IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Fund Management System

Gurjar Jignesh K* , K Ramya
Student*,Assistant Professor,
Department of Computer Engineering,Silver Oak University,Ahmedabad,India

Abstract

The **Fund Management System** is a web-based software solution designed to automate and streamline fund-related operations within an organization or institution. Its primary objective is to ensure efficient **fund utilization,** transparency in financial transactions, and accountability across multiple departments and agencies. The system minimizes manual paperwork, reduces processing delays, and **provides real-time tracking** of fund allocation and expenditure.

The process begins with the creation and approval of a provision grant, which is then allocated by higher authorities to respective **Heads of Departments (HODs).** The HODs further distribute the funds to sub-agencies based on predefined budgets and requirements. Each stage of allocation and utilization is digitally recorded and monitored to maintain financial integrity. Expenditure details, such as vendor payments and employee payrolls, are managed within the system through secure and automated workflows, reinforced by an **OTP-based approval** mechanism to prevent unauthorized transactions.

The application also generates comprehensive **MIS reports** that provide analytical insights into fund flow, utilization, and remaining balances. These reports assist administrators in making informed, data-driven financial decisions while ensuring transparency throughout the organization.

Technically, the system is developed using HTML, CSS, and JavaScript for the front end, with a **.NET Core (MVC) backend and MySQL** database for secure data storage and management. Key features include data encryption, access-level control, and multi-stage validation. Rigorous testing ensures reliability, performance, and user-friendliness.

Overall, the **Fund Management System** represents a significant step toward digital transformation in financial operations by automating fund allocation, monitoring, and reporting. It enhances efficiency, reduces human error, and strengthens financial accountability within institutional frameworks.

Key Words : Fund Management System, fund allocation, financial transparency, automation, real-time tracking, MIS reports, .NET Core MVC, data security.

INTRODUCTION

PROJECT SUMMARY

The **Fund Management System** is a web-based application developed to streamline the process of grant management, fund allocation, expenditure tracking, and beneficiary payment within an organization. The primary objective of this system is to automate the manual processes involved in fund distribution and utilization, ensuring transparency, efficiency, and accountability across all financial operations. The system provides an integrated platform where **departments, head of departments (HODs)**, and agencies can manage the complete lifecycle of fund allocation — from grant creation to payment processing.

The project begins with the Department initiating a grant proposal that is reviewed and approved by the Head of Department (HOD). Once approved, funds are allocated to the relevant **Agency**, which then disburses them to **Vendors or Employees based** on project requirements. The system maintains a digital trail of every transaction, ensuring accuracy and enabling real-time monitoring of fund utilization. This structured workflow minimizes the chances of manual error, duplication, or mismanagement of funds, which are common in traditional paper-based systems.

The User Login module enables secure access through mobile number and **OTP authentication,** ensuring that only authorized users can interact with the system. The Dashboard provides a comprehensive overview of fund allocations, pending approvals, and financial summaries, helping management make informed decisions. Modules such as Provision **Grant Setup, Fund Allocation, Beneficiary Creation, Expenditure Management, and Report Generation** have been developed to handle specific functional areas. Each module interacts with a centralized database to maintain data consistency and streamline cross-departmental communication.

From a technical perspective, the project follows the **Software Development Life Cycle (SDLC)** phases — including requirement analysis, system design, implementation, testing, and deployment. It is developed using .NET Core as the backend framework for its robustness and scalability, with a SQL Server database managing data storage and retrieval. The frontend interface is designed to be user-friendly, responsive, and accessible across devices. Security features such as **role-based authentication,** approval tracking, and audit logs have been integrated to ensure data integrity and secure operations.

The Fund Management System provides numerous benefits such as improved efficiency in fund disbursement, **reduced paperwork**, enhanced accountability, and accurate financial reporting. It also facilitates better decision-making through its reporting and analytics features, allowing administrators to track the performance of each fund and identify potential bottlenecks. By automating routine financial workflows, the system not only saves time but also contributes to better governance and transparency. Overall, this project represents a significant step toward digital transformation in public fund administration and organizational finance management.

PURPOSE

The purpose of developing the Fund Management System is to create a **centralized, automated**, and transparent platform for managing the entire process of fund allocation, utilization, and reporting within an organization or government department. In traditional systems, fund-related activities such as grant creation, approvals, and expenditure tracking are handled manually through paperwork or spreadsheets, which often lead to errors, delays, and lack of accountability. This project aims to overcome these limitations by introducing a structured digital system that ensures accuracy, speed, and reliability in fund management operations.

The system's main purpose is to streamline the workflow among different stakeholders such as the **Department, Head of Department (HOD)**, Agencies, and Vendors/Employees involved in financial transactions. Each stakeholder plays a crucial role in the fund lifecycle — starting from the department creating a grant proposal, followed by the HOD reviewing and approving it, then the agency managing and utilizing the allocated funds, and finally ensuring that the payments reach the concerned vendors or employees. By automating these processes, the system reduces the time taken for approvals and fund transfers while maintaining complete transparency and traceability at every stage.

Furthermore, the purpose extends to ensuring data integrity and security through a role-based authentication mechanism, where each **user type — Maker, Verifier, and Checker** — has specific rights and responsibilities. The system maintains a complete audit trail of all actions, making it easy to monitor and verify financial activities. Additionally, it minimizes human intervention and paperwork, which helps in reducing operational costs and promoting **eco-friendly digital** practices.

Another significant purpose of this project is to enable effective decision-making through real-time data insights and automated reporting. The system provides analytical reports and dashboards that help higher authorities track fund allocation efficiency, pending approvals, and overall financial performance. This not only assists in financial planning but also ensures that funds are utilized in the most optimal way to achieve organizational goals.

In essence, the Fund Management System serves the purpose of bringing transparency, accountability, and efficiency into the financial management process. It modernizes the way funds are handled, making the system more responsive, reliable, and aligned with digital governance standards. Through this solution, organizations can ensure that every fund movement is well-documented, properly approved, and effectively utilized, thereby fulfilling both administrative and regulatory objectives.

OBJECTIVES

The main objective of the Fund Management System is to develop an integrated and automated platform that efficiently manages the entire lifecycle of fund allocation — from proposal creation to utilization and reporting. The system is designed to minimize manual intervention and streamline the communication between different stakeholders such as the **Department, Head of Department (HOD),** Agency, and Vendors. It ensures that every financial transaction is properly recorded, approved, and monitored to maintain transparency and accountability in the management of public or organizational funds.

One of the key objectives is to digitize and simplify the fund approval workflow, enabling departments to create grant proposals, forward them to the HOD for review, and ensure smooth allocation to the implementing agency. This structured approach helps in reducing delays, eliminating redundant

paperwork, and promoting faster decision-making. Another objective is to ensure that fund utilization at the agency level is well-documented, with clear visibility into expenditure categories such as payroll, vendor payments, and project-related expenses.

The system also aims to provide role-based access control, ensuring that only authorized users can perform specific operations such as creating, verifying, or approving transactions. This enhances the security of financial data and prevents unauthorized access or manipulation. Through this objective, the project reinforces data integrity and trust across all operational levels.

Another important objective of the Fund Management System is to enable real-time monitoring and reporting of fund status and financial activities. The inclusion of automated dashboards and **Management Information System (MIS)** reports allows higher authorities to track fund allocation progress, monitor pending approvals, and generate performance insights. These analytical features assist in effective decision-making and long-term financial planning.

Furthermore, the system is intended to improve efficiency, transparency, and compliance with organizational and government financial policies. By maintaining a complete audit trail of transactions, it ensures accountability and supports future audits or evaluations. The system also contributes to reducing administrative costs and manual workload, promoting a sustainable, **paperless digital workflow**.

In conclusion, the objectives of the Fund Management System focus on achieving automation, transparency, security, and accuracy in financial operations. By aligning technology with organizational goals, the system ensures that funds are allocated, utilized, and tracked efficiently — paving the way for effective financial governance and improved service delivery.

SCOPE

The Fund Management System is designed to provide a comprehensive digital solution for managing financial activities within an organization, covering all stages from fund creation to final reporting. The scope of this project extends across multiple operational levels, including the Department, Head of Department (HOD), Agencies, Vendors, and Employees. Each stakeholder has defined roles and access privileges to ensure transparency, accountability, and efficiency throughout **the financial workflow**.

The system covers the entire fund lifecycle process — beginning with the creation of a grant proposal by the department, its review and approval by the HOD, allocation to relevant agencies, and subsequent utilization for various expenditures such as payroll and vendor payments. The process culminates in generating detailed reports that reflect the distribution, utilization, and remaining balance of funds. This comprehensive coverage ensures that the flow of funds is systematically monitored and fully traceable at every level.

Another major part of the project's scope involves automation of manual and repetitive financial processes. Tasks such as grant approvals, fund allocations, expenditure tracking, and report generation are digitized to minimize human error and enhance the speed of operations. By introducing automated approval workflows and OTP-based authentication for final transactions, the system ensures secure and verified financial operations.

The Fund Management System also provides robust data management and analytical capabilities. It integrates with a central database to store, retrieve, and update fund-related data in real time. The system's reporting module enables the generation of detailed Management Information System (MIS)

reports that help decision-makers evaluate fund utilization and optimize future allocations. This enhances the ability to plan budgets effectively and ensure that resources are used efficiently.

In terms of technological scope, the system can be implemented using modern platforms such as **.NET Core and SQL Server**, **ensuring scalability, security, and performance.** The web-based architecture allows access through browsers and supports multi-user functionality, enabling different departments and agencies to work collaboratively within a unified system.

Overall, the scope of the Fund Management System extends beyond basic fund tracking. It encompasses the automation, security, accountability, and transparency of financial operations within an organization. The project lays a foundation for transforming traditional fund management into a modern, efficient, and reliable digital process that aligns with organizational and governmental standards.

TECHNOLOGY AND LITERATURE REVIEW

The Fund Management System is built using a modern software development stack that ensures scalability, security, and high performance. The chosen technologies and frameworks were selected based on their ability to handle multi-user operations, data integrity, and workflow automation effectively. This section outlines the technological components used in the system and reviews related literature that supports the development of fund management and **financial automation systems**.

Technology Overview

The system is developed using .NET Core as the backend framework due to its cross-platform capabilities, robust performance, and modular architecture. .NET Core provides the foundation for building secure and scalable web applications, offering features such as dependency injection, middleware configuration, and strong authentication mechanisms. **The frontend is designed using HTML, CSS, JavaScript, and Bootstrap**, which provide an interactive and user-friendly interface, ensuring accessibility across different devices.

The database layer uses Microsoft SQL Server, which provides a reliable and efficient platform for storing and managing fund-related data. SQL Server supports relational database management, transaction safety, and data consistency, making it ideal for financial applications. Stored procedures and triggers are implemented to ensure data validation and prevent redundancy. The system also incorporates Entity Framework Core for Object-Relational Mapping (ORM), simplifying database operations and enhancing maintainability.

For authentication and security, the system integrates **OTP (One-Time Password) verification and role-based access control** to ensure that only authorized users can perform specific operations such as grant approvals and fund releases. This enhances the system's reliability and prevents unauthorized access to sensitive financial data.

In addition, the system utilizes ASP.NET Identity for managing users, roles, and permissions, while LINQ (Language Integrated Query) is used for efficient data querying and manipulation. These technologies collectively contribute to the seamless interaction between the front-end interface and the backend services, resulting in a secure and responsive application.

Literature Review

Several research studies and existing financial management systems were analyzed to identify best practices and common challenges in fund management automation. Traditional systems relied heavily on manual data entry and paper-based fund tracking, which often led to inefficiencies, delays, and lack of transparency. Modern approaches, as discussed in various literature sources, emphasize automation, integration, and digital record-keeping to enhance financial accountability.

Studies on e-Governance and financial automation systems suggest that incorporating technology into fund management significantly improves decision-making, auditability, and efficiency. Similar systems implemented in government and corporate sectors demonstrated that automated workflows reduce approval time, minimize errors, and ensure accurate fund allocation and reporting. The literature also highlights the importance of secure authentication mechanisms, **real-time reporting**, and cloud integration for improving accessibility and scalability

In summary, the technology and literature review establishes that the Fund Management System aligns with modern software engineering principles and financial management standards. By adopting the latest .NET technologies and integrating secure data handling practices, the system aims to deliver an effective, transparent, and reliable solution for managing financial operations within an organization

PROJECT PLANNING

The Fund Management System project was planned and executed using a **systematic and structured approach** to ensure timely delivery, quality, and functionality. Project planning serves as a blueprint for defining the project scope, objectives, deliverables, timeline, and required resources. A well-defined plan minimizes risks, ensures coordination among team members, and enhances the overall efficiency of development activities.

The project planning phase began with a requirement analysis stage, where the system's needs were identified based on the existing manual fund management challenges. Discussions with stakeholders such as Department staff, HODs, and financial officers helped gather functional and non-functional requirements. These requirements were documented clearly to provide a foundation for the system design and architecture.

Once the requirements were finalized, the team proceeded with system design and module breakdown. The project was divided into logical **modules such as Provision Grant Setup, Beneficiary Master, Fund Allocation, Expenditure, Payroll, and Reporting.** Each module was assigned specific tasks, timelines, and responsibilities. This modular approach ensured that development could occur in parallel streams, reducing dependency delays and improving overall productivity.

The project followed the Software Development Life Cycle (SDLC) model, primarily adopting the Waterfall methodology, where each phase—requirement gathering, design, implementation, testing, and deployment—was executed sequentially. This approach ensured clarity at every step and provided control over project progress. Gantt charts and milestone tracking were used to monitor progress, ensuring that deadlines were met effectively.

In terms of resource allocation, the project utilized a combination of software tools and technologies, including .NET Core for backend development, SQL Server for database management, and Bootstrap

for frontend UI design. Version control was managed using Git to maintain collaboration and track code changes efficiently. Each phase of development was reviewed and validated to maintain quality standards and compliance with organizational requirements.

Risk management and quality assurance were integral components of the project planning. Potential risks such as data inconsistency, system errors, and delayed approvals were identified early, and mitigation strategies were prepared. Regular testing cycles and feedback sessions were conducted to ensure that the system met the intended performance and reliability benchmarks.

Finally, the project planning phase concluded with a deployment and maintenance strategy, outlining how the system would be implemented, tested in real-time, and maintained post-deployment. Training and documentation were provided to end users to facilitate smooth adoption and reduce operational resistance.

In summary, the project planning ensured that the Fund Management System was developed in a controlled, step-by-step manner with clear goals, defined timelines, and efficient resource utilization. This structured approach contributed to achieving a robust, reliable, and user-friendly financial management solution.

SYSTEM ANALYSIS

STUDY OF CURRENT SYSTEM

The existing fund management process in most organizations is predominantly manual and paper-based, which makes it inefficient, time-consuming, and prone to errors. Currently, departments create and manage fund proposals, approvals, and expenditures through physical documentation or basic spreadsheets, leading to several operational challenges. The lack of a centralized digital system results in difficulties in tracking the status of funds, ensuring timely approvals, and maintaining accurate financial records.

In the current system, when a grant or fund request is initiated by a department, it must go through multiple levels of verification and approval. Each stage—submission, review, and sanctioning—is handled separately, often through manual communication or email correspondence. This causes unnecessary delays and miscommunication between departments, **Heads of Departments (HODs)**, and agencies. The absence of automated notifications and tracking mechanisms means that stakeholders frequently have to follow up manually to know the progress of a request.

Another major issue in the existing system is the lack of transparency and accountability. Since most transactions and records are maintained in physical form, it becomes difficult to trace fund utilization accurately. Financial discrepancies, duplication of entries, and incomplete audit trails are common occurrences. There is no centralized database where information on grants, allocations, expenditures, and beneficiaries can be stored and accessed securely. This fragmented system not only slows down operations but also increases the risk of data loss or mismanagement.

Additionally, fund allocation and expenditure tracking are not standardized. Different departments or agencies may use separate formats for data entry and reporting, which complicates the process of generating consolidated financial reports. Manual calculation of available and utilized funds often leads to inaccuracies, making decision-making difficult for higher authorities. The reporting process is also

cumbersome, as data must be collected from multiple sources, verified manually, and then compiled into a summary format.

From a security perspective, the current system provides minimal protection against unauthorized access or data manipulation. **Sensitive financial information**, such as vendor details, payment histories, and fund approvals, is often stored in unsecured files or shared through non-encrypted channels. This raises serious concerns about data privacy and compliance with financial governance standards.

In conclusion, the study of the current system reveals significant limitations in efficiency, transparency, and security. The manual approach is insufficient for managing complex financial workflows involving multiple stakeholders and large-scale fund transactions. Therefore, there is an urgent need for a centralized, automated, and secure Fund Management System that can streamline operations, ensure accountability, and provide real-time insights into fund activities across departments and agencies.

PROBLEM AND WEAKNESSES OF CURRENT SYSTEM

The existing Fund Management System, which often relies on manual or semi-digital processes such as spreadsheets, paper records, or basic software tools, faces several significant challenges that hinder efficiency and accuracy. One of the main problems is the lack of automation in handling financial data and transactions. Manual data entry increases the likelihood of human errors, leading to inconsistencies in records, incorrect calculations, and difficulties in tracking funds accurately. This inefficiency often results in delayed decision-making and poor financial forecasting.

Another **major weakness of the current system is limited data security**. Traditional systems or decentralized record-keeping methods are prone to unauthorized access, data loss, or manipulation. Since financial data is highly sensitive, the absence of robust authentication and encryption mechanisms exposes organizations to risks such as data breaches and fraud. Furthermore, backup and recovery options in such systems are often inadequate, making it difficult to restore data in case of accidental deletion or technical failures.

The current system also lacks real-time monitoring and reporting capabilities. Financial transactions and reports are often updated periodically rather than instantly, which means users cannot access the latest data when needed. This delay can lead to poor financial insights and difficulty in maintaining transparency across departments or stakeholders. Additionally, the reporting process is often time-consuming, as generating and consolidating financial reports manually requires significant effort and coordination.

Another issue is poor scalability and integration. The existing systems are usually not flexible enough to adapt to the growing volume of data or integrate with other enterprise applications like ERP or CRM systems. As organizations expand, these limitations make it challenging to manage multiple accounts, departments, or branches efficiently.

Lastly, user accessibility and convenience are major drawbacks of the current system. Many systems are desktop-based and lack web or mobile support, restricting access to authorized users outside the office network. This limits collaboration and makes remote fund management nearly impossible.

In summary, the current system suffers from inefficiencies related to manual processes, data inaccuracy, lack of security, poor scalability, and limited accessibility. These weaknesses clearly highlight the need for a modernized, automated Fund Management System built on a reliable framework like .NET Core, which can enhance performance, security, and overall user experience.

REQUIREMENTS OF NEW SYSTEM

The development of a new Fund Management System is essential to overcome the drawbacks of the existing manual and outdated methods. The new system should be designed with clear, well-defined requirements that ensure reliability, efficiency, and scalability. The requirements of the new system can be broadly classified into Functional Requirements, Non-Functional Requirements, and Technical

Functional Requirements

- User Authentication and Authorization: The system should allow secure login and access control, ensuring only authorized users can perform specific tasks based on their roles.
- Fund Management: The system should enable tracking, allocation, and management of funds across multiple departments or projects with real-time updates.
- Transaction Management: All financial transactions such as deposits, withdrawals, and fund transfers should be accurately recorded and processed automatically.
- Report Generation: The system must generate detailed reports on income, expenses, balances, and fund utilization with options for daily, monthly, and yearly summaries.
- Notifications and Alerts: The system should send automatic notifications for approvals, low balances, or transaction updates to maintain transparency and timely actions.
- Audit Trail: A record of all user activities should be maintained for accountability and verificatio

Non-Functional Requirements

- Performance: The system should respond to user actions within seconds and be capable of handling multiple users simultaneously without delay.
- Security: Strong encryption and data protection methods should be implemented to safeguard sensitive financial information from unauthorized access.
- Usability: The system should have an intuitive and easy-to-navigate interface for all users, minimizing the need for extensive training.
- Scalability: The system should support an increasing number of transactions and users as the organization grows.
- Reliability: The system should ensure continuous availability with minimal downtime and automatic data backup for recovery in case of failure.
- Maintainability: The system should be designed in a modular way to allow easy updates, debugging, and maintenance.

Technical Requirements

- Platform: Microsoft .NET Core Framework for cross-platform compatibility and efficient backend processing.
- Programming Language: C# for developing business logic and API functionality.
- Database: Microsoft SQL Server for secure and structured data storage and retrieval.
- Frontend: ASP.NET Core MVC or Blazor for a responsive and interactive web interface.
- Operating System: Compatible with Windows, Linux, or macOS servers.
- Development Tools: Visual Studio or Visual Studio Code for coding and debugging.
- Hosting Environment: Cloud-based deployment (e.g., Azure) for scalability, security, and remote accessibility

SYSTEM FEASIBILITY

System feasibility refers to the process of evaluating whether the proposed Fund Management System is practical, beneficial, and achievable within the available resources and constraints. It helps determine whether the system can be successfully developed, implemented, and maintained to meet organizational goals. The feasibility study ensures that the project is viable in terms of technology, economy, and operations. The main types of feasibility considered are Technical Feasibility, Economic Feasibility, and Operational Feasibility.

Technical Feasibility

- Technical feasibility assesses whether the existing technology, software, and hardware resources are sufficient to support the development and implementation of the proposed system.
- The Fund Management System is technically feasible because it will be developed using the latest .NET Core Framework, which is powerful, secure, and scalable for enterprise applications. The system uses Microsoft SQL Server for reliable database management and ASP.NET Core MVC for responsive and user-friendly interfaces. The required hardware and network infrastructure are minimal and can be easily provided by the organization. Additionally, since the development tools like Visual Studio are widely available, the project team will face no technical limitations. Thus, from a technology perspective, the system can be efficiently developed and deployed.

Economic Feasibility

- Economic feasibility evaluates the cost-effectiveness of the proposed system by comparing the expected benefits with the overall costs involved in development, implementation, and maintenance.
- The proposed Fund Management System is economically feasible because the initial investment in software, hardware, and manpower will result in significant long-term benefits. The automation of fund tracking, reporting, and approval processes will reduce manual effort, minimize human errors, and save valuable time. Operational costs such as paperwork, storage, and manpower will also decrease considerably. Since .NET Core and SQL Server can be used under existing organizational licenses, the cost of implementation will remain within a reasonable budget. Overall, the financial advantages and efficiency improvements far outweigh the initial development expenses.

Operational Feasibility

- Operational feasibility focuses on how well the proposed system fits within the existing environment and whether it will be accepted by users and administrators.
- The new system is operationally feasible as it simplifies fund-related processes, improves transparency, and enhances user satisfaction. The interface is user-friendly and designed to require minimal training for users at different levels (Department, HOD, and Corporation users). The automation of approvals, fund allocations, and expenditure tracking ensures smoother operations and faster decision-making. Moreover, the system's ability to generate **real-time reports** and maintain an audit trail will improve overall accountability and compliance. The involvement of users during system design and testing will ensure that the final product meets operational needs effectively.

FEATURES OF NEW SYSTEM

The proposed Fund Management System is designed to overcome the drawbacks of the existing manual process by introducing automation, accuracy, and transparency. This new system integrates all fund-related operations—from provision setup to expenditure tracking—within a unified digital platform. The main features of the new system are described below:

1. User Authentication and Role Management

The system provides a secure **mobile-based OTP login** to ensure that only authorized users can access the application. Different user roles such as Department User, HOD, and Corporation User are defined with specific access rights **(Maker, Verifier, Checker).** This ensures that each user can perform only the tasks relevant to their role, maintaining data confidentiality and accountability.

2. Provision Grant Setup

The system allows authorized users to create, modify, and approve grant provisions efficiently. Each provision grant is tracked with a status flow **(Created → Verified → Approved),** ensuring that funds are allocated based on verified and approved requests only.

3. Beneficiary Management

The new system includes a Beneficiary Master Module to manage and store all details of the agencies or individuals receiving funds. It allows easy creation, updating, and validation of beneficiary data to ensure accuracy in fund distribution.

4. Fund Allocation and Limit Setup

Funds can be allocated to various agencies or departments with predefined limits. The system ensures that fund allocation does not exceed the sanctioned budget and maintains a proper record of all allocations. This improves financial control and helps in effective budget utilization.

5. Expenditure and Payment Tracking

This feature allows users to record, monitor, and approve expenditures related to each fund allocation. The Expenditure Module helps track payments and maintain a transparent audit trail. It ensures that every transaction is verified before approval, minimizing the risk of errors or fraud.

6. Payroll Management

The Payroll Module automates salary processing for staff or project-based employees involved in fund management activities. It calculates and records all salary details, deductions, and approvals efficiently within the same system.

7. Dynamic Reporting and Data Analysis

The system provides real-time dashboards and generates various reports such as fund utilization, pending approvals, expenditure summaries, and allocation history. These reports assist higher authorities in decision-making and performance monitoring.

8. Secure Database Management

All information related to users, beneficiaries, allocations, and transactions is stored in a centralized SQL Server Database. Data integrity is maintained through proper indexing, constraints, and relationships between tables. Regular backups and data encryption further enhance system security.

9. User-Friendly Interface

Developed using .NET Core MVC, the system provides an intuitive, clean, and responsive user interface that simplifies navigation and data entry. Minimal training is required for end users to operate the application effectively.

10. Multi-Level Approval Workflow

Each fund-related activity follows a well-defined approval workflow involving Maker, Verifier, and Checker roles. This ensures that all transactions undergo a three-level verification process before being finalized.

11. Notification and Alerts System

The system automatically sends notifications and alerts to users for pending approvals, upcoming deadlines, or rejected requests. This helps maintain operational efficiency and ensures that no critical activity is missed.

LIST MAIN MODULES

The Fund Management System is divided into several core modules, each responsible for a specific functional area. These modules work together to ensure smooth fund operations, from grant creation to expenditure tracking. Below is the detailed description of the main modules included in the system:

1. User Master Module

This module manages all user information and access control. It includes user registration, login through mobile OTP, and role-based access management. It ensures that only authorized users (Department, HOD, Agency, etc.) can perform their assigned functions.

2. Role Master Module

The Role Master defines the different levels of user roles such as Maker, Verifier, and Checker. These roles determine who can create, verify, or approve fund-related entries within the system.

3. Menu Master Module

This module maintains the navigation structure of the application. It dynamically controls which menus or options are visible to each user based on their assigned role and permissions.

4. Provision Grant Setup

Module The Provision Grant Setup module allows the department to create and manage grants. It handles the complete lifecycle of grant creation — from initial entry to approval — and maintains all relevant financial data for each grant.

5. Beneficiary Master Module

This module stores and manages details of beneficiaries or agencies receiving the funds. It ensures accurate information is maintained for all beneficiaries to enable correct allocation and payment processing.

6. Fund Allocation Module

The Fund Allocation module helps the Head of Department (HOD) or Finance Officer allocate approved funds to different agencies. It ensures that allocations do not exceed the approved budget and keeps track of all financial distributions.

7. Expenditure Module

This module records all fund utilization activities. It allows users to log expenses, verify expenditure details, and approve payments. It also supports generating expenditure summaries for analysis and auditing.

8. Payroll Module

The Payroll module automates salary and wage-related transactions. It supports creating, verifying, and approving payroll entries for project staff, vendors, or employees linked with fund activities.

9. Reports Module

This module generates various analytical and operational reports, including fund utilization, pending approvals, expenditure details, and allocation status. Reports can be exported in multiple formats (PDF, Excel) for official use.

10. Approval Tracker Module

The Approval Tracker monitors the workflow progress of all fund-related operations. It records actions performed by Makers, Verifiers, and Checkers, ensuring complete transparency and traceability in approvals

SELECTION OF SOFTWARE, TECHNIQUES, APPROACHES & JUSTIFICATION

The development of the Fund Management System (FMS) requires the careful selection of suitable software tools, programming techniques, and development approaches to ensure efficiency, scalability, and maintainability. The choices made in this project are based on the nature of operations, security needs, and integration capabilities of the system.

Selection of Software

- Frontend:

Developed using HTML5, CSS3, Bootstrap, and JavaScript, ensuring a responsive and user-friendly interface. These technologies allow cross-browser compatibility and provide flexibility in designing a clean, intuitive dashboard for users.

- Backend:

The backend is implemented using .NET Core, a modern, open-source, and cross-platform framework developed by Microsoft. It offers powerful performance, scalability, and easy integration with databases and APIs, making it ideal for enterprise-level applications.

- Database:

Microsoft SQL Server is chosen as the relational database for storing all structured data. It ensures robust data integrity, strong security, and high-speed query performance, which are crucial for handling fund transactions.

- Development Environment:

Visual Studio / Visual Studio Code serves as the IDE (Integrated Development Environment) for development. It provides advanced debugging, version control integration, and seamless .NET Core support.

- Version Control:

Git and GitHub are used for maintaining code versions and collaboration among developers.

- Deployment:

The application can be deployed on IIS (Internet Information Services) or Azure Cloud, ensuring easy scalability and high availability.

Selection of Techniques

- Object-OrientedProgramming(OOP):

The project follows OOP principles such as abstraction, inheritance, and encapsulation, improving modularity and reducing redundancy.

- MVCArchitecture(Model-View-Controller):

The system uses the MVC pattern, where:

- Model handles data and business logic,

- View manages user interface and presentation,

- Data Validation & Security:

Data validation is performed both at the frontend and backend.

Security measures like JWT-based authentication, role-based access control (RBAC), and encrypted storage are implemented.

Selection of Approach

The Software Development Life Cycle (SDLC) approach adopted is the Agile Methodology, which allows:

- Iterative development and continuous feedback,
- Early identification of issues,
- Flexibility to incorporate user requirements during each sprint,
- Better collaboration among the project team.

Justification for the Selection

- .NET Core offers high performance, security, and is ideal for large-scale, modular systems like fund management.
- SQL Server ensures reliable data storage, fast transactions, and supports complex queries and stored procedures.
- Agile methodology enhances adaptability, ensuring the system meets stakeholder expectations.
- MVC structure ensures maintainability and makes future enhancements easier.
- Object-oriented design simplifies debugging, testing, and scaling the system.
- Cloud-ready deployment ensures the application can handle large user bases and high data loads efficiently.

SYSTEM DESIGN

SYSTEM DESIGN AND METHODOLOGY

The System Design and Methodology phase represents one of the most critical stages in the development of the Fund Management System (FMS). It serves as the blueprint for constructing the actual system and ensures that all technical and functional aspects are properly aligned with the project's objectives. The design phase transforms user requirements into a clear system structure that can be implemented efficiently and maintained easily. The methodology, on the other hand, defines the systematic approach followed throughout the development process to achieve accuracy, flexibility, and scalability.

The Fund Management System is designed to manage the complete workflow of fund handling — beginning from the creation of grants by the Department, followed by review and allocation by the Head of Department (HOD), to the utilization of funds by agencies, and finally the generation of reports showing detailed expenditure and payment transactions. To ensure modularity and clarity, the design process is divided into two levels: the High-Level Design (HLD) and the Low-Level Design (LLD). The HLD focuses on the overall structure and interaction between major modules such as Department, HOD, Agency, and Vendor. The LLD goes deeper into internal logic, database relationships, and process-level details within each module.

For this project, the Agile Software Development Methodology is adopted. Agile is best suited for projects that require continuous feedback and improvement throughout the development cycle. It enables the system to be developed in smaller, manageable iterations called sprints. Each sprint delivers a working module—such as Grant Creation, Fund Allocation, or Expenditure Tracking—that is tested and refined before moving to the next phase. This iterative approach enhances collaboration, reduces risk, and ensures that user requirements are met effectively.

The system architecture of the Fund Management System is based on a three-tier architecture, which divides the application into three layers: the presentation layer, the business logic layer, and the data layer. The presentation layer (front-end) is developed using HTML, CSS, JavaScript, and Bootstrap, providing a responsive and user-friendly interface for different types of users. The business logic layer, built using .NET Core MVC Framework, handles all operations related to fund approval, allocation, validation, and processing. The data layer, managed by Microsoft SQL Server, ensures secure storage, quick access, and consistency of information across modules. This layered design makes the system more scalable, maintainable, and easier to update in the future.

The design phase also includes key components such as user interface design, database design, and data flow diagrams (DFDs). The user interface is structured to be clean, simple, and easy to navigate, allowing users like Department staff, HODs, and Agency users to perform their tasks efficiently. The database design consists of well-defined tables like ProvisionGrant, FundAllocation, Expenditure, Payroll, and ApprovalTracker, which are interconnected using relational keys. The data flow diagrams represent how information moves between users and modules — for instance, when a grant is created by the Department, approved by the HOD, allocated to the Agency, and ultimately disbursed to vendors or employees. These visual designs ensure logical clarity and eliminate redundancy in the workflow.

Security is an integral part of the system design. The FMS includes OTP-based login authentication, ensuring that only authorized users can access the system. Role-based access control (RBAC) is implemented to distinguish between Department users, HODs, and Agencies. Additional measures such as input validation, encrypted data handling, and audit trails are included to protect sensitive information and prevent misuse. Every transaction, approval, and payment is logged, maintaining transparency and accountability throughout the system.

The workflow of the Fund Management System is structured in a linear yet flexible manner. The process starts when the Department creates a provision grant entry, which is then reviewed and approved by the HOD. Once approved, the HOD allocates the funds to the respective agencies, who then utilize these funds for various expenditures or payroll disbursements. Vendors or employees receive payments through the system, and comprehensive MIS reports are generated to summarize fund usage and transaction history. Each stage is designed with checks and validations to ensure smooth processing and compliance with organizational policies.

The choice of Agile methodology combined with a three-tier architecture is justified due to its efficiency, adaptability, and ease of maintenance. Agile allows for faster delivery and accommodates changes even during development. The three-tier model promotes separation of concerns, where each layer handles its own responsibility, ensuring flexibility in debugging, testing, and upgrading the application. Moreover, the use of .NET Core and SQL Server provides a stable and secure platform suitable for enterprise-level systems that manage financial data.

In conclusion, the System Design and Methodology phase lays a strong foundation for developing a reliable, secure, and user-friendly Fund Management System. By combining modular design, Agile methodology, and modern architectural principles, the system ensures efficient fund tracking, transparent reporting, and scalable performance. This structured design approach enables the project to

meet its objective of providing a digital, streamlined, and accountable fund management process across all administrative levels.

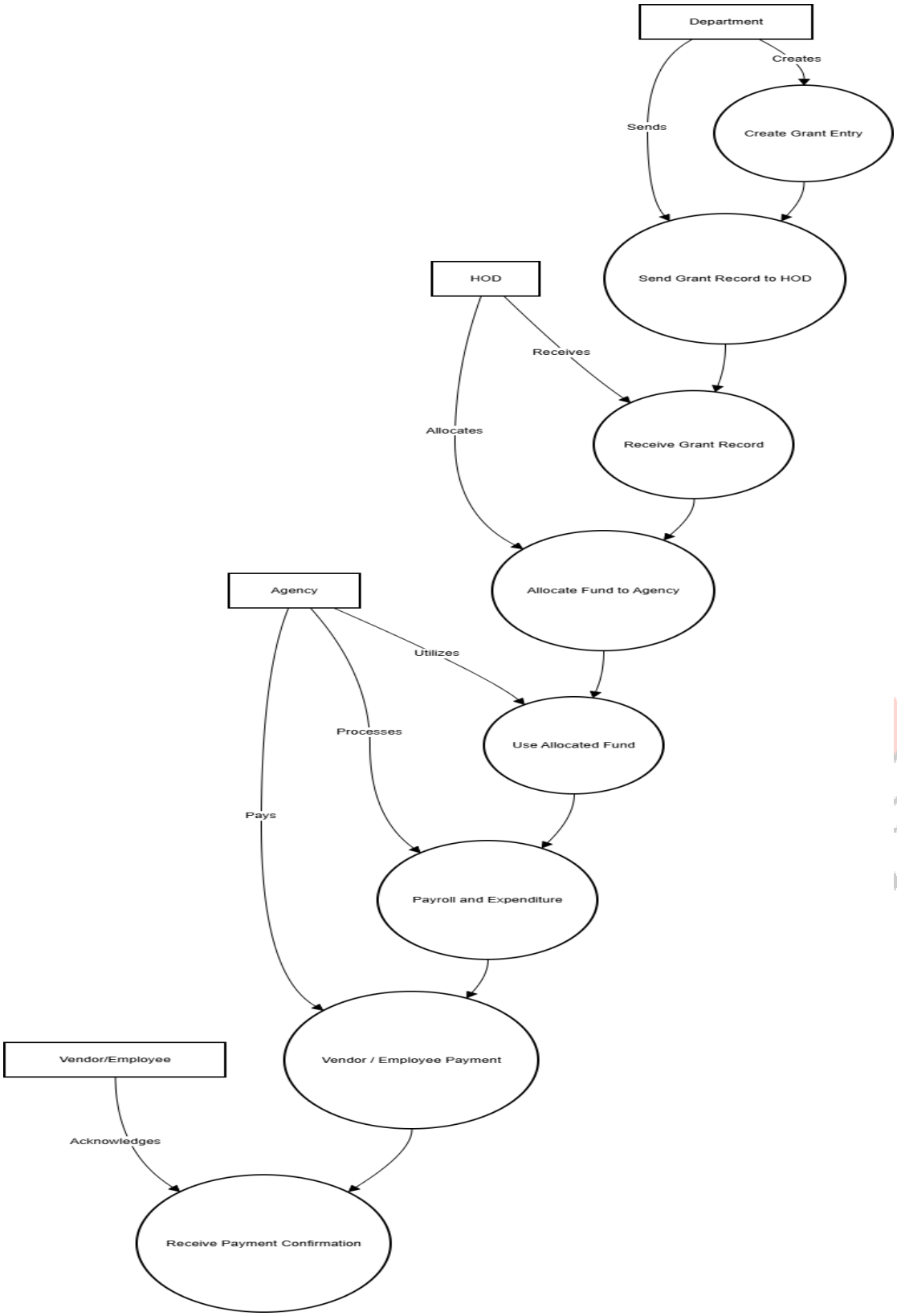

Fig 1 Fund Management System UseCase Diagram

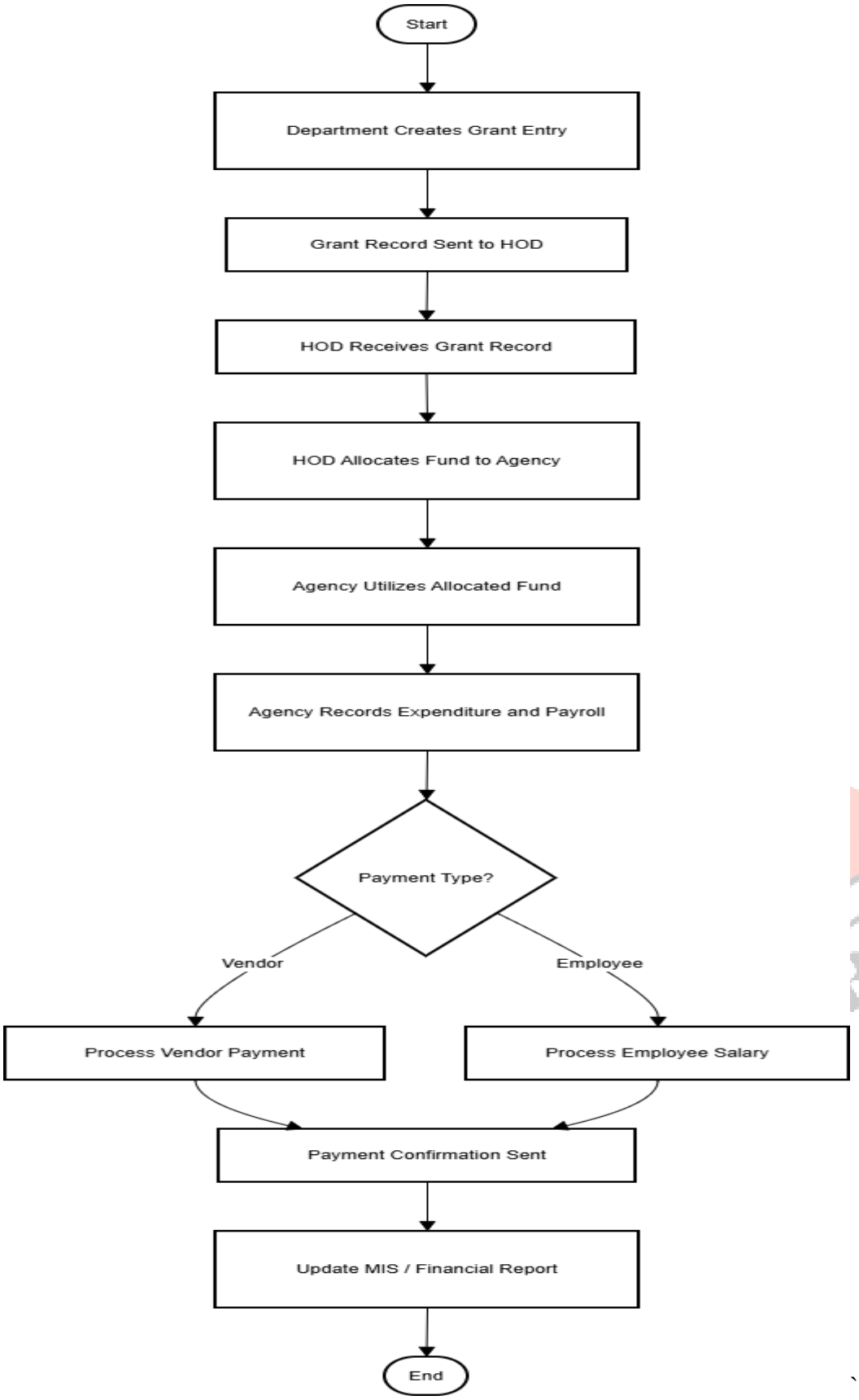

Fig 2 Activity Diagram

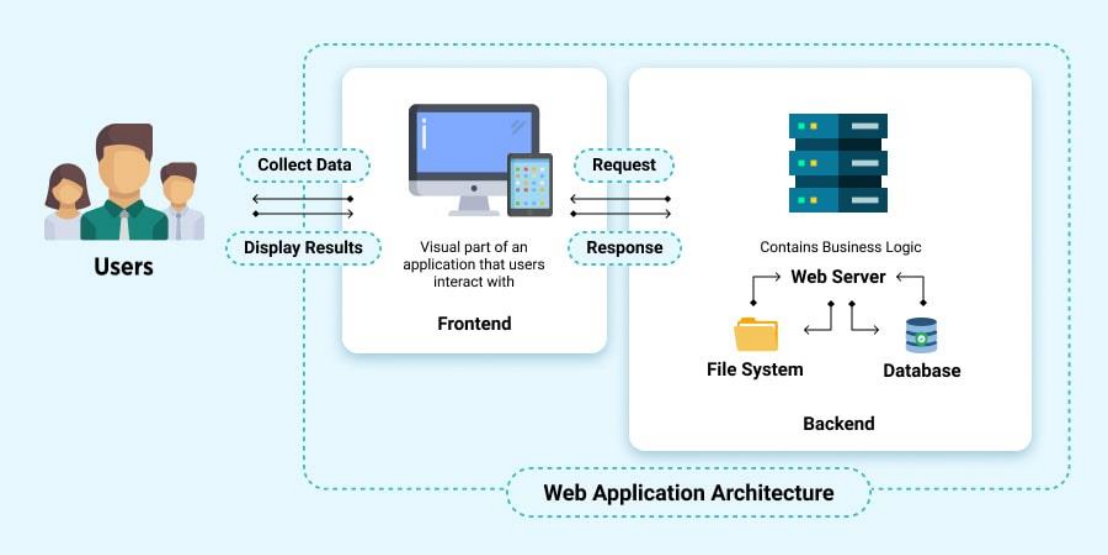

Fig 3 System Architecture Diagram

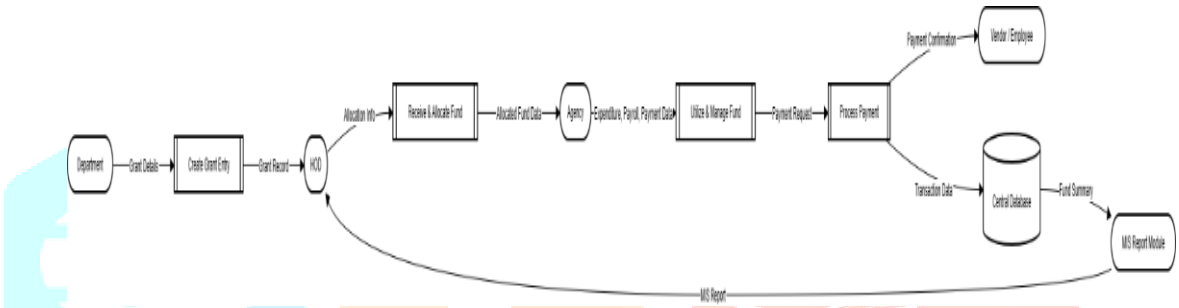

Fig 4 Data Flow Diagram

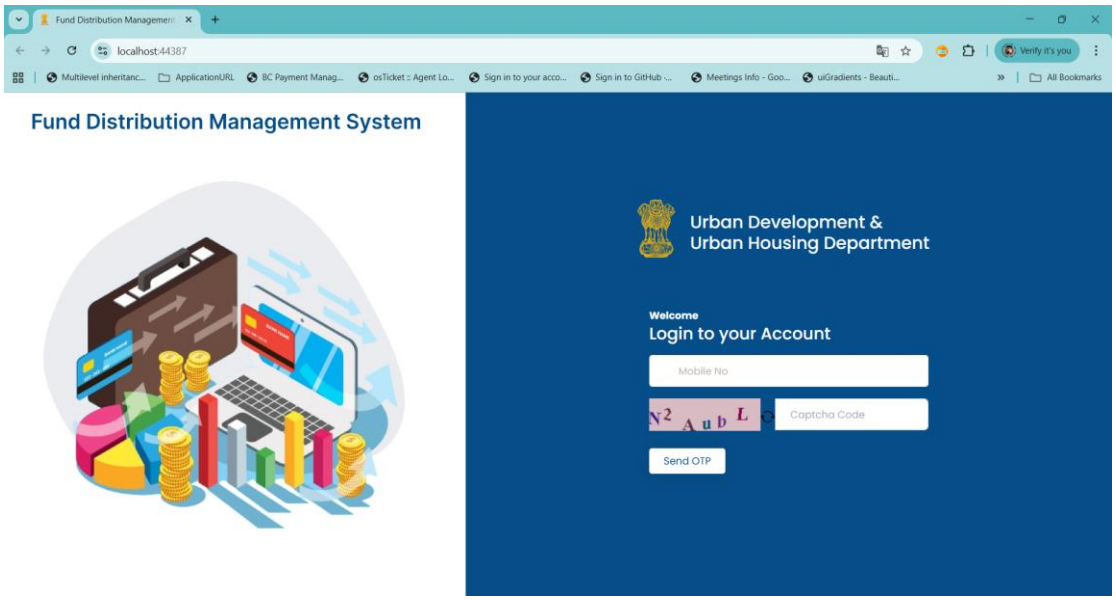

Fig 5 Login Page

- User signs in with mobile number; an OTP is sent and validated to authenticate and grant access.

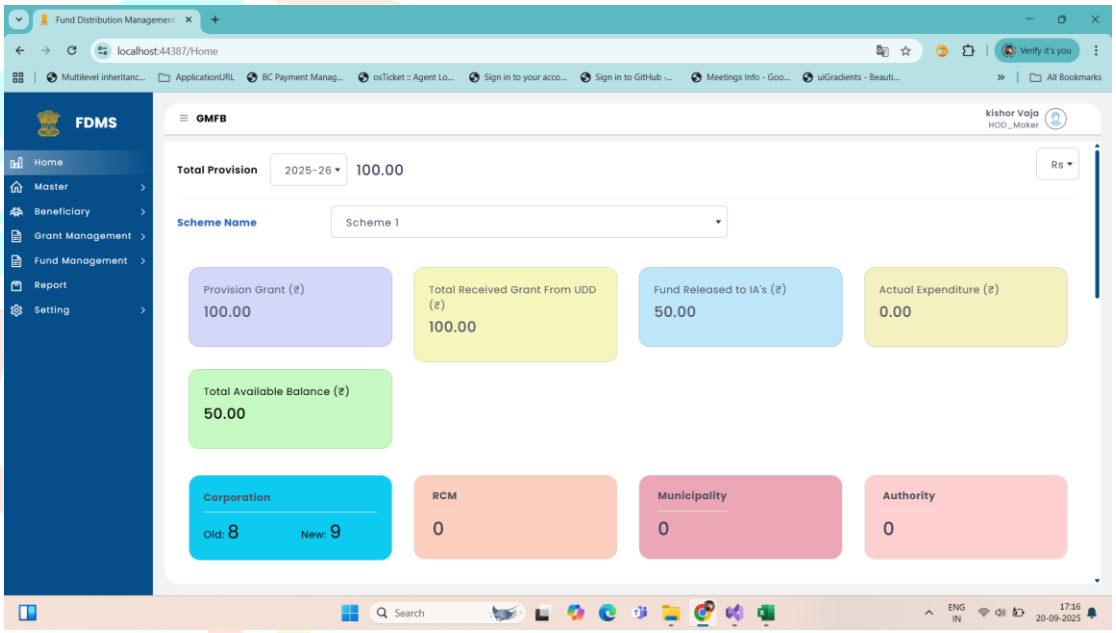

Fig 6 Dashboard Page

- Dashboard page displays an overview of fund transactions, allocations, approvals, and expenditure summaries in a single interface.

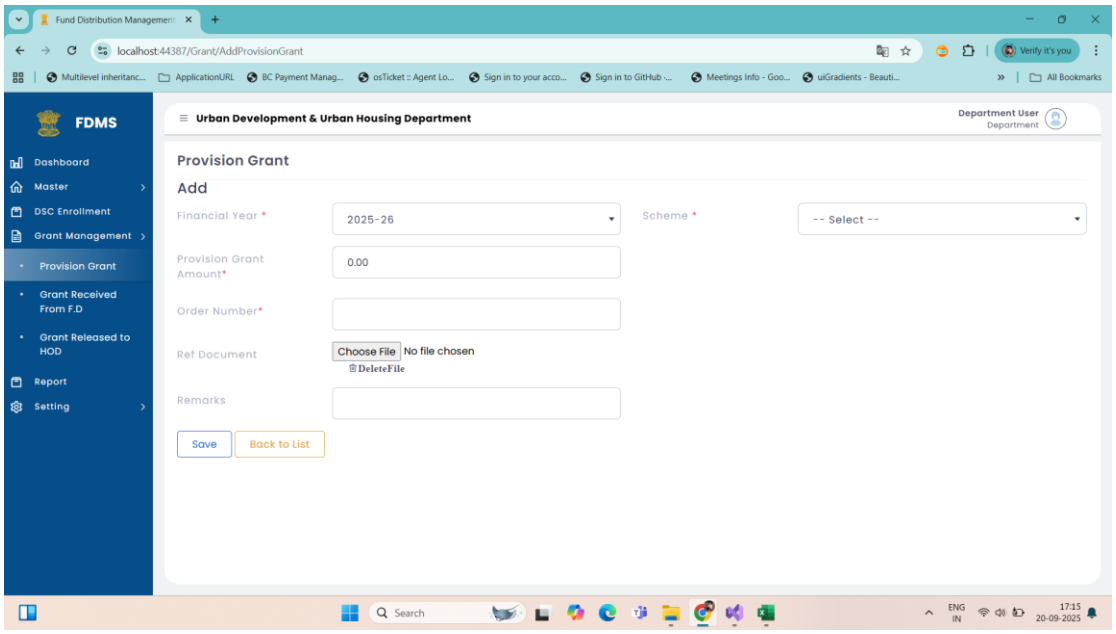

Fig 7 Provision Grant Create

- Interface for creating and managing initial grant provisions before allocation to HOD or agencies.

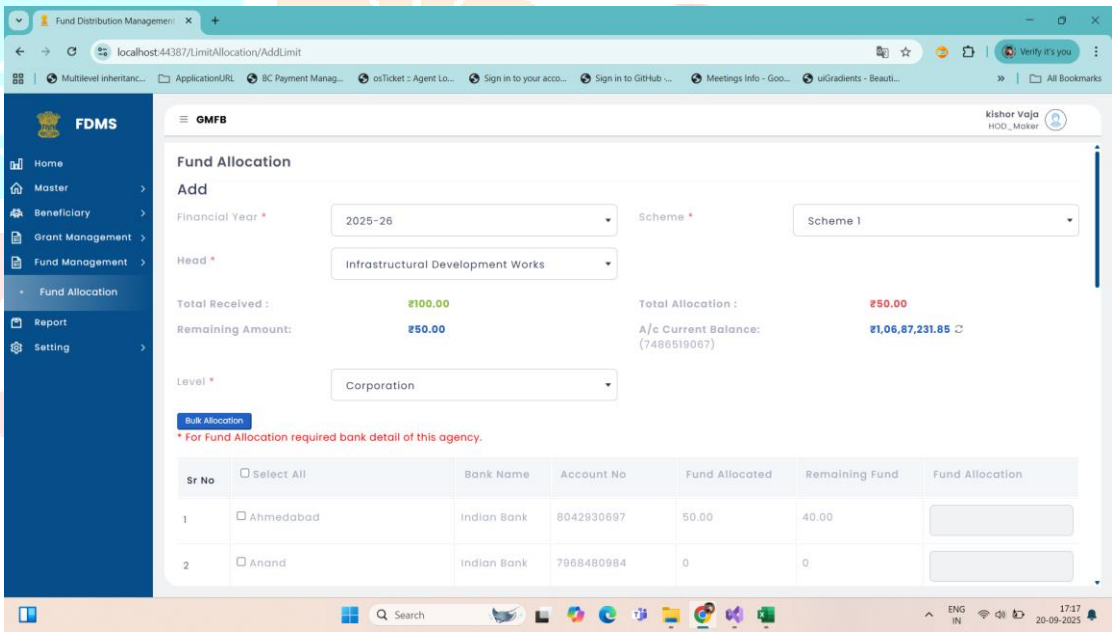

Fig 8 Fund Allocation to Agency

- Fund Allocation to Agency page enables the HOD to distribute approved grant amounts to respective agencies based on project requirements.

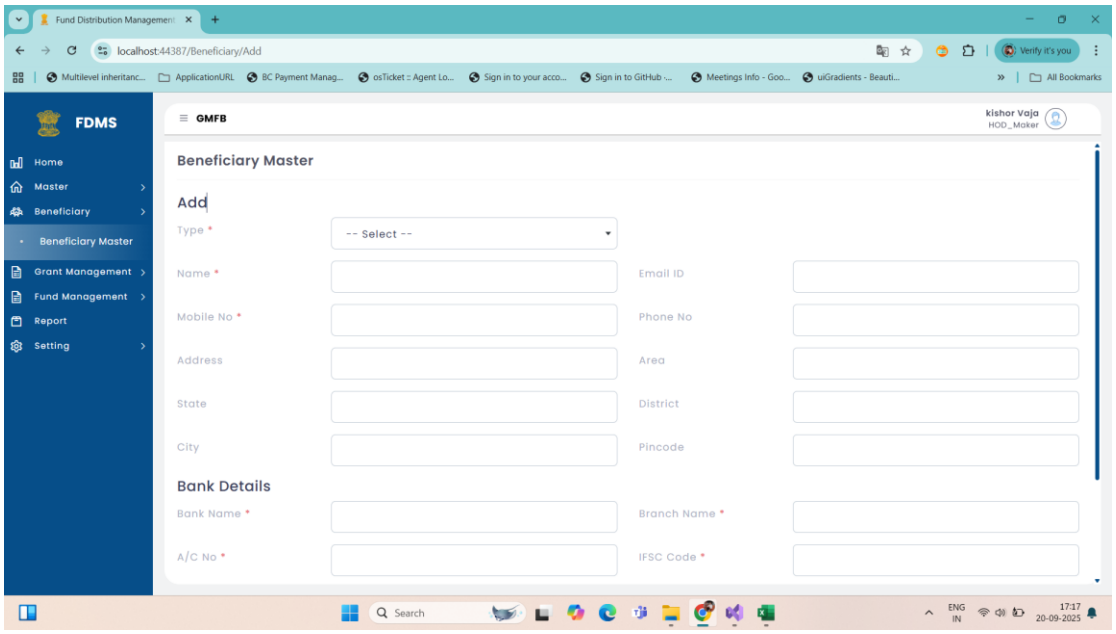

Fig 9 Beneficiary Creation Form

- Used by agencies or administrators to create and manage beneficiary records for payroll or vendor payments.

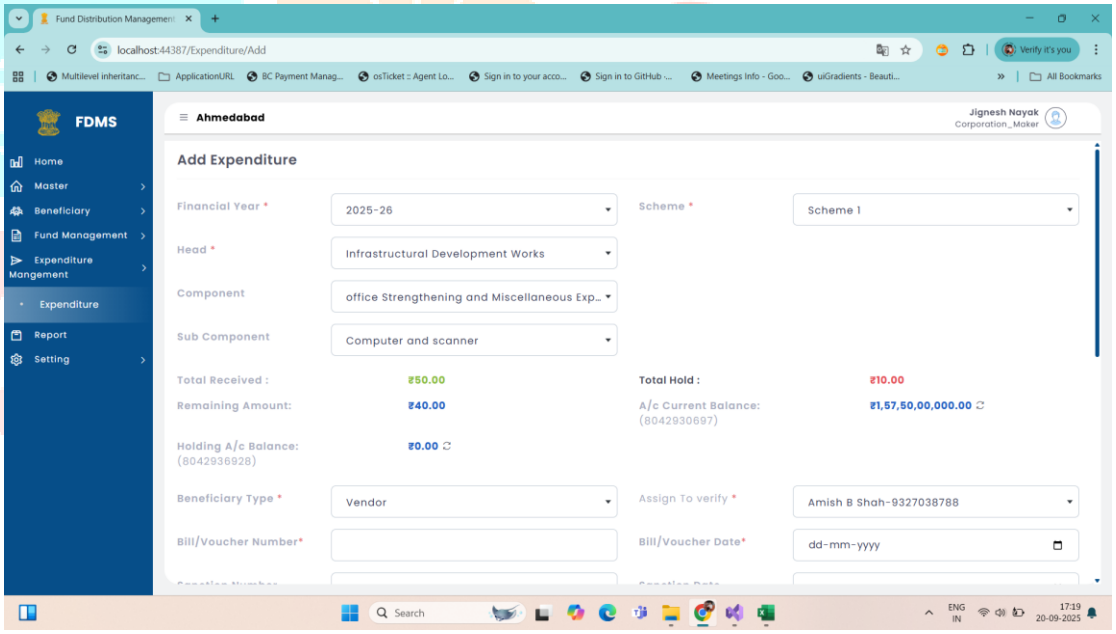

Fig 10 Expenditure Payment

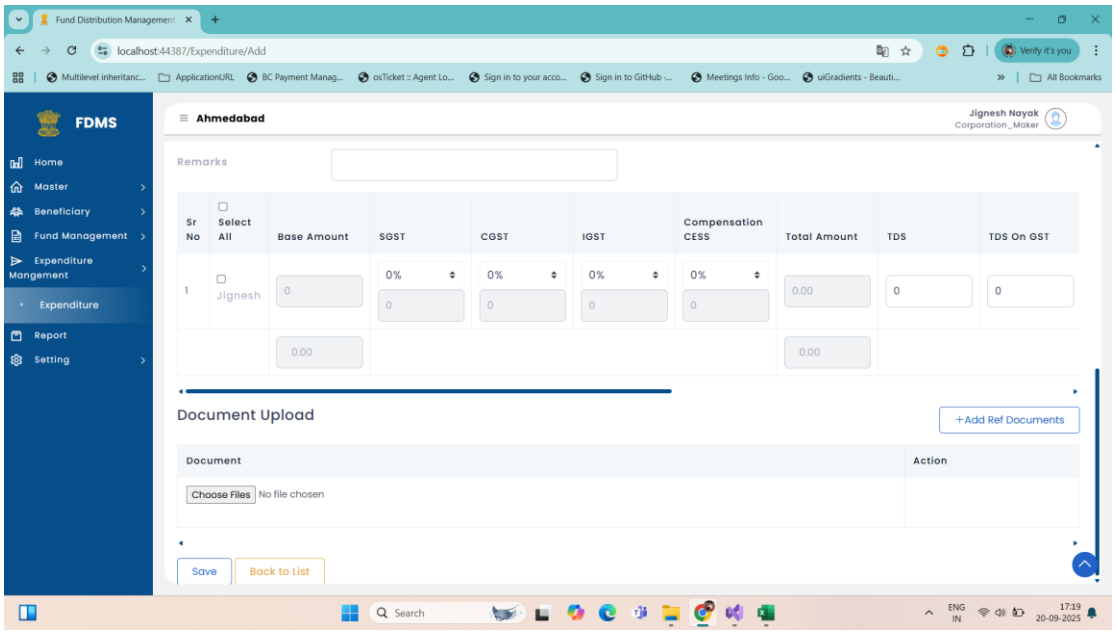

Fig 11 Expenditure Payment

- Used by agencies to execute fund payments and maintain a record of all financial transactions under each allocation.

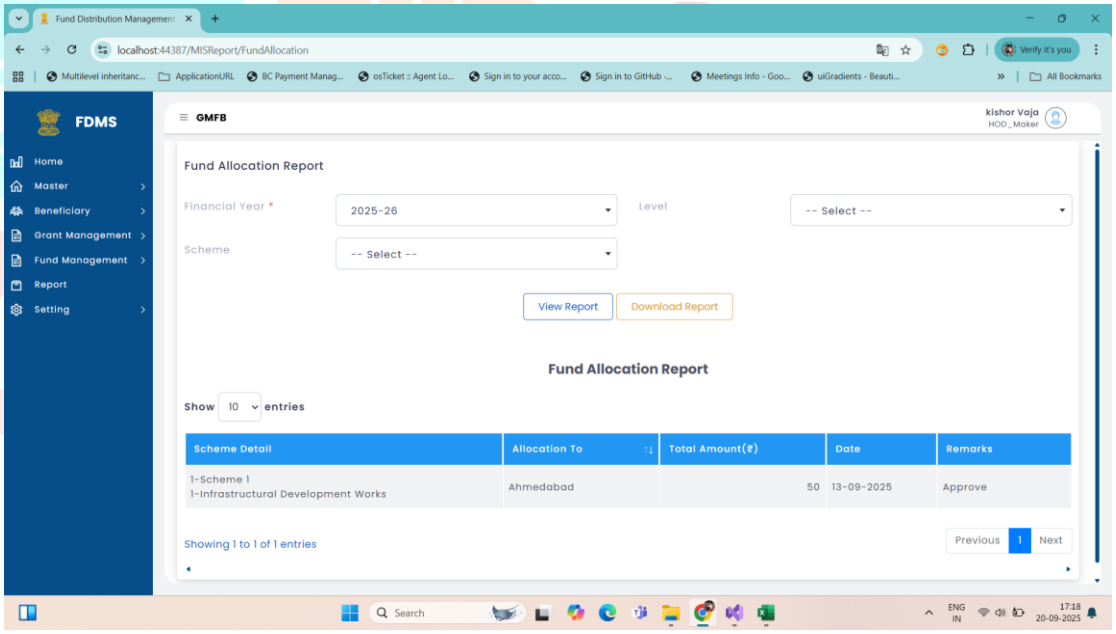

Fig 12 Report Page

- Report Page displays detailed summaries of fund allocations, expenditures, and payments across all departments and agencies.

IMPLEMENTATION

IMPLEMENTATION PLATFORM/ENVIRONMENT

The implementation platform for the Fund Management System is based on a modern, scalable, and secure technology stack designed to ensure high performance, maintainability, and ease of deployment. The project is implemented using .NET Core as the primary backend framework due to its cross-platform capabilities, robustness, and strong support for RESTful API development. It allows the system to efficiently handle business logic, database interactions, and user authentication. The C# programming language is used for server-side coding because of its rich libraries, strong typing, and integration with .NET Core, which enhances development speed and reduces runtime errors.

For the frontend, technologies like HTML5, CSS3, JavaScript, and Bootstrap are utilized to create an intuitive and responsive user interface. These technologies ensure that the application provides a seamless experience across different devices and browsers. The combination of Bootstrap and JavaScript enhances the design consistency and interactivity of the system, allowing users to easily navigate through various modules such as Fund Allocation, Expense Tracking, and Report Generation.

The database used for the project is Microsoft SQL Server, chosen for its reliability, security features, and compatibility with .NET Core. It stores all financial data, user details, and transaction records in a structured format that supports fast query execution and easy data retrieval. The Entity Framework Core (EF Core) is used as an Object-Relational Mapping (ORM) tool to simplify database operations, enabling developers to interact with the database using high-level C# objects instead of raw SQL queries.

The system is deployed on a Windows-based environment and can also run on Linux servers due to the cross-platform nature of .NET Core. The Visual Studio IDE serves as the primary development environment, providing integrated tools for coding, debugging, and version control. Version management is handled using GitHub, ensuring collaborative development and efficient tracking of changes throughout the software lifecycle.

Table 1 Project Module Overview

Sr No	Module Name	Table Name
1	Provision Grant Setup	ProvisionGrant,ApprovalTracker
2	Beneficiary Master	BeneficiaryMaster,ApprovalTracker
3	Fund Allocation	LimitAllocation,ApprovalTracker
4	Expenditure	Expenditure,ApprovalTracker
5	PayRoll	PayRoll,ApprovalTracker
6	User Master	UserMaster
7	Role Master	RoleMaster
8	Menu Master	MenuMaster

Table 2 User Types and Roles

Sr No	User Type	User Role
1	Department User	Maker (Entry Creation)
2	HOD User	Maker (Entry Creation) Verifier (Entry Verify) Checker (Entry Final Approval)
3	Corporation User (Agency User)	Maker (Entry Creation) Verifier (Entry Verify) Checker (Entry Final Approval)

Table 3 Database Table Structure

Sr No	Table Name	Column Name	Datatype
1	UserMaster	UserId	Uniqueidentifier
2	UserMaster	Name	varchar(max)
3	UserMaster	LasterName	varchar(max)
4	UserMaster	MobileNumber	varchar(10)
5	UserMaster	EmailId	varchar(Max)
6	UserMaster	RoleId	Uniqueidentifier
7	UserMaster	Password	varchar(max)
8	UserMaster	Remarks	varchar(max)
9	UserMaster	UserName	varchar(max)
10	UserMaster	DepartmentId	int
11	UserMaster	LevelId	int
12	UserMaster	CreateDate	datetime
13	UserMaster	CreateBy	Uniqueidentifier
14	UserMaster	UpdateDate	datetime
15	UserMaster	UpdateBy	Uniqueidentifier
16	UserMaster	AuthenticationType	int

Abbreviations

- **LIST OF SYMBOLS, ABBREVIATIONS AND NOMENCLATURE**

Symbol	Description
₹	Indian Rupee
%	Percentage
→	Flow Direction

Abbreviation	Full Form
FMS	Fund Management System
HOD	Head of Department
OTP	One-Time Password
MIS	Management Information System
SQL	Structured Query Language
UI	User Interface

Nomenclature	Description
Grant	Financial fund entry created by Department
Allocation	Distribution of approved funds
Agency	Implementing organization managing fund usage
Vendor/Employee	External party receiving payments
Beneficiary	Employee or Vendor receiving financial support

- **Software Development Life Cycle (SDLC)**

The Software Development Life Cycle (SDLC) is a structured process that defines the stages involved in developing high-quality software applications. In the context of the *Fund Management System* developed using .NET Core, SDLC ensures that the project is executed systematically, efficiently, and in alignment with the functional requirements. Each phase plays a critical role in transforming user needs into a fully functional, secure, and scalable web-based solution.

The first phase is **Requirement Gathering and Analysis**, where all the functional and non-functional requirements are collected from users and stakeholders. In this project, the main focus is on managing fund transactions such as grant creation, allocation, expenditure tracking, payroll, and report generation. The analysis phase identifies the data inputs, outputs, user roles, and integration requirements with other systems. The final output of this phase is a detailed requirement specification document that serves as the foundation for subsequent stages.

The second phase is **System and Architecture Design**, which defines the overall structure of the application and its components. In this stage, a **multi-tier architecture** is designed using .NET Core. The application is divided into layers such as the Presentation Layer (ASP.NET Core MVC or Web API), Business Logic Layer (services and domain models), and Data Access Layer (using Entity Framework Core). The system architecture ensures modularity, reusability, and maintainability. Database schemas and entity relationships are also designed at this stage using SQL Server or any other compatible database. Common design patterns such as Repository, Dependency Injection, and Unit of Work are applied to maintain clean code architecture.

The next stage is **Detailed Design and Prototyping**, where module-level designs are prepared, including database tables, entity classes, API endpoints, and user interface mockups. Using .NET Core tools such as Entity Framework Core and Swagger, the database structure and web APIs are created and documented. This phase helps in visualizing the system workflow and validates whether the design meets user expectations before full-scale implementation begins.

The **Implementation or Development Phase** is where actual coding takes place. The Fund Management System is developed using ASP.NET Core for web functionalities, Entity Framework Core for database interaction, and C# as the primary programming language. Each module such as Grant Creation, Fund Allocation, Expenditure, Payroll, and Report Generation is implemented following the architecture design. Standard coding practices, naming conventions, and error-handling mechanisms are followed. Dependency Injection is used to ensure low coupling between components, while asynchronous programming (async and await) is applied to improve performance. Logging and configuration management are handled using Serilog and appsettings.json files respectively.

Once development is complete, the **Testing Phase** ensures that the application performs as expected. Various types of testing such as unit testing, integration testing, and system testing are conducted. Tools like xUnit and MSTest are used to validate functionality, and Swagger helps in API-level testing. Bugs or issues found during testing are reported and fixed before moving to deployment. The main objective of this phase is to ensure accuracy, performance, and security in the system.

The **Build and CI/CD Phase** involves automating the software build, testing, and deployment processes. Using tools like GitHub Actions or Azure DevOps, a continuous integration and deployment (CI/CD) pipeline is created. The project is built, tested, and deployed automatically on each code change, ensuring faster and error-free releases. The application is packaged using Docker containers for consistent environment setup across servers.

The **Deployment Phase** makes the system live and accessible to users. The .NET Core application is deployed either on cloud platforms such as Azure App Service or on-premise servers. Entity Framework Core migrations are executed to update the database schema, and production configurations are applied. Docker and Kubernetes can also be used for scalable and containerized deployment environments.

After deployment, the **Monitoring and Maintenance Phase** ensures that the system remains stable and efficient. Application Insights, Prometheus, or ELK Stack can be used to monitor system performance,

detect errors, and analyze usage. Regular maintenance activities include bug fixing, performance tuning, database optimization, and implementing security patches to keep the system up-to-date.

The final stage is **Documentation and Handover**, where all necessary documents such as user manuals, system architecture diagrams, API documentation, and installation guides are prepared. This ensures that future developers or administrators can easily understand, maintain, and enhance the system.

In conclusion, the SDLC process in .NET Core provides a disciplined approach to software development, covering all aspects from planning to maintenance. By following these stages, the *Fund Management System* ensures a robust, maintainable, and secure solution that meets organizational goals and simplifies financial management workflows.

TESTING

TESTING STRATEGY

The testing strategy for the Fund Management System is designed to ensure that the application functions accurately, securely, and efficiently in all possible scenarios. The goal of testing is to identify and correct defects early in the development cycle, guaranteeing that the final product meets both functional and non-functional requirements. A structured testing approach is adopted, combining manual and automated testing techniques to validate the system's quality, performance, and reliability.

The process begins with Unit Testing, where individual components or modules such as Fund Allocation, Expense Tracking, and User Authentication are tested separately to ensure they perform their intended functions correctly. Each module is verified against the functional requirements using test cases designed for specific input and output conditions. NUnit or xUnit frameworks are used in the .NET Core environment for automated unit testing, allowing repetitive tests to be executed quickly and consistently.

Next, Integration Testing is performed to verify that all modules work together seamlessly. This stage ensures that the interaction between different system components—such as the database, API, and user interface—operates without conflicts or data mismatches. Integration tests focus on the flow of information across modules to detect any issues in data communication or logic handling.

After successful integration, System Testing is conducted to validate the complete Fund Management System as a unified application. This testing phase evaluates the overall functionality, user interface, database transactions, and performance under normal and peak load conditions. Testers simulate real-world usage scenarios to ensure that the system can handle concurrent users, financial transactions, and report generation without errors.

User Acceptance Testing (UAT) is the final stage, where end users—such as administrators, accountants, or fund managers—interact with the system to verify that it meets their needs and expectations. Their feedback helps identify any usability or functionality improvements before deployment.

Additionally, Non-functional Testing such as Performance Testing, Security Testing, and Compatibility Testing is carried out to ensure the system runs efficiently under various conditions. Performance testing verifies speed and scalability, while security testing ensures data integrity and protection against unauthorized access. Compatibility testing confirms that the application works across different devices, browsers, and operating systems.

TEST RESULTS AND ANALYSIS

Table 2.2.1 Access Control Policy Enforcement

TEST CASE ID	TEST CASE	ACP-01
TC001	Test Objective	To verify that only users with the assigned role (e.g., Maker, Verifier, Checker) can access their respective modules.
	Preconditions	User is logged in with valid credentials. Roles and permissions are already defined in the database.
	Test Steps	1. Login as a Maker user. 2. Try to access the Fund Approval page (restricted to Checker).
	Expected Result	Access should be denied with a message like "Access Restricted: You do not have permission to view this page."
	Actual Result	Access denied and message displayed correctly.
	Status	Pass
TEST CASE ID	TEST CASE	ACP-02
TC002	Test Objective	To ensure that department users cannot access modules belonging to another department.
	Preconditions	Multiple departments exist (e.g., Education Dept, Health Dept). User belongs to Education Dept.
	Test Steps	1. Login as Education Dept user. 2. Attempt to view fund data of the Health Dept.
	Expected Result	System should restrict access and log the attempt. A message should appear: "You are not authorized to view this

		data.”
	Actual Result	Access restricted as per policy. Attempt logged in the access log table.
	Status	Pass

Table 2.2.2 Unauthorized Access Detection and Real-Time Alerting

TEST CASE ID	TEST CASE	UAD-01
TC003	Test Objective	To verify that the system detects and logs unauthorized login attempts in real time.
	Preconditions	The system has an active alert mechanism configured.
	Test Steps	1. Attempt login with invalid credentials three times. 2. Monitor system behavior.
	Expected Result	System should lock the account temporarily and trigger a real-time alert to the admin dashboard or via email.
	Actual Result	Account locked and alert sent to admin successfully.
	Status	Pass

TEST CASE ID	TEST CASE	UAD-02
TC004	Test Objective	To check if the system triggers a real-time alert when a logged-in user tries to access a restricted module.
	Preconditions	User is logged in with Maker role; alert module is active.
	Test Steps	1. Maker user attempts to access

		Admin Settings page. 2. Monitor security alert logs.
	Expected Result	System should immediately restrict access, log the unauthorized attempt, and trigger a real-time notification/alert to the security admin.
	Actual Result	Unauthorized access blocked and alert generated instantly.
	Status	Pass

CONCLUSION AND DISCUSSION

0VERALL ANALYSIS OF PROJECT VIABILITIES

The overall analysis of project viabilities for the Fund Management System indicates that the proposed solution is technically, operationally, and economically sound. It automates and streamlines fund creation, allocation, and expenditure processes, ensuring accuracy, transparency, and security. The use of technologies like .NET Core and SQL Server provides a scalable and reliable platform for managing large datasets and user interactions efficiently. Operationally, the system simplifies workflows for departments, HODs, and agencies through intuitive interfaces and automated approvals, reducing manual errors and time delays. Economically, it minimizes administrative costs and improves resource utilization, making it a cost-effective and sustainable solution. Overall, the project is highly viable as it enhances accountability, performance, and efficiency in fund management operations.

PROBLEMS ENCOUNTERED AND POSSIBLE SOLUTION

During the development of this project, several challenges were faced, such as understanding detailed user requirements, managing database connectivity issues, and designing an intuitive user interface. Ensuring data security, maintaining system performance, and debugging runtime errors also created difficulties during the process. To resolve these problems, continuous learning through documentation, online resources, and discussions with peers was followed. Regular testing and validation techniques were applied to identify and fix errors early. Proper database structuring, use of reliable frameworks, and implementation of efficient error-handling strategies helped in overcoming technical barriers, resulting in a stable, secure, and user-friendly system.

LIMITATIONS AND FUTURES ENHANCEMENTS

Although the system is designed to be efficient and user-friendly, there are certain limitations. At present, some processes may require manual intervention, and advanced security measures such as multi-level authentication and biometric verification are not yet implemented. The system currently supports limited user roles and basic reporting features. In the future, enhancements can include integration of cloud storage for better scalability, advanced data analytics for improved decision-making, and real-time notifications for user activities. Additional features like AI-based automation, improved

UI/UX design, mobile application support, and enhanced security mechanisms can further strengthen the system, making it more powerful, flexible, and suitable for large-scale environments.

REFERENCES

1. Gupta, A. (2022). Database Management Systems and Applications. Pearson Education. : https://www.pearson.com
2. GitHub Community (2024). Open-source fund management system examples and code references. : https://github.com
3. Stack Overflow (2024). Developer community for troubleshooting and technical support. : https://stackoverflow.com
4. GoogleFirebase.(2024).FirebaseOfficialDocumentation: https://firebase.googlecom/docs