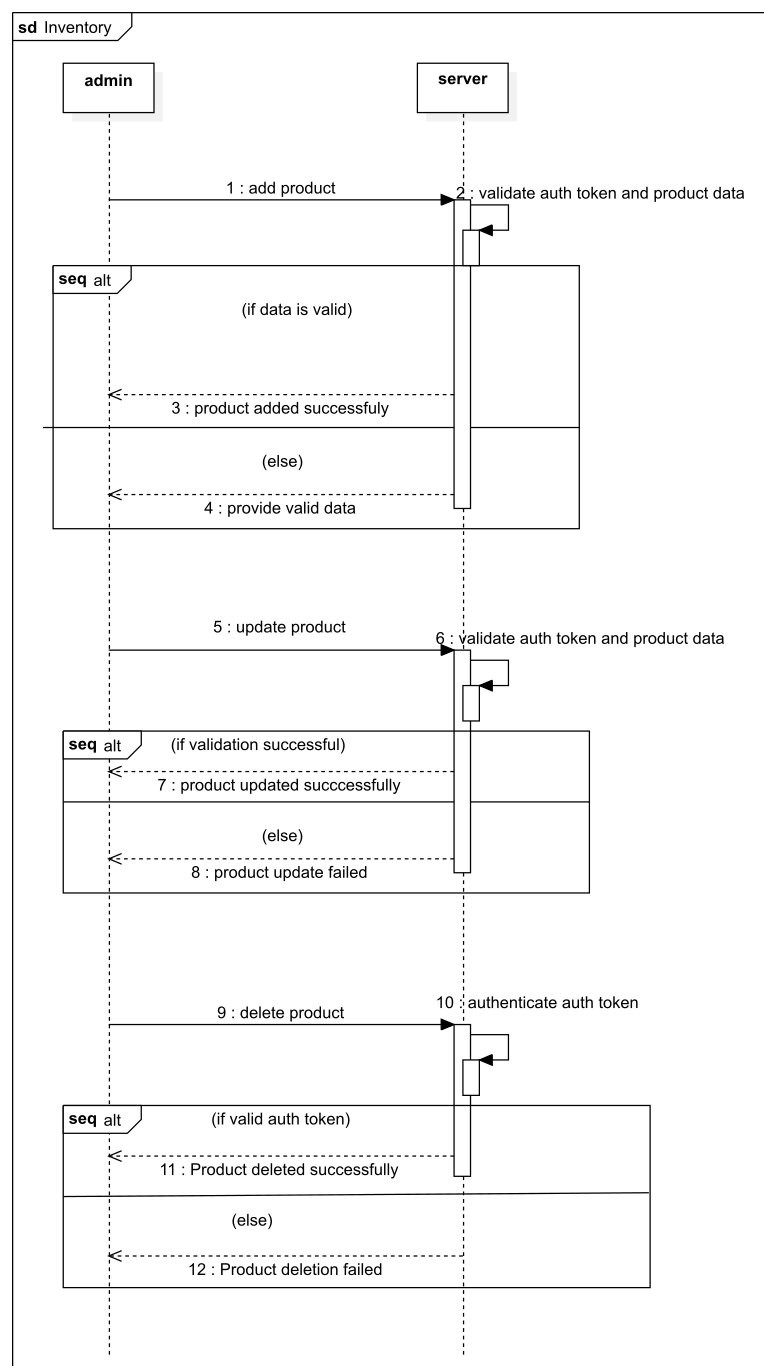


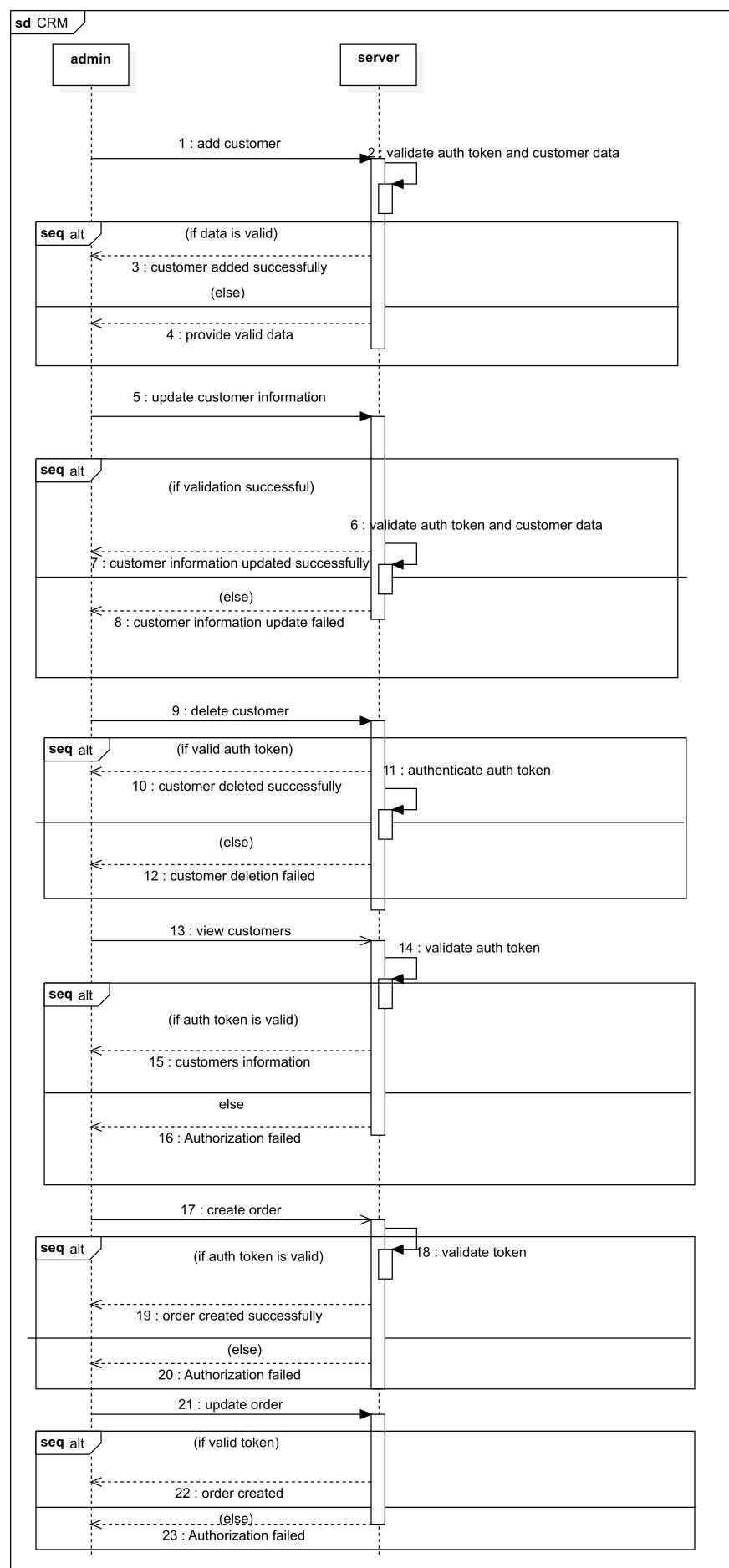
**Level 1 DFD**



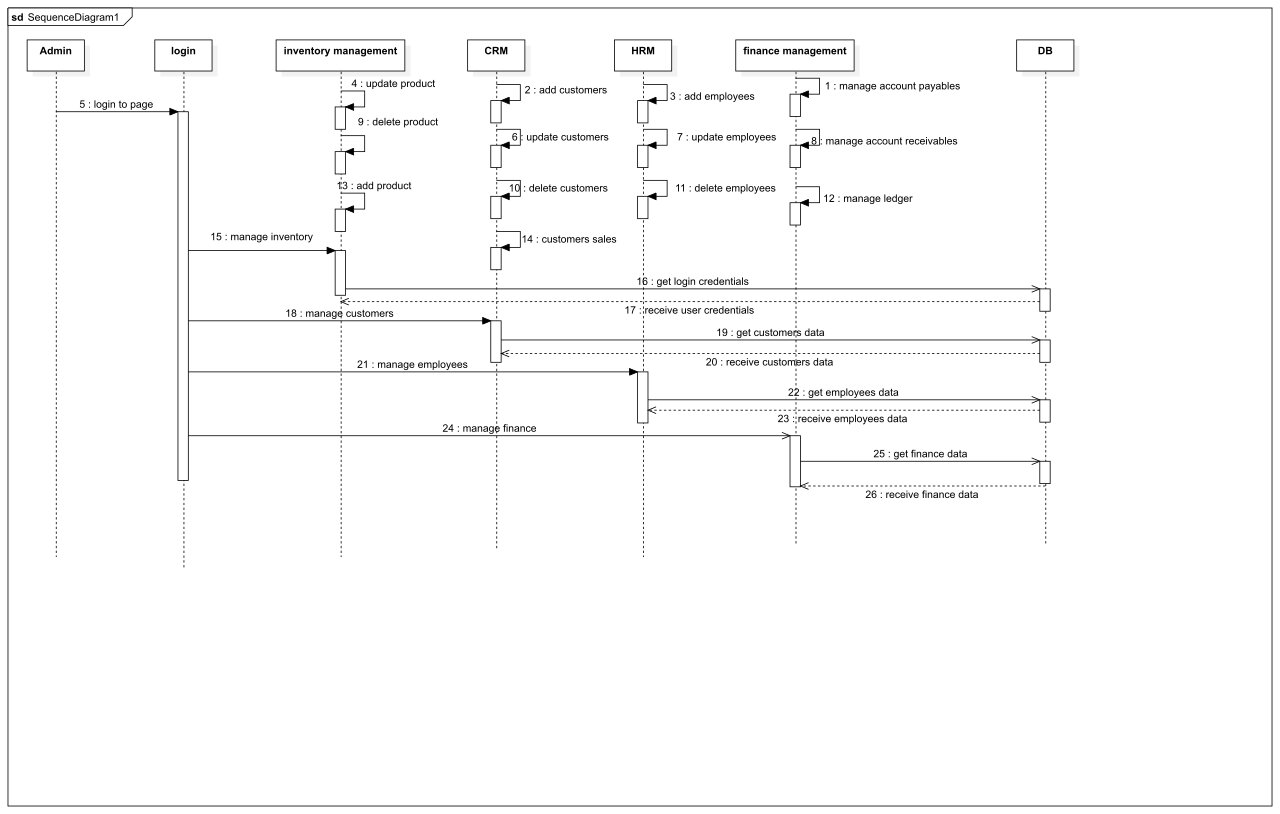
**System Sequence Diagrams**



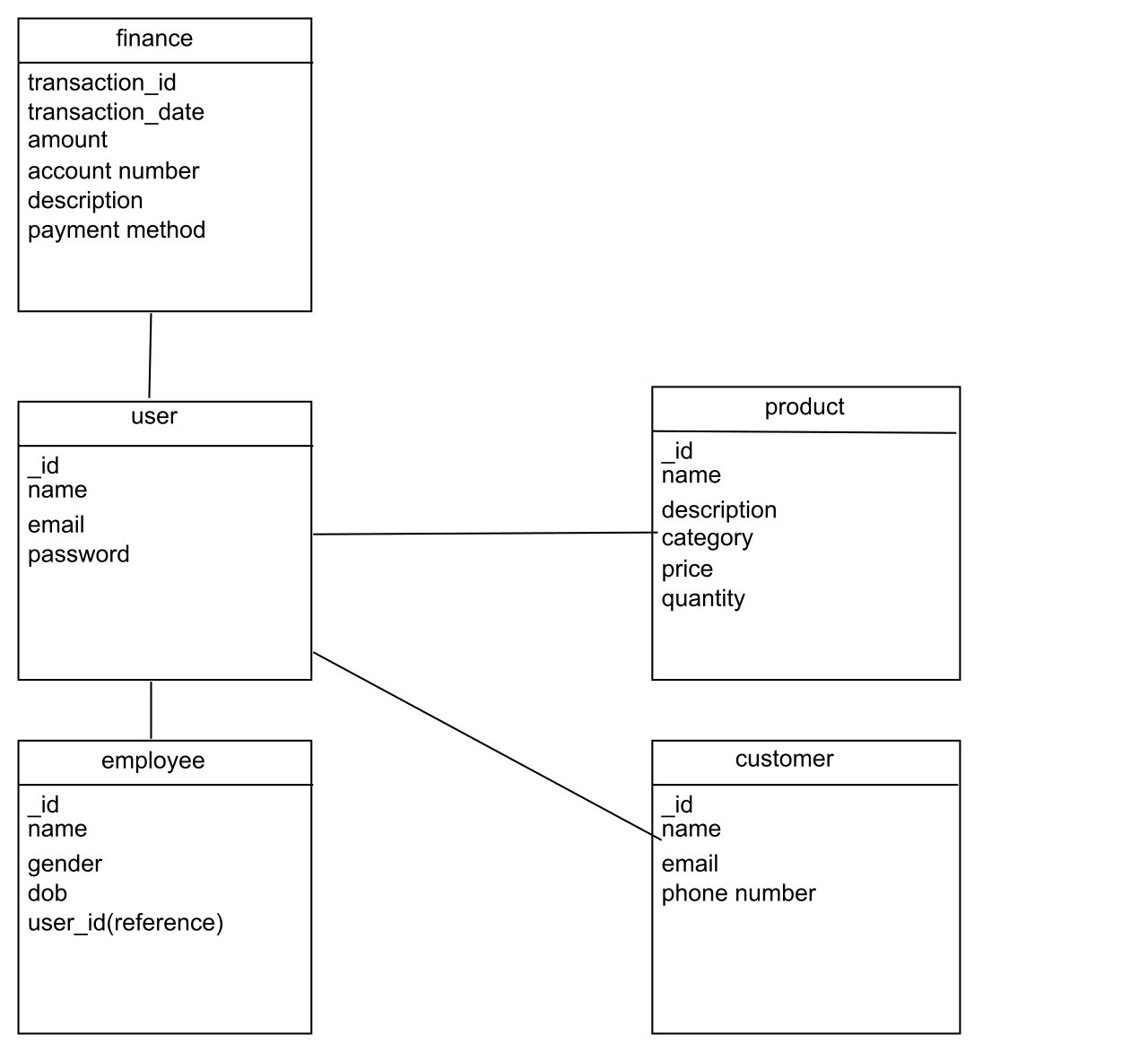




**Sequence Diagram**



**Document Schema Diagram**



**Data design**

As we are using MongoDB as database technology for the ERP system, the data structures and organization will differ from a relational database. MongoDB is a document-oriented NoSQL database, which means it stores data in flexible, self-contained documents rather than rigid tables with fixed schemas.

Here's an explanation of how the major data or system entities can be stored, processed, and organized in MongoDB:

* MongoDB uses collections to store data, similar to tables in relational databases.
* Each collection contains documents, which represent instances of entities like customers, products, or orders.
* Documents are flexible and can have varying structures, allowing for nested fields and arrays.
* Unique identifiers, such as the "\_id" field, ensure the uniqueness of each document within a collection.
* Indexing improves query performance by creating indexes on specific fields.
* MongoDB provides a powerful query language for flexible data retrieval and an aggregation framework for complex data analysis.
* Data integrity can be maintained through validation rules defined at the document level.
* Security features like authentication, access control, and encryption protect the data.
* MongoDB can integrate with other systems through data imports, exports, and APIs.
* Data design in MongoDB emphasizes the document-oriented model, enabling flexible data structures and efficient data processing.

**Data dictionary**

**Here's the alphabetical list of ERP system entities:**

1. **User:**

* Type: Entity
* Description: Represents a user of the ERP System.

Attributes:

* UserID: Unique identifier for the user.
* Username: User's username for authentication.
* Email: User's email address.
* Password: User's password (encrypted).
* Role: User's role within the ERP System (e.g., administrator).

1. **Customers:**

* Type: Entity
* Description: Represents the customers of the company.

Attributes:

* CustomerID: Unique identifier for the customer.
* Name: Name of the customer.
* ContactNumber: Contact number of the customer.
* Email: Email address of the customer.
* Address: Address of the customer.
* CreditLimit: Credit limit for the customer.

1. **Employees:**

* Type: Entity
* Description: Represents the employees of the company.

Attributes:

* EmployeeID: Unique identifier for the employee.
* Name: Name of the employee.
* ContactNumber: Contact number of the employee.
* Email: Email address of the employee.
* Position: Job position or role of the employee.
* Salary: Salary or compensation for the employee.

1. **Financial Transactions:**

* Type: Entity
* Description: Represents financial transactions recorded in the ERP System.

Attributes:

* TransactionID: Unique identifier for the transaction.
* Date: Date of the transaction.
* Amount: Monetary value of the transaction.
* Type: Type of the transaction (e.g., income, expense).
* Description: Description or details of the transaction.

1. **Invoices:**

* Type: Entity
* Description: Represents invoices generated for customers.

Attributes:

* InvoiceID: Unique identifier for the invoice.
* CustomerID: Identifier of the customer associated with the invoice.
* Date: Date of the invoice.
* TotalAmount: Total amount due in the invoice.
* Status: Status of the invoice (e.g., paid, unpaid).

1. **Inventory:**

* Type: Entity
* Description: Represents the inventory of products.

Attributes:

* ProductID: Unique identifier for the product.
* Name: Name of the product.
* Description: Description of the product.
* Quantity: Quantity of the product in stock.
* Price: Price of the product.

1. **Orders:**

* Type: Entity
* Description: Represents customer orders.

Attributes:

* OrderID: Unique identifier for the order.
* CustomerID: Identifier of the customer who placed the order.
* Date: Date of the order.
* Status: Status of the order (e.g., processing, shipped).

1. **Products:**

* Type: Entity
* Description: Represents the products available in the inventory.

Attributes:

* ProductID: Unique identifier for the product.
* Name: Name of the product.
* Description: Description of the product.
* Price: Price of the product.

**Algorithm & Implementation**

1. **Inventory Management:**
   * addProduct(productData): Adds a new product to the inventory with the provided product data and saves it in the database.
   * updateProduct(productID, updatedData): Updates the product identified by the product ID with the provided updated data.
   * deleteProduct(productID): Deletes the product with the specified product ID from the database.
2. **Financial Management:**
   * manageGeneralLedger(journalEntryData): Manages the general ledger by creating and updating journal entries based on the provided data.
   * processAccountsPayable(paymentData): Processes accounts payable by creating, tracking, and processing payments to vendors based on the provided payment data.
   * processAccountsReceivable(paymentData): Processes accounts receivable by creating, tracking, and processing customer payments based on the provided payment data.
3. **Customer Relationship Management (CRM):**
   * addCustomer(customerData): Adds a new customer to the CRM database with the provided customer data.
   * updateCustomer(customerID, updatedData): Updates the customer identified by the customer ID with the provided updated data.
   * deleteCustomer(customerID): Deletes the customer with the specified customer ID from the CRM database.
4. **Human Resource Management (HRM):**
   * addEmployee(employeeData): Adds a new employee to the HRM module with the provided employee data.
   * updateEmployee(employeeID, updatedData): Updates the employee identified by the employee ID with the provided updated data.
   * deleteEmployee(employeeID): Deletes the employee with the specified employee ID from the HRM module.
   * trackAttendance(employeeID, attendanceData): Tracks the attendance of an employee by recording their time in and time out based on the provided attendance data.

**Software requirements traceability matrix**

|  |  |  |  |
| --- | --- | --- | --- |
| Req. Number | Ref. Item | Design Component | Component Items |
| FR01 | General Ledger | Ledger | FunctionName(s) |
| FR02 | General Ledger | Ledger | Create Entry, Manage Entry |
| FR03 | Accounts Payable | Accounts Payable System | Create Payment, Track Payment, Process Payment |
| FR04 | Accounts Receivable | Accounts Receivable System | Create Payment, Track Payment, Process Payment |
| FR05 | CRM | Customer Database | Add Customer, Capture Information, Sales History |
| FR06 | CRM | Customer Database | Update Customer Information |
| FR07 | CRM | Customer Database | Delete Customer |
| FR08 | CRM | Sales Management | Manage Customer Sales |
| FR09 | Inventory | Product Management | Add Product, Product Details |
| FR10 | Inventory | Product Management | Update Product Information |
| FR11 | Inventory | Product Management | Delete Product |
| FR12 | HRM | Employee Database | Add Employee, Capture Information |
| FR13 | HRM | Employee Database | Update Employee Information |
| FR14 | HRM | Employee Database | Delete Employee |
| FR15 | HRM | Attendance Management | Track Employee Attendance |

**Human interface design**

The ERP system offers a user-friendly interface for efficient task completion. Users can manage customers, track inventory, handle financial tasks, and access reports.

1. User Registration and Login:

* Users can create an account and log in using their credentials.
* Feedback: The system displays a success message upon successful registration or login. In case of any errors, appropriate error messages are shown.

1. Dashboard:

* Users are presented with a personalized dashboard upon logging in.
* The dashboard provides an overview of key metrics, such as sales, inventory status, pending tasks, and notifications.
* Feedback: The dashboard displays real-time data and visually represents information through charts, graphs, and widgets.

1. Inventory Management:

* Users can add, update, and delete products in the inventory.
* They can track stock levels, set reorder points, and generate reports on inventory status.
* Feedback: The system provides feedback on successful product updates, displays stock availability, and triggers alerts when inventory levels are low.

1. Customer Relationship Management (CRM):

* Users can add new customers, update customer information, and manage customer sales.
* They can view customer details, track interactions, and record sales transactions.
* Feedback: The system provides confirmation messages for successful customer additions and updates. It also displays sales history and customer feedback.

1. Financial Management:

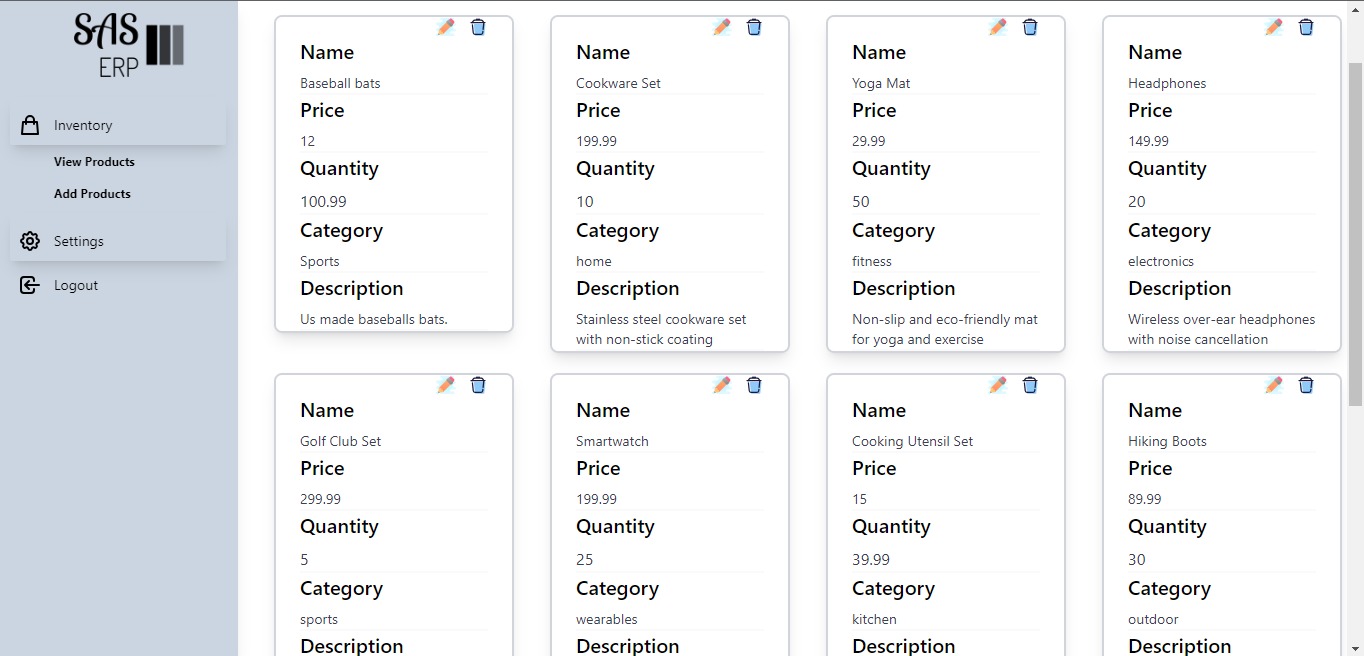
* Users can manage financial transactions, including accounts payable and accounts receivable.
* They can record payments, generate invoices, and track financial data.
* Feedback: The system provides feedback on successful transactions, displays payment receipts, and updates financial records.

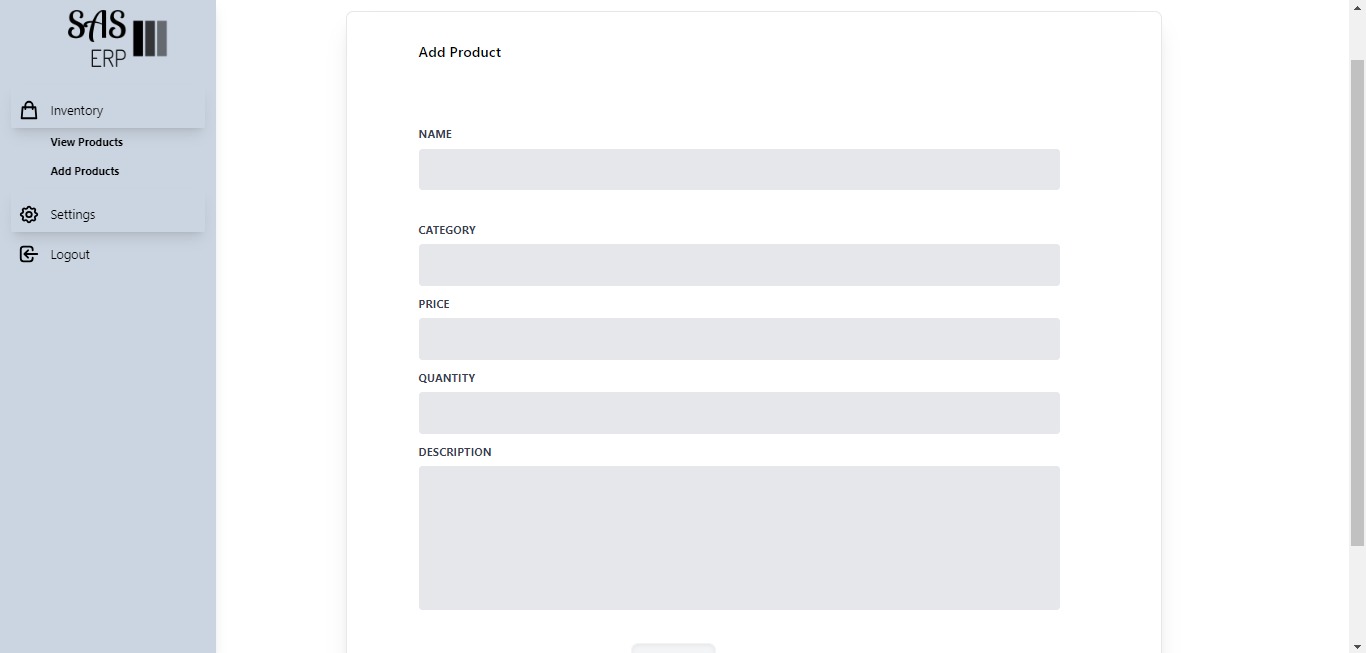
1. Human Resource Management (HRM):

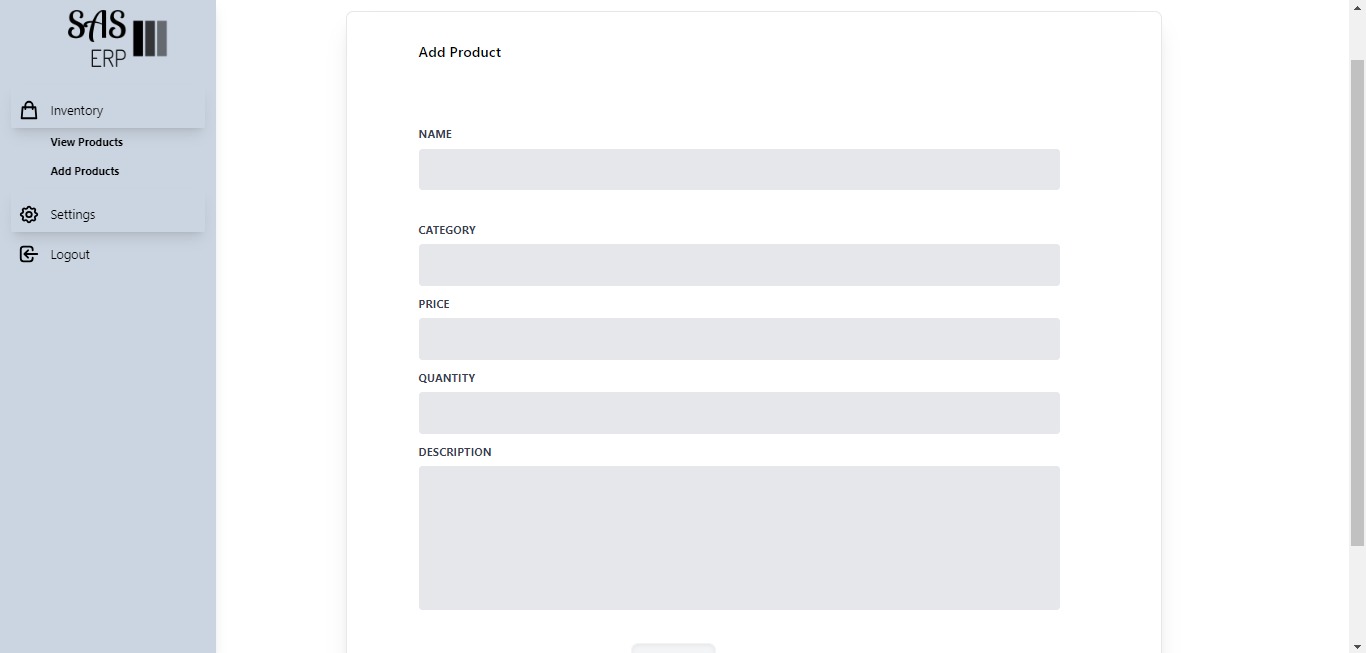
* Users can add, update, and delete employee records.
* They can manage employee attendance, track leave requests, and access HR-related information.
* Feedback: The system provides feedback on successful employee record management, displays attendance records, and notifies users about approved leave requests.

**Screen images**









**Appendix I**

* How to design using UML (OOP): For guidance please follow the instructions mentioned in the link: http://agilemodeling.com/artifacts/
* How and when to design ER diagrams: For guidance please follow the instructions mentioned in the link:

<http://people.inf.elte.hu/nikovits/DB2/Ullman_The_Complete_Book.pdf>

* Data flow diagrams: For guidance please follow the instructions mentioned in the link and book:
  + http://www.agilemodeling.com/artifacts/dataFlowDiagram.htm
  + Software Engineering –A Practitioner’s approach by Roger Pressman
* Architecture diagram: For guidance please follow the instructions mentioned in the link and book:
  + Ian Sommerville – Software Engineering 9th Edition– Chapter 6